Signals and Communication Technology



Ljubiša Stanković
Ervin Sejdić *Editors*

Vertex-Frequency Analysis of Graph Signals

Springer

Signals and Communication Technology

The series “Signals and Communications Technology” is devoted to fundamentals and applications of modern methods of signal processing and cutting-edge communication technologies. The main topics are information and signal theory, acoustical signal processing, image processing and multimedia systems, mobile and wireless communications, and computer and communication networks. Volumes in the series address researchers in academia and industrial R&D departments. The series is application-oriented. The level of presentation of each individual volume, however, depends on the subject and can range from practical to scientific.

More information about this series at http://www.springer.com/series/4748

Ljubiša Stanković · Ervin Sejdić
Editors

Vertex-Frequency Analysis of Graph Signals

Springer

Editors
Ljubiša Stanković
Faculty of Electrical Engineering
University of Montenegro
Podgorica, Montenegro

Ervin Sejdić
Department of Electrical
and Computer Engineering
Swanson School of Engineering
University of Pittsburgh
Pittsburgh, PA, USA

ISSN 1860-4862 ISSN 1860-4870 (electronic)
Signals and Communication Technology
ISBN 978-3-030-03573-0 ISBN 978-3-030-03574-7 (eBook)
https://doi.org/10.1007/978-3-030-03574-7

Library of Congress Control Number: 2018961193

© Springer Nature Switzerland AG 2019
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To my beloved family for their endless support.

—Ljubiša Stanković

To my family! For your unconditional love and support.

—Ervin Sejdić

Preface

This book introduces new methods to analyze vertex-varying graph signals. In many real-world scenarios, the data sensing domain is not a regular grid, but a more complex network that consists of sensing points (vertices) and edges (relating the sensing points). Furthermore, sensing geometry or signal properties define the relation among sensed signal points. Even for the data sensed in the well-defined time or space domain, the introduction of new relationships among the sensing points may produce new insights in the analysis and result in more advanced data processing techniques. The data domain, in these cases and discussed in this book, is defined by a graph. Graphs exploit the fundamental relations among the data points. Processing of signals whose sensing domains are defined by graphs resulted in graph data processing as an emerging field in signal processing.

Although signal processing techniques for the analysis of time-varying signals are well established, the corresponding graph signal processing equivalent approaches are still in their infancy. This book presents novel approaches to analyze vertex-varying graph signals. The vertex-frequency analysis methods use the Laplacian or adjacency matrix to establish connections between the vertex domain and the spectral (frequency) domain in order to analyze local signal behavior where edge connections are used for graph signal localization. The book applies combined concepts from time-frequency and wavelet analyses of classical signal processing to the analysis of graph signals.

Covering analytical tools for vertex-varying applications, this book is of interest to researchers and practitioners in engineering, science, neuroscience, genome processing, just to name a few. It is also a valuable resource for postgraduate students and researchers looking to expand their knowledge of the vertex-frequency analysis theory and its applications.

The book consists of 15 chapters contributed by 41 leading researches in the field. Chapter "Introduction to Graph Signal Processing" provides an introduction to graph signal processing. A methodology to transform a graph into a collection of signals and back is presented in Chapter "Transformation from Graphs to Signals and Back." The spectral graph wavelet transform, along with a Chebyshev polynomial approximation-based fast implementation algorithm, is given in Chapter

"The Spectral Graph Wavelet Transform: Fundamental Theory and Fast Computation." Chapter "Spectral Design of Signal-Adapted Tight Frames on Graphs" presents a tight frame spectral design that is adapted to the graph Laplacian spectral content of a given class of graph signals. Chapter "Wavelets on Graphs via Deep Learning" introduces a machine learning framework for constructing graph wavelets that can sparsely represent a given class of signals. Chapter "Oversampled Transforms for Graph Signals" addresses the oversampled transforms for graph signals, with oversampling the graph Laplacian and the graph transforms. Chapter "Local-Set-Based Graph Signal Sampling and Reconstruction" discusses the concepts of a local set and centerless local set and describes several iterative methods to reconstruct a bandlimited graph signal from decimated data, based on these concepts. Chapter "Time-Varying Graph Signals Reconstruction" presents two batch reconstruction methods of time-varying graph signals by exploiting the smoothness of the temporal difference signals, along with their uniqueness and the reconstruction error analysis. The uncertainty principle in the context of graph signal processing is presented in Chapter "Uncertainty Principle on Graphs." In Chapter "A Filtering Framework for Time-Varying Graph Signals," three classes of filters for time-varying graph signals are given and illustrated on application examples in the design of ideal bandpass filters. Chapter "Vertex-Frequency Energy Distributions" introduces vertex-frequency energy distributions in graph signal processing. In Chapter "Shape Analysis of Carpal Bones Using Spectral Graph Wavelets," the authors present a graph signal processing approach to the shape analysis of carpal bones of the human wrist by exploiting local structure information among shape features. Chapter "Estimating the Complexity of the Cerebral Cortex Folding with a Local Shape Spectral Analysis" shows how a local spectral analysis of the cortical surface mean curvature can address the problem of the human cerebral cortex folding complexity and provides two gyrification indices by extending the concept of graph windowed Fourier transform to the framework of surfaces modeled by triangular meshes. Chapter "Wavelet-Based Visual Data Exploration" reviews how the wavelet coefficients can be visually interpreted and explores which visual analytics resources can be leveraged for two real-world problems: georeferenced urban data and interpersonal contact graphs. In Chapter "Graph-Based Wavelet Multiresolution Modeling of Multivariate Terrain Data," a multiresolution analysis framework is proposed based on a graph-based wavelet construction, which produces a sequence of intermediate resolution approximations of the terrain model.

We would like to thank to all contributors for their excellent work, comprehensive descriptions of the problems along with valuable and diverse approaches to the solutions, illustrated through numerous theoretical and real-world experimental data example. We hope that you will enjoy reading this book on the emerging topic of vertex-varying graph signal processing.

Podgorica, Montenegro — Ljubiša Stanković
Pittsburgh, PA, USA — Ervin Sejdić
September 2018

Contents

Part I Introduction

Introduction to Graph Signal Processing 3
Ljubiša Stanković, Miloš Daković and Ervin Sejdić

Part II Vertex-Frequency Theory

Transformation from Graphs to Signals and Back 111
Ronan Hamon, Pierre Borgnat, Patrick Flandrin and Céline Robardet

The Spectral Graph Wavelet Transform: Fundamental Theory and Fast Computation 141
David K. Hammond, Pierre Vandergheynst and Rémi Gribonval

Spectral Design of Signal-Adapted Tight Frames on Graphs 177
Hamid Behjat and Dimitri Van De Ville

Wavelets on Graphs via Deep Learning 207
Raif M. Rustamov and Leonidas J. Guibas

Oversampled Transforms for Graph Signals 223
Yuichi Tanaka and Akie Sakiyama

Local-Set-Based Graph Signal Sampling and Reconstruction 255
Yuantao Gu and Xiaohan Wang

Time-Varying Graph Signals Reconstruction 293
Xianghui Mao and Yuantao Gu

Uncertainty Principle on Graphs 317
Bastien Pasdeloup, Vincent Gripon, Réda Alami and Michael G. Rabbat

A Filtering Framework for Time-Varying Graph Signals 341
Addison W. Bohannon, Brian M. Sadler and Radu V. Balan

Vertex-Frequency Energy Distributions 377
Ljubiša Stanković, Miloš Daković and Ervin Sejdić

Part III Applications

Shape Analysis of Carpal Bones Using Spectral Graph Wavelets 419
Majid Masoumi, Mahsa Rezaei and A. Ben Hamza

Estimating the Complexity of the Cerebral Cortex Folding with a Local Shape Spectral Analysis 437
Hamed Rabiei, Frédéric Richard, Olivier Coulon and Julien Lefèvre

Wavelet-Based Visual Data Exploration 459
Alcebiades Dal Col, Paola Valdivia, Fabiano Petronetto, Fabio Dias, Claudio T. Silva and L. Gustavo Nonato

Graph-Based Wavelet Multiresolution Modeling of Multivariate Terrain Data 479
Teodor Cioacă, Bogdan Dumitrescu and Mihai-Sorin Stupariu

Editors and Contributors

About the Editors

Prof. Ljubiša Stanković was born in Montenegro in 1960. He received the B.S. degree in EE from the University of Montenegro (UoM) with the Best Student at the University award, winning twice the EE student competition in mathematics in Yugoslavia (1980 and 1982). He obtained the M.S. degree in communications from the University of Belgrade and the Ph.D. degree in theory of electromagnetic waves from the UoM.

As a *Fulbright* grantee, he spent 1984–1985 academic year at the Worcester Polytechnic Institute, USA. In 1997–1999, he was on leave at the Ruhr University Bochum, Germany, supported by the *Alexander von Humboldt Foundation*. At the beginning of 2001, he was at the Technische Universiteit Eindhoven, the Netherlands, as a visiting professor. He was a visiting academic at the Imperial College, London, in 2012–2013.

Since 1982, he has been on the faculty at the UoM, where he has been a full professor since 1995. He was vice president of Montenegro in 1989–1990. He was Rector of the UoM in 2003–2008 and the ambassador of Montenegro to the UK, Ireland, and Iceland from 2011 to 2015.

His current interest is in signal processing. He published several books and about 400 technical papers, more than 150 of them in the leading journals. Professor Stanković received the highest state awards of Montenegro in 1997 and 2015, for scientific achievements. He was awarded the best Signal Processing journal award in 2017 by the European Association for Signal Processing. He is a senior area editor of the *IEEE Transactions on Image Processing*, member of the Editorial Board of *Signal Processing* (Elsevier), and an associate editor of the *IET Signal Processing*. He was an associate editor of the *IEEE Signal Processing Letters* and the *IEEE Transactions on Signal Processing*.

He is a member and of the National Academy of Science and Arts of Montenegro (CANU) since 1996 (its vice president since 2016), a member of the

European Academy of Sciences and Arts (2011), and a member of the International Advisory Board of the Alexander von Humboldt Foundation. Stanković is a fellow of the IEEE since 2012 for contributions to time-frequency signal analysis.

Prof. Ervin Sejdić received B.E.Sc. and Ph.D. degrees in electrical engineering from the University of Western Ontario, London, Ontario, Canada, in 2002 and 2008, respectively. From 2008 to 2010, he was a postdoctoral fellow at the University of Toronto with a cross-appointment at Bloorview Kids Rehab, Canada's largest children's rehabilitation teaching hospital. From 2010 until 2011, he was a research fellow at Harvard Medical School with a cross-appointment at Beth Israel Deaconess Medical Center. In 2011, Prof. Sejdić joined the Department of Electrical and Computer Engineering at the University of Pittsburgh (Pittsburgh, PA, USA) as a tenure-track assistant professor. In 2017, he was promoted to a tenured associate professor. He also holds secondary appointments in the Department of Bioengineering (Swanson School of Engineering), the Department of Biomedical Informatics (School of Medicine), and the Intelligent Systems Program (Kenneth P. Dietrich School of Arts and Sciences) at the University of Pittsburgh.

Professor Sejdić is a senior member of IEEE and a recipient of many awards. As a graduate student, he was awarded two prestigious awards from the Natural Sciences and Engineering Research Council of Canada. In 2010, he received the Melvin First Young Investigator's Award from the Institute for Aging Research at Hebrew Senior Life in Boston, MA, USA. In February 2016, President Obama named Prof. Sejdić as a recipient of the Presidential Early Career Award for Scientists and Engineers, the highest honor bestowed by the US Government on science and engineering professionals in the early stages of their independent research careers. In 2017, Prof. Sejdić was awarded the National Science Foundation CAREER Award, which is the National Science Foundation's most prestigious award in support of career-development activities of those scholars who most effectively integrate research and education within the context of the mission of their organization.

From his earliest exposure to research, he has been eager to contribute to the advancement of scientific knowledge through carefully executed experiments and groundbreaking published work. He is an area editor of the *IEEE Signal Processing Magazine* and an associate editor of *Biomedical Engineering Online*. Professor Sejdić's passion for discovery and innovation drives his constant endeavors to connect advances in engineering to society's most challenging problems. Hence, his research interests include biomedical signal processing, gait analysis, swallowing difficulties, advanced information systems in medicine, rehabilitation engineering, assistive technologies, and anticipatory medical devices.

Contributors

Réda Alami Orange Labs, Lannion, France

Radu V. Balan Department of Mathematics and Center for Scientific Computation and Mathematical Modeling, University of Maryland, College Park, MD, USA

Hamid Behjat Lund University, Lund, Sweden

Addison W. Bohannon US Army Research Laboratory, Aberdeen Proving Ground, College Park, MD, USA; Center for Scientific Computation and Mathematical Modeling, University of Maryland, College Park, MD, USA

Pierre Borgnat Laboratoire de Physique, University of Lyon, Ens de Lyon, Univ Claude Bernard, CNRS, Lyon, France

Teodor Cioacă University Politehnica of Bucharest, Bucharest, Romania; University of Bucharest, Bucharest, Romania

Olivier Coulon Institut de Neurosciences de la Timone, Aix-Marseille Université, Marseille, France

Miloš Daković University of Montenegro, Podgorica, Montenegro

Alcebiades Dal Col ICMC - USP, São Carlos, Brazil

Fabio Dias University of Toronto, Toronto, Canada

Bogdan Dumitrescu University Politehnica of Bucharest, Bucharest, Romania

Patrick Flandrin Laboratoire de Physique, University of Lyon, Ens de Lyon, Univ Claude Bernard, CNRS, Lyon, France

Rémi Gribonval Univ Rennes, Inria, CNRS, IRISA, Rennes, France

Vincent Gripon IMT Atlantique, Technopole Brest-Iroise, Plouzané, France

Yuantao Gu Department of Electronic Engineering, Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing, China

Leonidas J. Guibas Computer Science Department, Stanford University, Stanford, CA, USA

L. Gustavo Nonato ICMC - USP, São Carlos, Brazil

David K. Hammond Oregon Institute of Technology - Portland Metro, Wilsonville, OR, USA

Ronan Hamon Laboratoire de Physique, University of Lyon, Ens de Lyon, Univ Claude Bernard, CNRS, Lyon, France

A. Ben Hamza Institute for Information Systems Engineering, Concordia University, Montreal, Canada

Julien Lefèvre Institut de Neurosciences de la Timone, Aix-Marseille Université, Marseille, France

Xianghui Mao Beijing National Research Center for Information Science and Technology (BNRist) and the Department of Electronic Engineering, Tsinghua University, Beijing, China

Majid Masoumi Institute for Information Systems Engineering, Concordia University, Montreal, Canada

Bastien Pasdeloup EPFL, Lausanne, Switzerland

Fabiano Petronetto Federal University of Espirito Santo, Vitória, Brazil

Michael G. Rabbat McGill University, Montréal, QC, Canada

Hamed Rabiei Institut de Neurosciences de la Timone, Aix-Marseille Université, Marseille, France; Aix Marseille Univ, CNRS, Marseille, France

Mahsa Rezaei Institute for Information Systems Engineering, Concordia University, Montreal, Canada

Frédéric Richard Aix Marseille Univ, CNRS, Marseille, France

Céline Robardet University of Lyon, Insa Lyon, Univ Claude Bernard, CNRS, LIRIS, Lyon, France

Raif M. Rustamov Data Science and AI Research, AT&T Labs Research, Bedminster, NJ, USA; Computer Science Department, Stanford University, Stanford, CA, USA

Brian M. Sadler US Army Research Laboratory, Adelphi, MD, USA

Akie Sakiyama Tokyo University of Agriculture and Technology, Fuchu, Japan

Ervin Sejdić University of Pittsburg, Pittsburg, PA, USA

Claudio T. Silva NYU, New York, USA

Ljubiša Stanković University of Montenegro, Podgorica, Montenegro

Mihai-Sorin Stupariu University of Bucharest, Bucharest, Romania

Yuichi Tanaka Tokyo University of Agriculture and Technology, Fuchu, Japan

Paola Valdivia ICMC - USP, São Carlos, Brazil

Dimitri Van De Ville Institute of Bioengineering, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland; Department of Radiology and Medical Informatics, University of Geneva, Geneva, Switzerland

Pierre Vandergheynst Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland

Xiaohan Wang Huawei Corp., Shanghai, China

Part I
Introduction

Introduction to Graph Signal Processing



Ljubiša Stanković, Miloš Daković and Ervin Sejdić

Abstract Graph signal processing deals with signals whose domain, defined by a graph, is irregular. An overview of basic graph forms and definitions is presented first. Spectral analysis of graphs is discussed next. Some simple forms of processing signal on graphs, like filtering in the vertex and spectral domain, subsampling and interpolation, are given. Graph topologies are reviewed and analyzed as well. Theory is illustrated through examples, including few applications at the end of the chapter.

1 Introduction

Graph signal processing is an active research area in recent years resulting in many advanced solutions in various applications. In numerous practical cases the signal domain is not a set of equidistant instants in time or a set of points in space on a regular grid. The data sensing domain could be irregular and, in some cases, not related to the time or space. The data sensing domain is then related to other properties of the considered system/network. For example, in many social or web related networks, the sensing points and their connectivity are related to specific objects and their links. In some physical processes other properties than the space or time coordinates define the relation between points where the signal is sensed. Even for the data sensed in the well defined time and space domain, the introduction of new relations between the sensing points may produce new insights in the analysis and result in more advanced data processing techniques.

L. Stanković (✉) · M. Daković
University of Montenegro, Podgorica, Montenegro
e-mail: ljubisa@ac.me

M. Daković
e-mail: milos@ac.me

E. Sejdić
University of Pittsburg, Pittsburg, PA, USA
e-mail: esejdic@ieee.org

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_1

The data domain, in these cases, is defined by a graph. The graph consists of vertices, where the data values are defined/sensed, and the edges connecting these vertices. Graph exploit the fundamental relations among the data based on their relevant properties. Processing of signals whose sensing domains are defined by graphs resulted in graph data processing as an emerging field in big data signal processing today. This is a big step forward from the classical time (or space) series data analysis.

Here we will present one simplified example for graph signal analysis. Assume that we measure temperature in a geographical region. The temperature sensing points are chosen according to a plan and significance of specific areas (for example, according to the population density). They are illustrated in Fig. 1.

The distance between sensing points is proportional to their distances in Fig. 1a. The measured temperature at each location is denoted by $x(n)$, Fig. 1b. It is equal

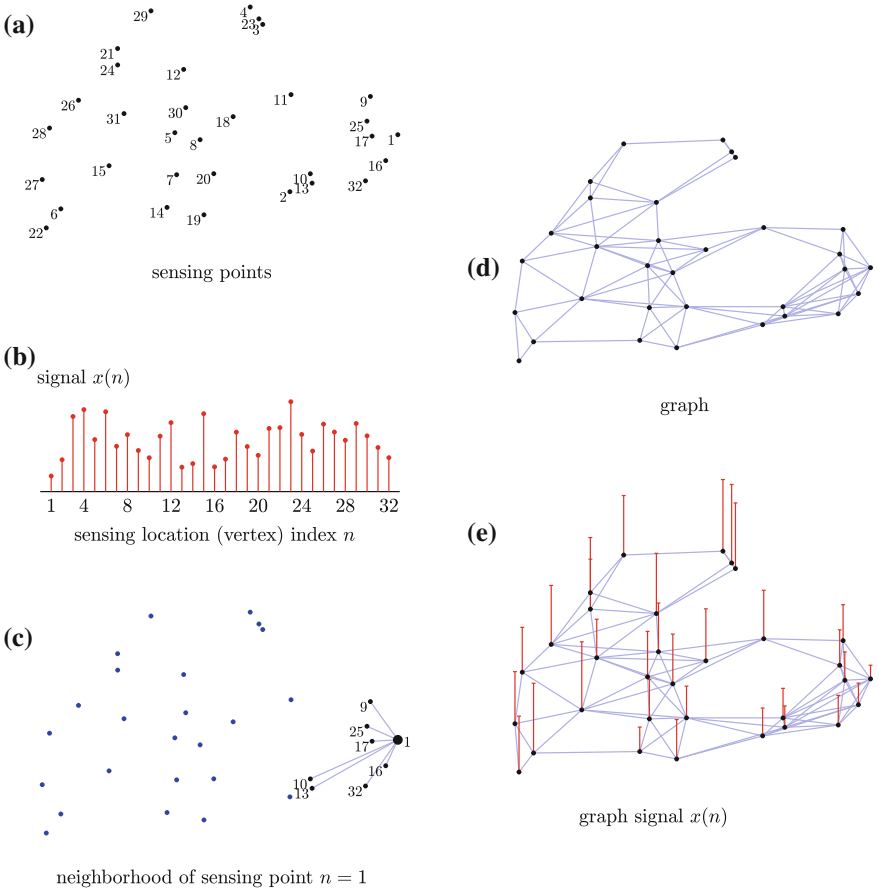


Fig. 1 Graph and a signal on the graph illustration

to $x(n) = s(n) + \varepsilon(n)$, where $s(n)$ is a temperature that would be obtained in the ideal measuring conditions and $\varepsilon(n)$ represents influence of the measurement micro-location, for example the proximity of trees, concrete structures, or ventilation. This part of the signal is called noise. We want to average the measured temperatures in order to reduce the influence of random noise to the temperature at a specific measurement point. Of course, if we average over a too large area around the considered sensing point we will lose the local temperature and produce a bias caused by very distant and probably different temperatures. Therefore, we have to average temperatures from sensing points within a small area around each of the measurement points. From the sensing locations, we can see that some points have more dense structures around them, while around some of the sensing locations the structure of measurement points is sparse. Intuitively we can conclude that it would be the best to perform averaging over a local region of each point, taking measured values from its neighborhood, as

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

Graphical illustration of this calculation formula can be obtained by using lines that connect the considered point with its neighboring points. The sensing locations included in the calculation of $y(1)$ for the sensing point $n = 1$ are shown in Fig. 1c. The matrix form of this calculation, for all sensing points, can be written as

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

where matrix $\mathbf{A}$ indicates what neighboring measurement locations should be included for each $y(n)$ calculation. This matrix will be referred to as the connectivity or adjacency matrix. Sensing locations with corresponding connections are shown in Fig. 1d. The sensing locations and their connectivity, along with the measured signal at each location, are presented in Fig. 1e.

In the calculation, we may add weights for different measurement points,

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

The weights are higher for close points and smaller for distant points. They are zero outside a specified local area. This case can be represented by lines connecting the measurement points that are included in averaging each measurement point. Each line has its weight W_{nm}. A matrix form of the weighted averaging operation is

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

If we want the averaging to be appropriately scaled, in order to produce unbiased estimates, then the sum of all summation coefficients, for each $y(n)$, should be 1. It means that we can calculate

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

where the elements of diagonal normalization matrix $\mathbf{D}$ are $D_{nn} = \sum_m W_{nm}$. This matrix is called the degree matrix.

The simple example presented here can be interpreted within the graph signal processing framework as follows:

- The measurement points are the graph vertices, Fig. 1a.
- The lines indicating mutual relation and connectivity of the measurement points are the graph edges.
- The vertices and edges form a graph, Fig. 1d. The graph is now the signal domain. It indicates the points where the signal is measured and how these sensing points are related to each other. The graph is then used for the analysis and processing of the measured data.
- The measured temperatures are the signal samples on this graph. They represent a graph signal Fig. 1e. This signal may have many realizations on the same graph. It can include noise.
- The presented local averaging, taking into account the location of measurement points, that is, taking into account the graph structure, is one simple graph signal processing algorithm (linear first-order graph system).

Now we can use this framework for many different scenarios. For example, we can perform an opinion poll among the members of a social network. The members of a social network are the vertices (measurement points). Their friendship relations are the edges, representing graph connectivity. The answers they give are the graph signal values. This is then the basis for various graph signal processing algorithms.

The graph-based data processing approach can be applied not only to technological, biological, and social networks but its application has also lead to improvements and new methods in classical signal processing. In these cases the classical signal domain (that may be represented as a linear or circular graph) is structured in a more advanced way, considering the sensing points connectivity from their properties or signal similarity points of view.

In graph signal processing, the first step is to define the graph as a signal domain. While the data sensing points (graph vertices) are commonly well defined, that is not the case with their connectivity (graph edges). In some applications, the vertex connectivity is naturally defined, resulting in an exact underlying graph topology, such as the various computer, social, road, transportation, and electrical networks. For some other cases, the data domain definition in a graph form is a part of the problem itself. The vertex connectivity is not defined in advance. It can be determined based on the properties of the sensing positions or the acquired data. The definition of appropriate graph structure is of crucial importance for a meaningful and efficient application of the graph signal processing approach.

In this chapter, we will review the basic definitions and properties of graphs. Next, the signal on a graph and basic signal processing techniques in vertex and spectral domains are described. In the third part, some graph topologies are reviewed and discussed.

2 Graphs

Graph theory as a branch in mathematics has existed for almost three centuries. It has been used for a long time in chemistry, operational research, engineering, social networks, and computer sciences. The beginning of graph theory applications in electrical engineering dates back to the mid-XIX century when Kirchoff's laws were defined. Recently, graph theory has been a rapidly developing application and research area in signal processing. A short review of the basic graph definitions and properties will be presented in this section [1–23].

2.1 Basic Definitions

A graph is defined as a set of vertices $\mathcal{V}$ and set of edges $\mathcal{B} \subset \mathcal{V} \times \mathcal{V}$ connecting the vertices, where $\times$ is a direct product operation.

Examples of graphs with $N = 8$ vertices $\mathcal{V} = \{0, 1, 2, 3, 4, 5, 6, 7\}$ are presented in Fig. 2, along with the corresponding edges. The vertices are presented as points (circles) and the edges are presented as lines. A line between vertices n and m means that $(m, n) \in \mathcal{B}$. The graph from Fig. 2b is described by

$$\mathcal{V} = \{0, 1, 2, 3, 4, 5, 6, 7\}$$
$$\mathcal{B} \subset \{0, 1, 2, 3, 4, 5, 6, 7\} \times \{0, 1, 2, 3, 4, 5, 6, 7\}$$
$$\mathcal{B} = \{(0,1),(1,3),(1,7),(2,0),(2,1),(3,2),(3,5),(4,6),(4,7),(5,3),(5,4),(6,5),(7,0),(7,1),(7,3)\}.$$

A graph can be undirected and directed. In the case of undirected graphs, as in Fig. 2a, it is assumed that the edge connecting the vertex n to the vertex m also connects the vertex m to the vertex n. This means that if $(n, m) \in \mathcal{B}$ then $(m, n) \in \mathcal{B}$.

In general, this property does not hold for directed graphs. An example of a directed graph is shown in Fig. 2b. The undirected graphs can be considered as a special case of directed graphs.

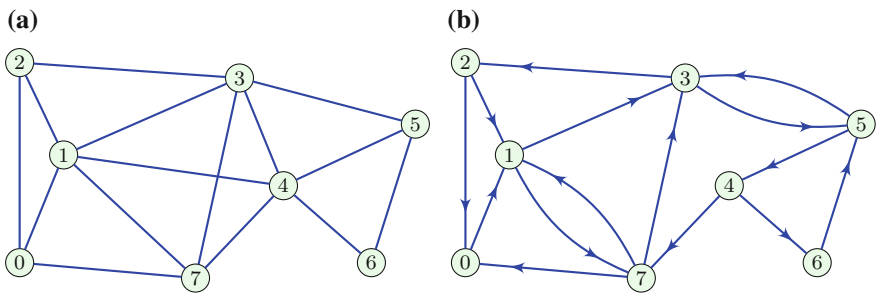


Fig. 2 Examples of: **a** Undirected graph and **b** Directed graph

For a given set of vertices and edges, the graph can be represented by an adjacency matrix **A**. This matrix describes the vertices connectivity. If there are N vertices then **A** is an $N \times N$ matrix. The elements A_{mn} of the adjacency matrix **A** assume values $A_{mn} \in \{0, 1\}$. The value $A_{mn} = 0$ is assigned if the vertices m and n are not connected with an edge, and $A_{mn} = 1$ if these vertices are connected,

$$A_{mn} = \begin{cases} 1 & \text{if } (m,n) \in \mathcal{B} \\ 0 & \text{if } (m,n) \notin \mathcal{B}. \end{cases}$$

The adjacency matrices for the graphs from Fig. 2a, b are

$$\mathbf{A} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}, \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad (1)$$

respectively.

In the case of an undirected graph the adjacency matrix is symmetric

$$\mathbf{A} = \mathbf{A}^T.$$

A graph is fully determined by its adjacency matrix, defined for a given set of vertices. If the vertex numbering is changed it will cause corresponding changes in the adjacency matrix. However vertices renumbering does not change the graph itself (these graphs are isomorphic). Relation between adjacency matrix of original and renumerated graphs is defined using a permutation matrix **P** as

$$\mathbf{A}_2 = \mathbf{P}\,\mathbf{A}_1\mathbf{P}^T. \quad (2)$$

If we change the vertex numbering as [0, 1, 2, 3, 4, 5, 6, 7] → [3, 2, 4, 5, 1, 0, 6, 7], the corresponding permutation and adjacency matrices of a graph isomorph to the graph presented in Fig. 2a are

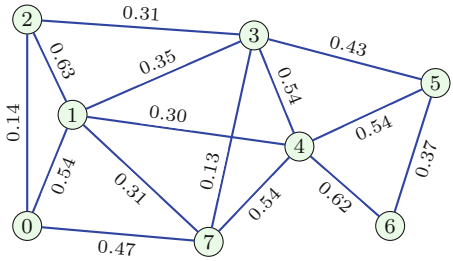


Fig. 3 An example of a weighted graph

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}. \tag{3}$$

Relation (2) can easily be checked for this example. A permutation matrix has exactly one value equal to 1 in each row and in each column.

In general, the edges can be weighted. If the weights of edges are defined, a weighted graph is obtained. The set of weights $\mathcal{W}$ corresponds to the set of edges $\mathcal{B}$. A weighted graph is a more general case than the unweighted graph. Commonly, it is assumed that the edge weights are nonnegative real numbers. If we associate weight 0 to the nonexisting edges then the graph can be described with a weight matrix $\mathbf{W}$ similar to the adjacency matrix $\mathbf{A}$. A nonzero element W_{mn} describes an edge between the vertices m and n and the corresponding weight. The value $W_{mn} = 0$ means that there is not an edge between the vertices m and n.

An example of a weighted undirected graph is presented in Fig. 3.

The weight matrix for this graph is

$$\mathbf{W} = \begin{bmatrix} 0 & 0.54 & 0.14 & 0 & 0 & 0 & 0 & 0.47 \\ 0.54 & 0 & 0.63 & 0.35 & 0.30 & 0 & 0 & 0.31 \\ 0.14 & 0.63 & 0 & 0.31 & 0 & 0 & 0 & 0 \\ 0 & 0.35 & 0.31 & 0 & 0.54 & 0.43 & 0 & 0.13 \\ 0 & 0.30 & 0 & 0.54 & 0 & 0.54 & 0.62 & 0.54 \\ 0 & 0 & 0 & 0.43 & 0.54 & 0 & 0.37 & 0 \\ 0 & 0 & 0 & 0 & 0.62 & 0.37 & 0 & 0 \\ 0.47 & 0.31 & 0 & 0.13 & 0.54 & 0 & 0 & 0 \end{bmatrix}. \tag{4}$$

In this manner the adjacency matrix $\mathbf{A}$ can be considered as a special case of the weight matrix $\mathbf{W}$ where all nonzero weights are equal to 1.

For undirected graphs the weighting matrix is symmetric,

$$\mathbf{W} = \mathbf{W}^T.$$

For directed graphs this property does not hold, in general.

A degree matrix for an undirected graph, denoted by $\mathbf{D}$, is a diagonal matrix where the diagonal elements D_{mm} are equal to the sum of weights of all edges connected with the vertex m

$$D_{mm} = \sum_n W_{mn}.$$

In the case of an unweighted and undirected graph, the value of D_{mm} is equal to the number of edges connected to the mth vertex.

The degree matrix for the graph from Fig. 3 is

$$\mathbf{D} = \begin{bmatrix} 1.15 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2.13 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.08 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.76 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.54 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.34 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.99 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.45 \end{bmatrix}. \tag{5}$$

The Laplacian matrix $\mathbf{L}$ of a graph is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W}.$$

For an undirected graph the Laplacian matrix is symmetric $\mathbf{L} = \mathbf{L}^T$.

The Laplacian for the graph from Fig. 3 is

$$\mathbf{L} = \begin{bmatrix} 1.15 & -0.54 & -0.14 & 0 & 0 & 0 & 0 & -0.47 \\ -0.54 & 2.13 & -0.63 & -0.35 & -0.30 & 0 & 0 & -0.31 \\ -0.14 & -0.63 & 1.08 & -0.31 & 0 & 0 & 0 & 0 \\ 0 & -0.35 & -0.31 & 1.76 & -0.54 & -0.43 & 0 & -0.13 \\ 0 & -0.30 & 0 & -0.54 & 2.54 & -0.54 & -0.62 & -0.54 \\ 0 & 0 & 0 & -0.43 & -0.54 & 1.34 & -0.37 & 0 \\ 0 & 0 & 0 & 0 & -0.62 & -0.37 & 0.99 & 0 \\ -0.47 & -0.31 & 0 & -0.13 & -0.54 & 0 & 0 & 1.45 \end{bmatrix}. \tag{6}$$

The normalized Laplacian is defined as

$$\mathbf{L}_N = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{W})\mathbf{D}^{-1/2}.$$

The normalized Laplacian for the graph from Fig. 3 is

$$\mathbf{L}_N = \begin{bmatrix} 1 & -0.35 & -0.13 & 0 & 0 & 0 & 0 & -0.36 \\ -0.35 & 1 & -0.42 & -0.18 & -0.13 & 0 & 0 & -0.18 \\ -0.13 & -0.42 & 1 & -0.22 & 0 & 0 & 0 & 0 \\ 0 & -0.18 & -0.22 & 1 & -0.26 & -0.28 & 0 & -0.08 \\ 0 & -0.13 & 0 & -0.26 & 1 & -0.29 & -0.39 & -0.28 \\ 0 & 0 & 0 & -0.28 & -0.29 & 1 & -0.32 & 0 \\ 0 & 0 & 0 & 0 & -0.39 & -0.32 & 1 & 0 \\ -0.36 & -0.18 & 0 & -0.08 & -0.28 & 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

2.2 Properties of Graphs and Associated Matrices

1. For an undirected graph, the matrices $\mathbf{A}$, $\mathbf{W}$, and $\mathbf{L}$ are symmetric.
2. A graph is complete if there is an edge between each pair of vertices. The adjacency matrix of a complete graph has elements $A_{mn} = 1$ for all $n \neq m$ and $A_{nn} = 0$. An example of a complete graph is presented in Fig. 4a.
3. Bipartite graph is a graph where the graph vertices $\mathcal{V}$ could be partitioned into two disjunct sets $\mathcal{E}$ and $\mathcal{H}$, $\mathcal{V} = \mathcal{E} \cup \mathcal{H}$ and $\mathcal{E} \cap \mathcal{H} = \emptyset$, such that there are no edges between vertices in the same set. If the vertex ordering is done in a such way that all vertices belonging to $\mathcal{E}$ are before vertices belonging to $\mathcal{H}$ then the adjacency matrix can be written as

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_{\mathcal{EH}} \\ \mathbf{A}_{\mathcal{HE}} & \mathbf{0} \end{bmatrix},$$

   where the matrices $\mathbf{A}_{\mathcal{EH}}$ and $\mathbf{A}_{\mathcal{HE}}$ define the connections between the vertices in set $\mathcal{E}$ and set $\mathcal{H}$. For an undirected bipartite graph $\mathbf{A}_{\mathcal{EH}} = \mathbf{A}_{\mathcal{HE}}^T$.
   An example of a bipartite undirected graph is given in Fig. 4b, with $\mathcal{E} = \{1, 2, 3\}$ and $\mathcal{H} = \{4, 5, 6, 7\}$. The graph in Fig. 4b is also a complete bipartite graph because all possible edges are present.
4. An unweighted graph is regular (or K-regular) if all vertices are with the same degree K. An example of the regular graph with $K = 4$ is given in Fig. 4c. The Laplacian and the normalized Laplacian of a K-regular graph are

$$\mathbf{L} = K\,\mathbf{I} - \mathbf{A} \text{ and } \mathbf{L}_N = \mathbf{I} - \frac{1}{K}\mathbf{A}.$$

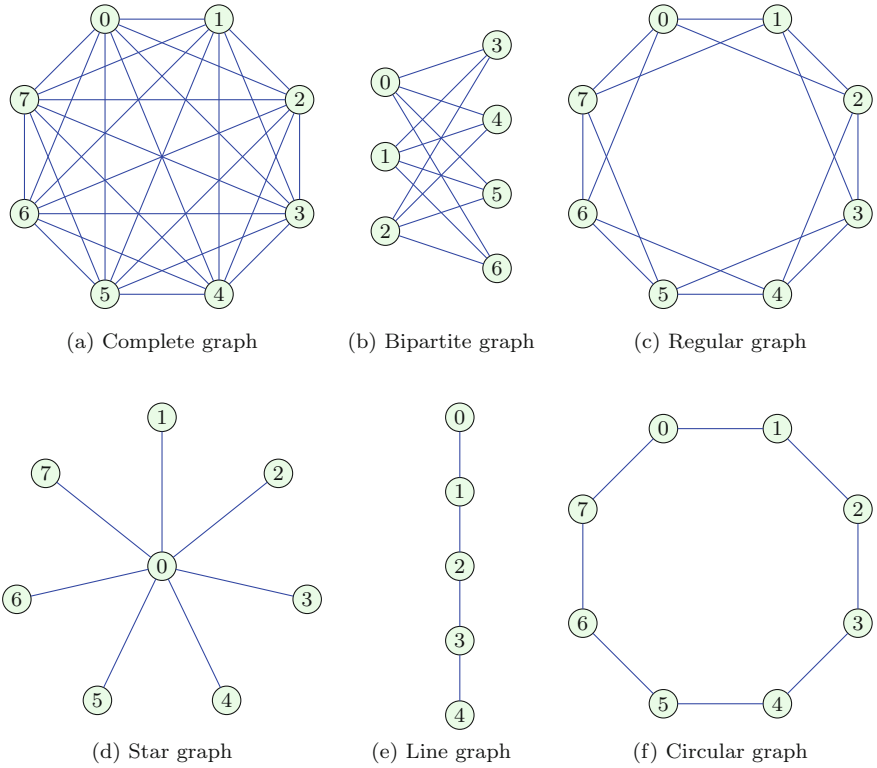


Fig. 4 Special cases: **a** complete graph with 8 vertices, **b** complete bipartite graph, **c** regular graph where each vertex is connected to 4 vertices, **d** star graph, **e** line graph, **f** circular graph

5. A star graph has one central vertex that is connected to all other vertices. There are no other edges. An example of a star graph is given in Fig. 4d. A star graph is also a complete bipartite graph where there is only one vertex in the first set.
6. A line graph is defined by a series of connected vertices. The first and the last vertex have a degree equal to 1, while all other vertices are with the degree 2. An example of a line graph with 5 vertices is presented in Fig. 4e.
7. A graph is circular if each vertex has the degree 2. This graph is also a regular graph with $K = 2$. An example of a circular graph with 8 vertices is given in Fig. 4f.
8. A walk between a vertex n and a vertex m is a connected sequence of edges and vertices that begins at the vertex n and ends at the vertex m. Edges and vertices can be included into a walk more than once.
   The length of a walk is equal to the number of included edges.
   The number of walks between the vertex n and the vertex m of the length K is equal to the element of matrix $\mathbf{A}^K$ in the nth row and the mth column.
   As an example consider vertex 1 and vertex 5 in the graph from Fig. 5. Consider the walks of length $K = 2$. There are two such walks ($1 \rightarrow 3 \rightarrow 5$ and $1 \rightarrow$

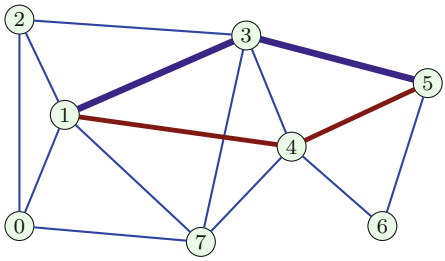


Fig. 5 Walks of length $K = 2$ from vertex 1 to vertex 5

$4 \rightarrow 5$). The element in the second row and the sixth column of matrix $\mathbf{A}^2$ is 2 indicating that there are two walks between these vertices.

$$\mathbf{A}^2 = \begin{bmatrix} 3 & 2 & 1 & 3 & \mathbf{2} & 0 & 0 & 1 \\ 2 & 5 & 2 & 3 & 2 & 2 & 1 & 3 \\ 1 & 2 & 3 & 1 & 2 & 1 & 0 & 3 \\ 3 & 3 & 1 & 5 & 3 & 1 & 2 & 2 \\ 2 & 2 & 2 & 3 & 5 & 2 & 1 & 2 \\ 0 & 2 & 1 & 1 & 2 & 3 & 1 & 2 \\ 0 & 1 & 0 & 2 & 1 & 1 & 2 & 1 \\ 1 & 3 & 3 & 2 & 2 & 2 & 1 & 4 \end{bmatrix}. \tag{8}$$

9. The number of walks between the vertices n and m of the length not higher than K is equal to the element of matrix

$$\mathbf{B}_K = \mathbf{A} + \mathbf{A}^2 + \cdots + \mathbf{A}^K$$

in the nth row and the mth column.

10. The K-neighborhood of a vertex n is defined as a set of vertices that are reachable from the vertex n in up to K length walks. It can be obtained as a set of positions of non-zero elements in the nth row of matrix $\mathbf{B}_K$. The K-neighborhoods of vertex 0 for $K = 1$ and $K = 2$ are illustrated in Fig. 6.

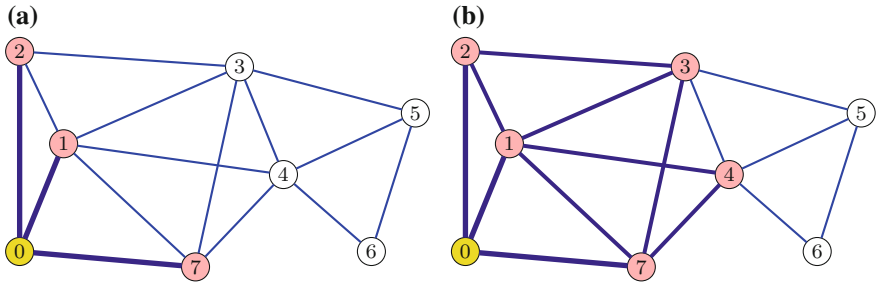


Fig. 6 The K-neighborhoods of vertex 0 for: **a** $K = 1$ and **b** $K = 2$. The neighboring vertices are shaded

11. Path is a walk where each vertex can be included only once. The path length is equal to the number of edges included in a path.
12. Distance between two vertices is equal to the minimal path length between them. The distance between vertex 1 and vertex 5 for the graph presented in Fig. 5 is 2.
13. The diameter d of a graph is equal to the largest distance between all pairs of the vertices in the graph. The diameter of a complete graph is $d = 1$.
    For the graph presented in Fig. 5 its diameter is 3.
14. An undirected graph is connected if there exists a walk between each pair of its vertices.
15. If the graph is not connected, it consists of two or more connected graphs (components). The components represent disjoint graphs. Components produce a block diagonal form of the adjacency matrix and Laplacian. For M components of a graph this form would be

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_M \end{bmatrix} \quad \text{and} \quad \mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{L}_M \end{bmatrix}.$$

    Note that the block diagonal form is obtained only if the vertex numbering follows graph components.
    As an example, let us consider a graph derived form Fig. 2a by removing some edges. This graph is presented in Fig. 7.
    The adjacency matrix for this graph is

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{9}$$

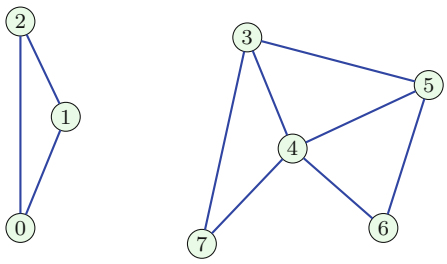


Fig. 7 A disconnected graph

with the corresponding Laplacian

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & -1 & -1 & 0 & -1 \\ 0 & 0 & 0 & -1 & 4 & -1 & -1 & -1 \\ 0 & 0 & 0 & -1 & -1 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 2 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 0 & 2 \end{bmatrix}. \tag{10}$$

These matrices are in a block-diagonal form with two blocks.
If there is an isolated vertex in a graph, then the corresponding row and column of the matrices $\mathbf{A}$ and $\mathbf{L}$ will be zero-valued.

16. If we have two graphs defined on the same vertices, with adjacency matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, we can define a sum of the graphs as a new graph with the adjacency matrix

$$\mathbf{A} = \mathbf{A}_1 + \mathbf{A}_2.$$

If we want to keep binary values 0, 1 in the adjacency matrix then the logical (Boolean) summation rule $1 + 1 = 1$ should be used in the matrix addition. In this chapter we will use the arithmetic summation rule only.

17. Kronecker (tensor) product of two disjoint graphs $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{B}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{B}_2)$ is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{B})$ where $\mathcal{V} = \mathcal{V}_1 \times \mathcal{V}_2$ and $\big((n_1, m_1), (n_2, m_2)\big) \in \mathcal{B}$ only if $(n_1, n_2) \in \mathcal{B}_1$ and $(m_1, m_2) \in \mathcal{B}_2$. The adjacency matrix of $\mathcal{G}$ is equal to the Kronecker product of adjacency matrices $\mathbf{A}_1$ and $\mathbf{A}_2$,

$$\mathbf{A} = \mathbf{A}_1 \otimes \mathbf{A}_2.$$

The Kronecker product of two simple graphs is illustrated in Fig. 8.

17. Cartesian product (graph product) of two disjoint graphs $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{B}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{B}_2)$ is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{B})$, where $\mathcal{V} = \mathcal{V}_1 \times \mathcal{V}_2$ and $\big((n_1, m_1), (n_2, m_2)\big) \in \mathcal{B}$ only if

$$m_1 = m_2 \text{ and } (n_1, n_2) \in \mathcal{B}_1 \text{ or}$$
$$n_1 = n_2 \text{ and } (m_1, m_2) \in \mathcal{B}_2.$$

The adjacency matrix of a Cartesian product of two graphs is

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

where $\mathbf{A}_1$ and $\mathbf{A}_2$ are the adjacency matrices of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively. The numbers of vertices in $\mathcal{G}_1$ and $\mathcal{G}_2$ are denoted by N_1 and N_2, respectively. The Cartesian product of two simple graphs is illustrated in Fig. 9.

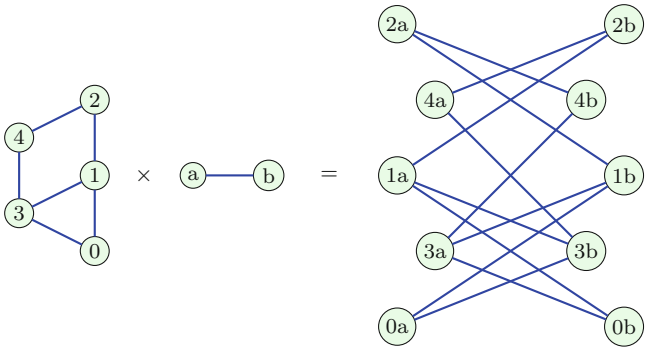


Fig. 8 Kronecker (tensor) product of two graphs

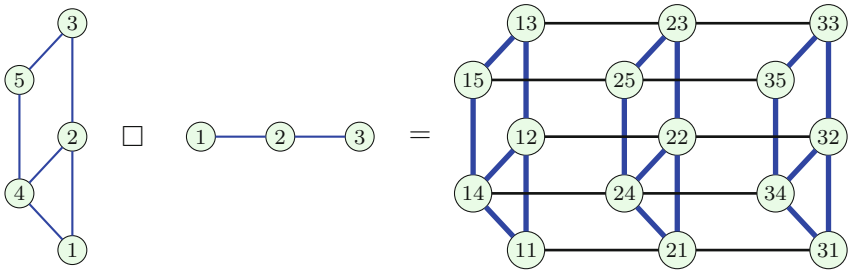


Fig. 9 Cartesian product of two graphs

2.3 Eigenvalue Decomposition of the Adjacency Matrix

Graph matrices can be decomposed using the eigenvalue decomposition. A column vector $\mathbf{u}$ is an eigenvector of the adjacency matrix $\mathbf{A}$ if

$$\mathbf{Au} = \lambda\mathbf{u}$$

holds, where the constant λ is called the eigenvalue, corresponding to the eigenvector $\mathbf{u}$.

The previous relation can be written as $(\mathbf{A} - \lambda\mathbf{I})\mathbf{u} = \mathbf{0}$. A nontrivial solution for $\mathbf{u}$ exists if

$$\det ||\mathbf{A} - \lambda\mathbf{I}|| = 0.$$

The determinant $\det ||\mathbf{A} - \lambda\mathbf{I}||$ is a polynomial of λ. It is called the characteristic polynomial of matrix $\mathbf{A}$,

$$P(\lambda) = \det ||\mathbf{A} - \lambda\mathbf{I}|| = \lambda^N + c_1\lambda^{N-1} + \cdots + c_N.$$

Order of the characteristic polynomial is equal to the number of vertices N. Eigenvalues are the roots of the characteristic polynomial, $P(\lambda) = 0$. In general, there are N eigenvalues λ_0, $\lambda_1, \ldots, \lambda_{N-1}$. In some cases the eigenvalues can be repeated meaning that the zeros of algebraic multiplicity higher than one exist in the characteristic polynomial. Denote distinct eigenvalues as $\mu_1, \mu_2, \ldots, \mu_{N_m}$, and their corresponding algebraic multiplicities as $p_1, p_2, \ldots, p_{N_m}$, where $p_1 + p_2 + \cdots + p_{N_m} = N$ is equal to the order of the considered matrix/polynomial and $N_m \leq N$ is the number of distinct eigenvalues. The characteristic polynomial can be written in a form

$$P(\lambda) = (\lambda - \mu_1)^{p_1}(\lambda - \mu_2)^{p_2} \cdots (\lambda - \mu_{N_m})^{p_{N_m}}.$$

The minimal polynomial of the considered matrix is obtained from the characteristic polynomial by reducing algebraic multiplicities of each eigenvalue to 1. Its form is

$$P_{min}(\lambda) = (\lambda - \mu_1)(\lambda - \mu_2) \cdots (\lambda - \mu_{N_m}).$$

Properties of the Characteristic and Minimal Polynomial

- The characteristic polynomial order is equal to the number of vertices.
- For $\lambda = 0$, $P(0) = \det(\mathbf{A}) = (-\lambda_0)(-\lambda_1) \cdots (-\lambda_{N-1})$.
- A sum of the eigenvalues is equal to the sum of diagonal elements of matrix $\mathbf{A}$. It means that $c_1 = 0$ for the characteristic polynomial of the adjacency matrix.
- Coefficient c_2 in $P(\lambda)$ is equal to the number of edges multiplied by -1.
- Degree of the minimal polynomial N_m is larger than the graph diameter. As an example consider a connected graph with only two distinct eigenvalues λ_0 and λ_1. The minimal polynomial order is 2. It means that the diameter of this graph is $d = 1$. This is a complete graph.
- For a connected graph the multiplicity of the largest eigenvalue is 1.

The characteristic polynomial of the adjacency matrix for the graph from Fig. 2a is

$$P(\lambda) = \lambda^8 - 15\lambda^6 - 18\lambda^5 + 33\lambda^4 + 60\lambda^3 + 16\lambda^2 - 6\lambda$$

with eigenvalues $(-2.193, -1.75, -1.321, -0.796, 0, 0.204, 1.796, 4.06)$. The minimal polynomial is equal to characteristic polynomial $P_{min}(\lambda) = P(\lambda)$.

The characteristic polynomial of the adjacency matrix for the graph from Fig. 7 is

$$P(\lambda) = \lambda^8 - 10\lambda^6 - 8\lambda^5 + 24\lambda^4 + 34\lambda^3 + 3\lambda^2 - 12\lambda - 4$$

with eigenvalues $(-\frac{1+\sqrt{5}}{2}, -1.4728, -1, -1, -0.4626, \frac{\sqrt{5}-1}{2}, 2, 2.9354)$. There is an eigenvalue of multiplicity higher than 1. The minimal polynomial is

$$P_{min}(\lambda) = \lambda^7 - \lambda^6 - 9\lambda^5 + \lambda^4 + 23\lambda^3 + 11\lambda^2 - 8\lambda - 4.$$

If all eigenvalues are distinct (of algebraic multiplicity 1), instead of N equations $\mathbf{A}\mathbf{u}_k = \lambda_k \mathbf{u}_k$, $k = 0, 1, \ldots, N-1$, we can write one matrix equation for the adjacency matrix

$$\mathbf{A}\mathbf{U} = \mathbf{U}\mathbf{\Lambda}$$

or

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1},$$

where $\mathbf{\Lambda}$ is a diagonal matrix with eigenvalues on the diagonal and $\mathbf{U}$ is a matrix composed of eigenvectors $\mathbf{u}_k$ as columns. Since one coefficient of the eigenvector can be arbitrarily chosen, common choice is such that $\|\mathbf{u}_k\|_2^2 = 1$ for each $k = 0, 1, \ldots, N-1$.

For an undirected graph, the matrix $\mathbf{A}$ is symmetric. The eigenvalues of a symmetric matrix are real-valued. For undirected graphs, if matrix $\mathbf{A}$ is diagonalizable then

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

All real symmetric matrices are diagonalizable. The adjacency matrix of an undirected graph is always diagonalizable. The square matrix is diagonalizable if all its eigenvalues are distinct (this condition is sufficient, but not necessary). For some directed graphs, when the eigenvalues of algebraic multiplicity higher than one exist, the matrix $\mathbf{A}$ may not be diagonalizable. In such cases the algebraic multiplicity is higher than the geometrical multiplicity of the considered eigenvalue and the Jordan normal form could be used.

The set of the adjacency matrix eigenvalues is called the graph adjacency spectrum.

For the graph presented in Fig. 2a, the graph adjacency spectrum is given in Fig. 10. The spectrum of the graph adjacency matrix transformed with perturbation matrix (3) is given in Fig. 11. We can see that vertex renumbering with the perturbation matrix does not change the eigenvalues. The eigenvectors are also the same with coefficient reordering induced by the vertex renumbering.

The DFT Basis Functions as a Special Case of Adjacency Matrix Eigenvectors

The eigenvalue decomposition for the directed circular graph presented in Fig. 19 will follow from the definition $\mathbf{A}\mathbf{u}_k = \lambda_k \mathbf{u}_k$. For this particular graph it reduces to

$$u_k(n-1) = \lambda_k u_k(n),$$

where $u_k(n)$ are the elements of vector $\mathbf{u}_k$. A solution of this linear difference equation is

$$u_k(n) = \frac{1}{\sqrt{N}} e^{j2\pi nk/N} \text{ and } \lambda_k = e^{-j2\pi k/N}, k = 0, 1, \ldots, N-1. \tag{11}$$

The eigenvectors correspond to the DFT basis function in this case.

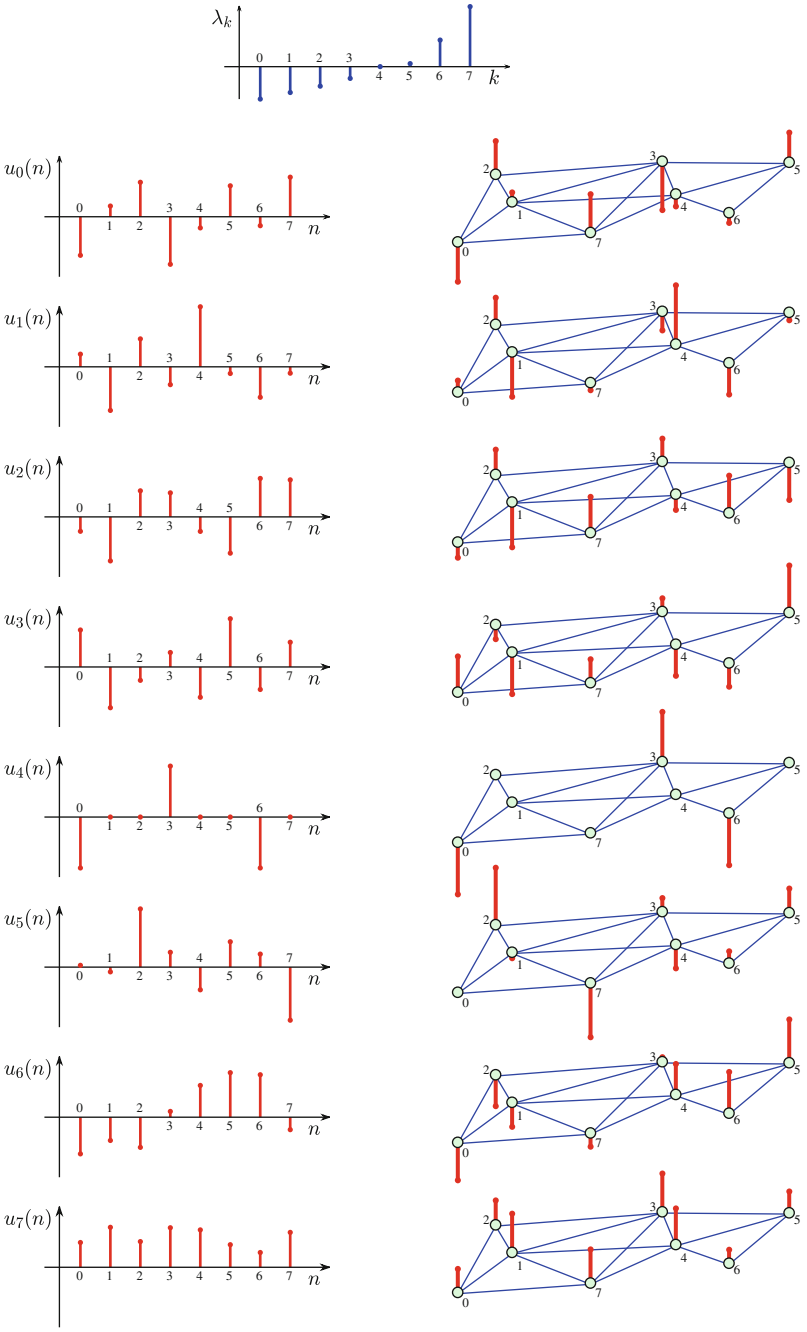


Fig. 10 Eigenvalues λ_k and corresponding eigenvectors $u_k(n)$ for the adjacency matrix of the graph presented in Fig. 2a. The eigenvectors are shown on the vertex index line (left) and on the graph (right)

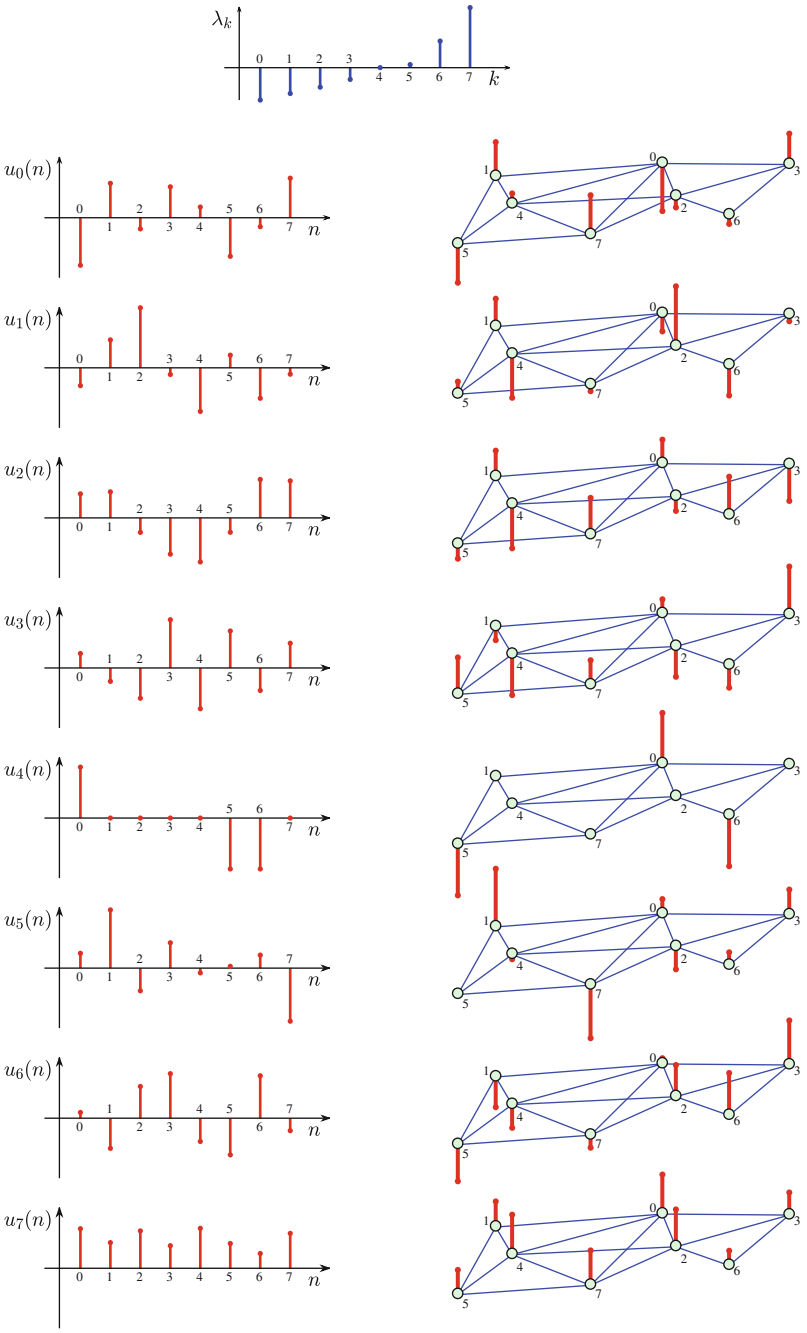


Fig. 11 Eigenvalues λ_k and corresponding eigenvectors $u_k(n)$ for the adjacency matrix of the graph presented in Fig. 2a with vertex renumbering [0, 1, 2, 3, 4, 5, 6, 7] → [3, 2, 4, 5, 1, 0, 6, 7]

Decomposition of Matrix Powers and Polynomials

The eigenvalue decomposition of the adjacency matrix $\mathbf{AA} = \mathbf{A}^2$ is

$$\mathbf{A}^2 = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1} = \mathbf{U}\boldsymbol{\Lambda}^2\mathbf{U}^{-1},$$

assuming that $\mathbf{U}^{-1}$ exists, that is, that matrix $\mathbf{A}$ is diagonalizable.

This form can easily be generalized for an arbitrary integer power

$$\mathbf{A}^n = \mathbf{U}\boldsymbol{\Lambda}^n\mathbf{U}^{-1}.$$

In general, for any matrix function $f(\mathbf{A})$ that can be written in a polynomial form

$$f(\mathbf{A}) = h_0\mathbf{A}^0 + h_1\mathbf{A}^1 + h_2\mathbf{A}^2 + \cdots + h_{N-1}\mathbf{A}^{N-1}$$

the eigenvalue decomposition is

$$f(\mathbf{A}) = \mathbf{U}f(\boldsymbol{\Lambda})\mathbf{U}^{-1}.$$

The proof is evident using the matrix power and linearity properties.

2.4 Eigenvalue Decomposition of the Laplacian

Eigenvalue decomposition can be done for the Laplacian as well. Here we will use the same notation for the eigenvalues and eigenvectors, as general mathematical forms, although they are not related to the eigenvalues and eigenvectors of the adjacency matrix. For an undirected graph the Laplacian can be written as

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T \text{ or } \mathbf{LU} = \mathbf{U}\boldsymbol{\Lambda},$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with the Laplacian eigenvalues and $\mathbf{U}$ is the matrix of its eigenvectors (as columns), with $\mathbf{U}^{-1} = \mathbf{U}^T$. Note that the Laplacian of an undirected graph is always diagonalizable since it is a real symmetric matrix.

Each eigenvector $\mathbf{u}_k$ of a Laplacian satisfies

$$\mathbf{L}\mathbf{u}_k = \lambda_k\mathbf{u}_k.$$

The set of the Laplacian eigenvalues is called the graph spectrum (or graph Laplacian spectrum).

The Laplacian spectrum of the graph from Fig. 2a is presented in Fig. 12, along with the corresponding eigenvectors. The Laplacian spectrum of the disconnected graph from Fig. 7 is plotted in Fig. 13.

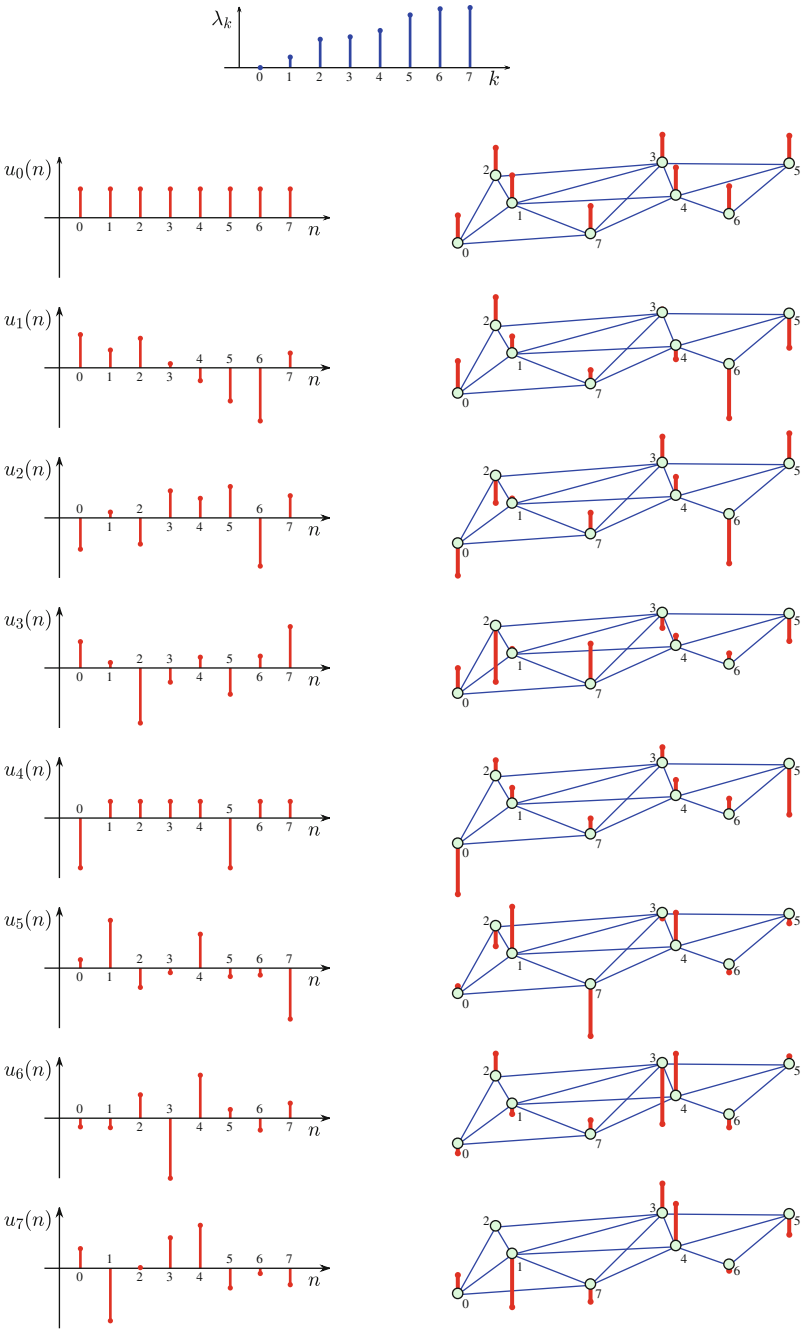


Fig. 12 Eigenvalues λ_k and corresponding eigenvectors $u_k(n)$ for the Laplacian of the graph presented in Fig. 2a. The eigenvectors are shown on the vertex index line (left) and on the graph (right)

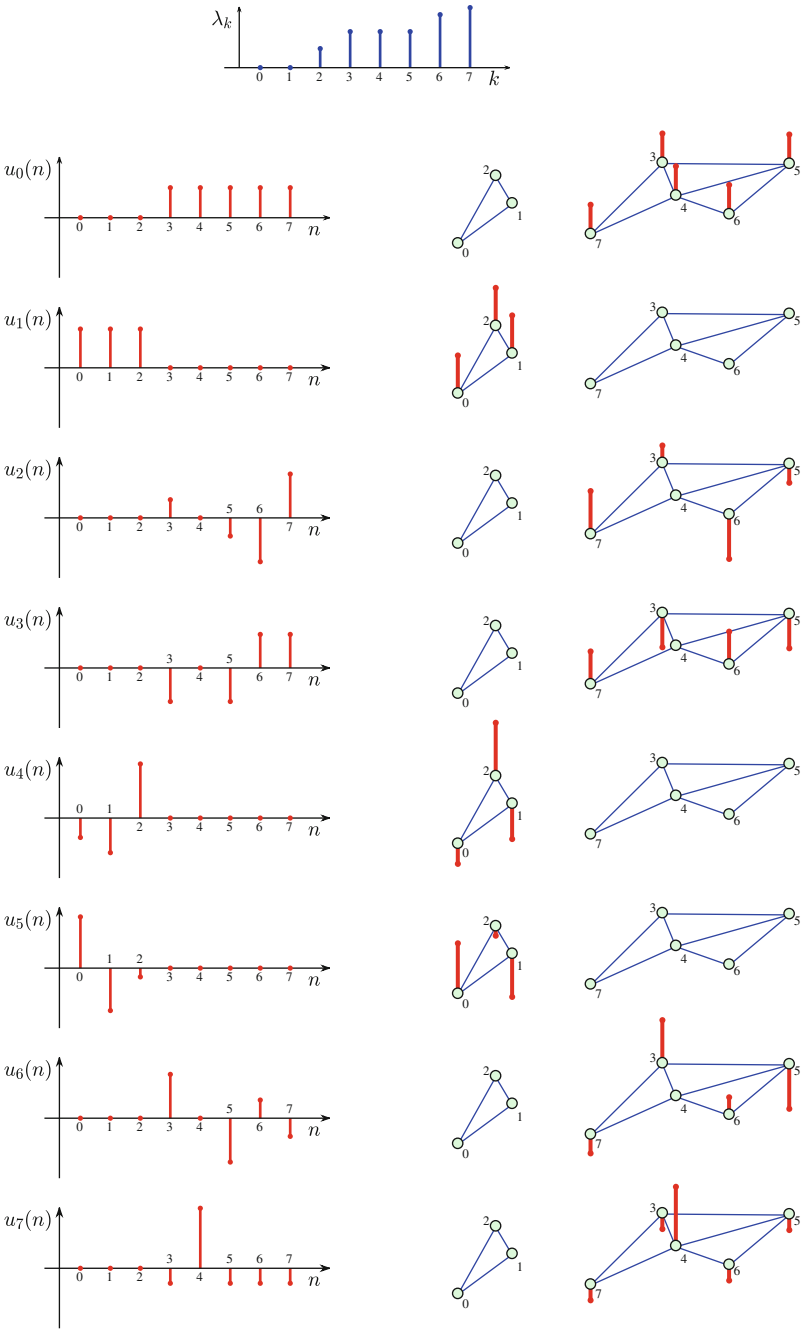


Fig. 13 Eigenvalues λ_k and corresponding eigenvectors $u_k(n)$ for Laplacian of the graph presented in Fig. 7

Properties of the Laplacian Eigenvalue Decomposition

- Since the Laplacian is defined in such a way that the sum of each row (column) elements is zero, then at least one eigenvalue of the Laplacian is zero with the corresponding eigenvector $\mathbf{u}_0 = [1, 1, \ldots, 1]^T/\sqrt{N} = \mathbf{1}/\sqrt{N}$. The relation $\mathbf{L}\mathbf{u}_0 = 0\,\mathbf{u}_0$ is always satisfied.
- Multiplicity of zero as an eigenvalue of the Laplacian is equal to the number of connected components in a graph. For example, if $\lambda_0 = \lambda_1 = 0$ then the graph is not connected. If $\lambda_2 > 0$ then there are two connected components in this graph.
- Sum of the eigenvalues is equal to the trace of the Laplacian matrix. For the normalized Laplacian the sum of its eigenvalues is equal to N if there are no isolated vertices.
- Coefficient c_N in the Laplacian characteristic polynomial

$$P(\lambda) = \det ||\mathbf{L} - \lambda\mathbf{I}|| = \lambda^N + c_1\lambda^{N-1} + \cdots + c_N$$

  is equal to 0. Coefficient c_1 is equal to the number of edges multiplied by -2.
  The characteristic polynomial of the Laplacian for graph from Fig. 2a is

$$P(\lambda) = \lambda^8 - 30\lambda^7 + 374\lambda^6 - 2500\lambda^5 + 9618\lambda^4 - 21106\lambda^3 + 24094\lambda^2 - 10712\lambda$$

  with eigenvalues (0, 1.134, 3.054, 3.317, 4, 5.679, 6.342, 6.473). In this case all eigenvalues are of multiplicity one so the minimal polynomial is equal to characteristic polynomial $P_{min}(\lambda) = P(\lambda)$.
  The characteristic polynomial of the Laplacian for the graph from Fig. 7 is

$$P(\lambda) = \lambda^8 - 20\lambda^7 + 163\lambda^6 - 692\lambda^5 + 1611\lambda^4 - 1944\lambda^3 + 945\lambda^2.$$

  with eigenvalues $(0, 0, 3 - \sqrt{2}, 3, 3, 3, 3 + \sqrt{2}, 5)$. Eigenvalue 0 is of multiplicity 2 and eigenvalue 3 is of multiplicity 3, so the minimal polynomial is

$$P_{min}(\lambda) = \lambda^5 - 14\lambda^4 + 70\lambda^3 - 146\lambda^2 + 105\lambda$$

- Graphs with the same spectrum are called isospectral or cospectral graphs. Construction of isospectral graphs that are not isomorph is an interesting topic in graph theory. A complete graph is uniquely defined by its spectrum.
- For a K-regular graph the eigenvectors of the Laplacian and adjacency matrix are the same, while for the eigenvalues the relation

$$\lambda_k^{(L)} = K - \lambda_k^{(A)}.$$

  holds. It follows from $\mathbf{U}^T\mathbf{L}\mathbf{U} = \mathbf{U}^T(K\mathbf{I} - \mathbf{A})\mathbf{U}$.
- Eigenvalues of the normalized Laplacian $\mathbf{L}_N = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ satisfy the relation

$$0 \leq \lambda \leq 2.$$

The upper bound equality holds if and only if the graph is a bipartite graph.

- The eigenvalues and eigenvectors of the normalized Laplacian of a bipartite graph with vertices $\mathcal{E}$ and $\mathcal{H}$ satisfy the relation (graph spectrum folding)

$$\lambda_k = 2 - \lambda_{N-k}$$
$$\mathbf{u}_k = \begin{bmatrix} \mathbf{u}_{\mathcal{E}} \\ \mathbf{u}_{\mathcal{H}} \end{bmatrix} \text{ and } \mathbf{u}_{N-k} = \begin{bmatrix} \mathbf{u}_{\mathcal{E}} \\ -\mathbf{u}_{\mathcal{H}} \end{bmatrix}, \tag{12}$$

where $\mathbf{u}_k$ is the eigenvector and $\mathbf{u}_{\mathcal{E}}$ is its part indexed on the first set of vertices, while $\mathbf{u}_{\mathcal{H}}$ is the part of eigenvector $\mathbf{u}_k$ indexed on the second set of vertices.
In order to prove this property, we can write the adjacency and the normalized Laplacian matrices of an undirected bipartite graph in block forms

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{A}_{\mathcal{EH}} \\ \mathbf{A}_{\mathcal{EH}}^T & \mathbf{0} \end{bmatrix} \quad \text{and} \quad \mathbf{L}_N = \begin{bmatrix} \mathbf{I} & \mathbf{L}_{\mathcal{EH}} \\ \mathbf{L}_{\mathcal{EH}}^T & \mathbf{I} \end{bmatrix}.$$

The eigenvalue relation is

$$\mathbf{L}_N \mathbf{u}_k = \begin{bmatrix} \mathbf{u}_{\mathcal{E}} + \mathbf{L}_{\mathcal{EH}} \mathbf{u}_{\mathcal{H}} \\ \mathbf{L}_{\mathcal{EH}}^T \mathbf{u}_{\mathcal{E}} + \mathbf{u}_{\mathcal{H}} \end{bmatrix} = \lambda_k \begin{bmatrix} \mathbf{u}_{\mathcal{E}} \\ \mathbf{u}_{\mathcal{H}} \end{bmatrix}.$$

From this relation we get $\mathbf{L}_{\mathcal{EH}} \mathbf{u}_{\mathcal{H}} = (\lambda_k - 1)\mathbf{u}_{\mathcal{E}}$ and $\mathbf{L}_{\mathcal{EH}}^T \mathbf{u}_{\mathcal{E}} = (\lambda_k - 1)\mathbf{u}_{\mathcal{H}}$, resulting in

$$\mathbf{L}_N \begin{bmatrix} \mathbf{u}_{\mathcal{E}} \\ -\mathbf{u}_{\mathcal{H}} \end{bmatrix} = (2 - \lambda_k) \begin{bmatrix} \mathbf{u}_{\mathcal{E}} \\ -\mathbf{u}_{\mathcal{H}} \end{bmatrix}.$$

It completes the proof of the property.

Fourier Analysis as a Special Case of the Laplacian Spectrum

For the undirected circular graph from Fig. 4f the eigenvalues of the Laplacian are

$$\lambda_k = \begin{cases} 2 - 2\cos(\pi(k+1)/N) & \text{for odd } k \\ 2 - 2\cos(\pi k/N) & \text{for even } k. \end{cases}$$

Note that for $k = 0$ we have $\lambda_0 = 0$. Most of the eigenvalues are of algebraic multiplicity 2, i.e., $\lambda_1 = \lambda_2$, $\lambda_3 = \lambda_4$, and so on. If N is odd then $\lambda_{N-2} = \lambda_{N-1}$. For an even N we have $\lambda_{N-1} = 2$ of algebraic multiplicity 1.

The corresponding eigenvectors $\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}$, are

$$u_k(n) = \begin{cases} \sin(\pi(k+1)n/N) & \text{for odd } k,\ k < N-1 \\ \cos(\pi k n/N) & \text{for even } k \\ \cos(\pi n) & \text{for odd } k,\ k = N-1, \end{cases} \tag{13}$$

where $k = 0, 1, \ldots, N-1$ and $n = 0, 1, \ldots, N-1$.

Note that an arbitrary linear combination of eigenvectors $\mathbf{u}_{2k-1}$ and $\mathbf{u}_{2k}$, $1 \leq k < N/2$ is also an eigenvector since the corresponding eigenvalues are equal. Having this fact in mind we can write an alternative set of the eigenvectors as

$$u_k(n) = \begin{cases} 1 & \text{for } k = 0 \\ \cos(\pi(k+1)n/N) + j\sin(\pi(k+1)n/N) & \text{for odd } k,\ k < N-1 \\ \cos(\pi kn/N) - j\sin(\pi kn/N) & \text{for even } k,\ k > 0 \\ \cos(\pi n) & \text{for odd } k,\ k = N-1, \end{cases}$$

where $j^2 = -1$. It can be easily checked that this set of eigenvectors is orthonormal and the eigenvectors correspond to the DFT basis functions.

2.5 *Vertex Ordering, Coloring, and Segmentation*

The ordering of vertices of a graph can be arbitrary. This is an important difference from classical signal processing where the ordering is assumed and inherent. Any change or data ordering would produce significant changes in the classical signal processing results, in general. In the previous section we have seen (Figs. 10 and 11) that a reordering of the vertices will imply corresponding indices reordering within each eigenvector.

However, the presentation of a graph signal, in any other form than the presentation which uses the graph as the domain, would benefit from an appropriate vertex ordering. This is of particular importance in vertex-frequency graph signal representations.

Here we will describe a possible method of vertex ordering. It is based on the Laplacian eigendecomposition. The aim of this vertex ordering is to get the smoothest possible representation of the eigenvectors, corresponding to low eigenvalues, if the vertices are represented sequentially. The smoothness of a graph signal can be defined using the Laplacian quadratic form. The Laplacian quadratic form of an eigenvector is equal to the corresponding eigenvalue

$$\mathbf{u}_k^T(\mathbf{L}\mathbf{u}_k) = \mathbf{u}_k^T(\lambda_k\mathbf{u}_k) = \lambda_k.$$

This will be discussed in details in Sect. 3.6. The eigenvector corresponding to $\lambda_0 = 0$ is constant (maximally smooth for any vertex ordering) and it is not appropriate for the vertex ordering. The next smoothest eigenvector is $\mathbf{u}_1$ with eigenvalue λ_1. The aim here is to order vertices in a such way that the presentation of this vector, as a function of the vertex index, is also maximally smooth. This can be achieved by sorting the values of $\mathbf{u}_1$ into a nondecreasing order (the second eigenvalue λ_1 is called the Fiedler value or algebraic connectivity, while the corresponding eigenvector $\mathbf{u}_1$ is known as the Fiedler vector). The order of vertices in the sorted $\mathbf{u}_1$ corresponds to its smoothest presentation.

As an example, consider the vector $\mathbf{u}_1$ in Fig. 12. We see that there are big changes of the subsequent values of $u_1(n)$ with the presented vertex ordering, for example, the change from $u_1(6)$ to $u_1(7)$ is significant. The representation of $u_1(n)$ would be smoother if we sort the values of $u_1(n)$ and appropriately reorder the vertices. This kind of reordering for the graph presented in Fig. 12, would produce vertex order

$$[6, 5, 4, 3, 7, 1, 2, 0]$$

instead of the presented order $[0, 1, 2, 3, 4, 5, 6, 7]$. With this vertex ordering, the presentation of $u_1(n)$ would be smoother.

In general, the spectral similarity of vertices can be defined using more than one eigenvector. Coefficients $u_k(n)$, $k = 0, 1, \ldots, N-1$ are assigned to the vertex n. We can assign an N dimensional spectral vector

$$\mathbf{q}_n = [u_0(n), u_1(n), \ldots, u_{N-1}(n)]^T$$

to each vertex n. If $\mathbf{u}_0$ is omitted then $\mathbf{q}_n = [u_1(n), \ldots, u_{N-1}(n)]^T$.

The spectral similarity between vertices n and m is defined using norm-two $\|\mathbf{q}_n - \mathbf{q}_m\|_2$. We can restrict spectral similarity to a few lower-order (smooth) spectral coefficients. If we restrict the spectral similarity to the two (or three) smoothest coefficients then the spectral vector is $\mathbf{q}_n = [u_1(n), u_2(n)]^T$ in the two-dimensional case (or $\mathbf{q}_n = [u_1(n), u_2(n), u_3(n)]^T$ for the three dimensional case).

From this point we can proceed in two ways:

- The first one is to keep the original vertex positions and to color them according to the spectral vector $\mathbf{q}_n$. Single color vertex coloring using values of $\mathbf{u}_1$ for the vertex color intensity is presented in Fig. 14. Based on this coloring we can clearly see three graph segments, $\{0, 1, 2\}$, $\{3, 4, 7\}$, and $\{5, 6\}$. Three color vertex coloring, using $\mathbf{u}_2$, $\mathbf{u}_3$, and $\mathbf{u}_4$ for the Minnesota graph is given in Fig. 15. Eigenvectors $\mathbf{u}_0$ and $\mathbf{u}_1$ are omitted in the Minnesota graph case, since corresponding eigenvalues are $\lambda_0 = \lambda_1 = 0$. The graph segmentation can be done by grouping vertices with similar colors, using appropriate thresholds, and assigning the vertices from each group to segments (with constant colors).
- Another approach is to use the spectral vector $\mathbf{q}_n$ as a position of a vertex in a new space (Laplacian eigenmaps or LE). In the case of two or three dimensional spectral

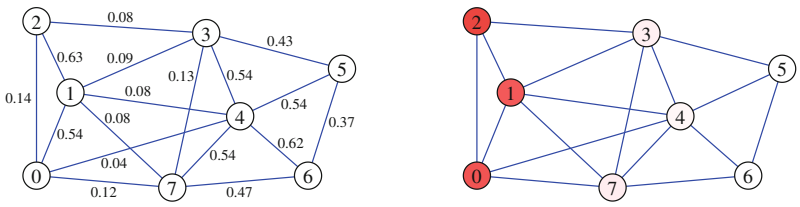


Fig. 14 Vertex coloring in a graph with some weak connections using the Laplacian eigenvector $\mathbf{u}_1$ and corresponding intensities of red color



Fig. 15 Vertex three-dimensional coloring in the Minnesota road-map graph using the Laplacian eigenvectors $\{\mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4\}$ as the coordinates in the RGB coloring system

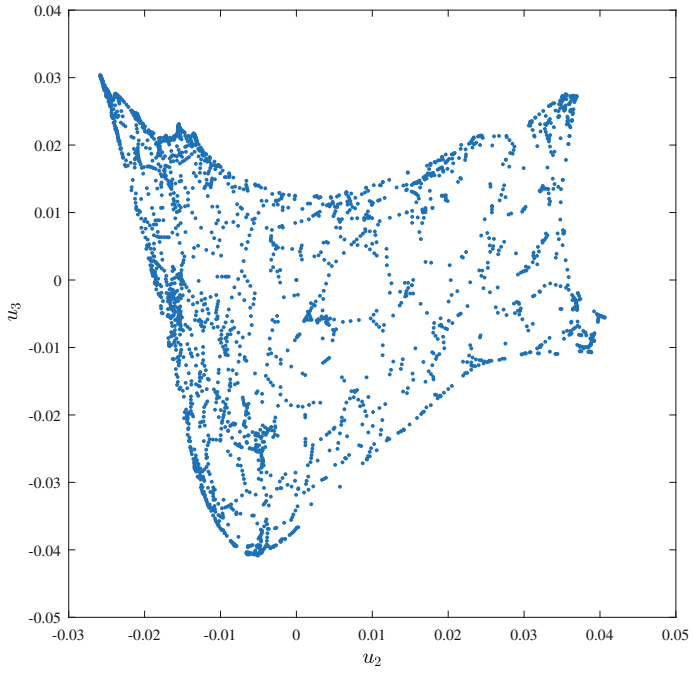


Fig. 16 The Minnesota road-map graph with new two-dimensional vertex positions defined by the Laplacian eigenvectors $\{\mathbf{u}_2, \mathbf{u}_3\}$ as the vertex coordinates (the 2D Laplacian eigenmap)

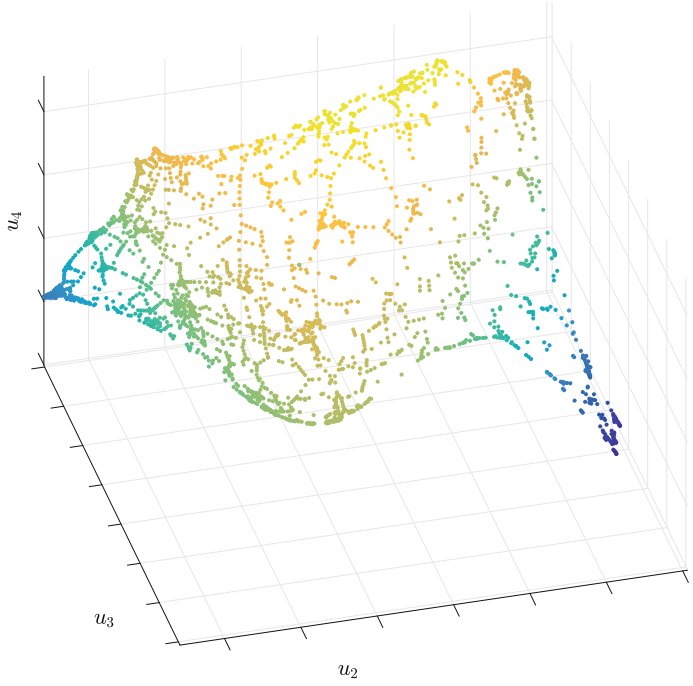


Fig. 17 The Minnesota road-map graph with new thee-dimensional vertex positions defined by the Laplacian eigenvectors $\{\mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4\}$ as the vertex coordinates (the 3D Laplacian eigenmap)

vectors, this approach can be used to graphically present vertex spectral similarity. For the Minnesota graph, the Laplacian eigenmap for the two-dimensional case is given in Fig. 16, and for the three-dimensional case in Fig. 17.

3 Signals on Graphs

In classical signal processing, signal is sampled at successive, equally spaced, time instants. The ordering of signal samples is then obvious with $x(n)$ being preceded by $x(n-1)$ and succeeded by $x(n+1)$. The time distance between samples is considered as the basic parameter in various processing algorithms. This relation between sampling instants can be represented in a graph form. The vertices correspond to the instants when the signal is sampled and the edges define sampling (vertex) ordering. The fact that sampling instants are equally spaced can be represented with same weights for all edges (for example normalized to 1). Graphical illustration of this signal is given in Fig. 18.

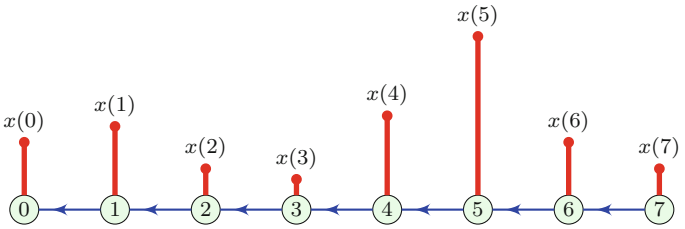


Fig. 18 Graph representation of a classical time-domain signal

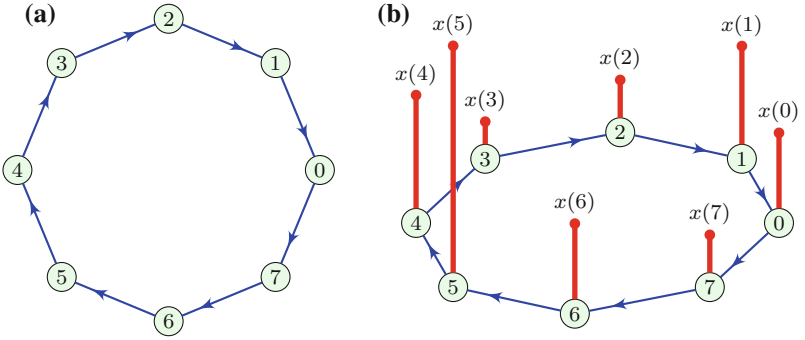


Fig. 19 **a** A circular graph. **b** A periodic signal on a graph. Signal values are presented as vertical lines starting from the corresponding vertex

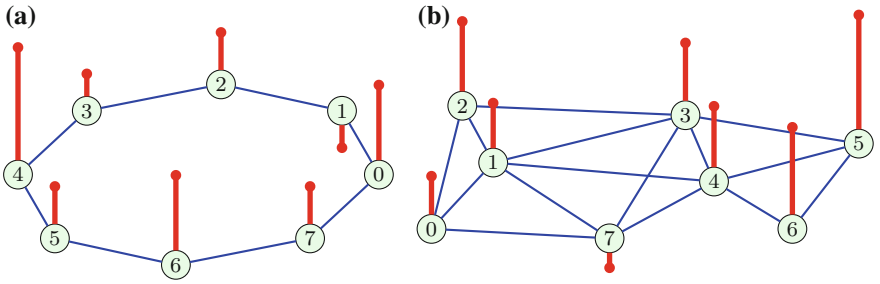


Fig. 20 **a** A signal on an undirected circular graph. **b** Undirected arbitrary graph. Signal values are presented as vertical lines starting from the corresponding vertex

In digital signal processing algorithms, periodicity of the analyzed signals is usually assumed, meaning that sample $x(N-1)$ is succeeded by $x(0)$. This case correspond to the circular graph, Fig. 19. This model is used in many common transforms, like DFT, DCT, wavelets, and corresponding processing algorithms based on these transforms.

Signal on a graph is defined by associating real (or complex) values $x(n)$ to each vertex, Fig. 20. Signal values can be arranged in a vector form

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

The graph is considered as a generalized signal domain.

In general, any linear processing of a graph signal at a vertex n can be defined as a linear combination of the signal value $x(n)$ at this vertex and the signal samples $x(m)$ at vertices around this vertex

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

where $\mathcal{V}_n$ is the set of vertices in the neighborhood of vertex n. This form is highly vertex-varying. Only in a specific case of regular graphs can it be vertex invariant. Then $\mathcal{V}_n$ is a K-neighborhood of the vertex n with $h(n,m) = h(n-m)$.

For a general graph we can define a vertex-invariant filtering function, using shifts on a graph. Various forms of signal shifts on a graph will be introduced in the next sections. They are used to introduce efficient graph signal processing methods [24–38].

3.1 Adjacency Matrix and Graph Signal Shift

Consider a graph signal $\mathbf{x}$. Its sample at a vertex n is $x(n)$. The signal shift on a graph can be defined as the movement of the signal sample from the vertex n along all walks, with the length equal to one. The movement is done for all vertices. The signal shifted in this way is denoted by $\mathbf{x}_1$. Its values can be defined using the graph adjacency matrix as

$$\mathbf{x}_1 = \mathbf{A}\mathbf{x} \tag{14}$$

As an illustration of a signal and its shifted version, consider classical signal processing, where the adjacency matrix is defined by graph Fig. 19a. The original signal $\mathbf{x}$ is presented in Fig. 21a. The shifted version of this signal $\mathbf{x}_1$ is shown in Fig. 21b. Two simple signals on an undirected graph are presented on the left of Fig. 22a. The corresponding shifted signals with $\mathbf{x}_1 = \mathbf{A}\mathbf{x}$ are presented on the right of Fig. 22b.

A signal shifted by two is obtained by a shift for one of the shifted signals. The resulting, twice shifted, signal is

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

In general, a graph signal shifted for m is obtained as a shift by one of the graph signal already shifted for $m-1$

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

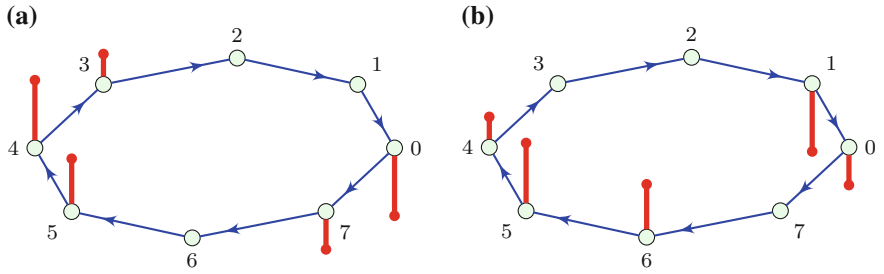


Fig. 21 **a** A signal on the directed circular graph. **b** A shifted version of the graph signal from **a**

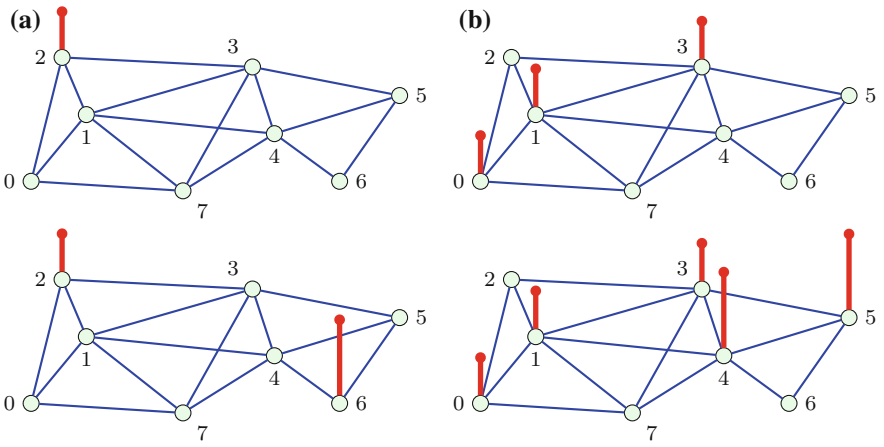


Fig. 22 **a** Two simple signals on an undirected graph. **b** Shifted versions of the graph signals from **a**

3.2 Systems Based on a Graph Shifted Signals

A system on a graph can be implemented as a linear combination of a graph signal and its graph shifted versions. The output signal from a system on a graph can be written as

$$\mathbf{y} = h_0\mathbf{A}^0\,\mathbf{x} + h_1\mathbf{A}^1\,\mathbf{x} + \cdots + h_{M-1}\mathbf{A}^{M-1}\,\mathbf{x} = \sum_{m=0}^{M-1} h_m\mathbf{A}^m\,\mathbf{x} \tag{15}$$

where $\mathbf{A}^0 = \mathbf{I}$, by definition.

The system coefficients are h_0, $h_1, \ldots, h_{M-1}$. For a circular (classical signal processing) graph this relation reduces to the well known FIR filter,

$$y(n) = h_0 x(n) + h_1 x(n-1) + \cdots + h_{M-1} x(n-M+1). \tag{16}$$

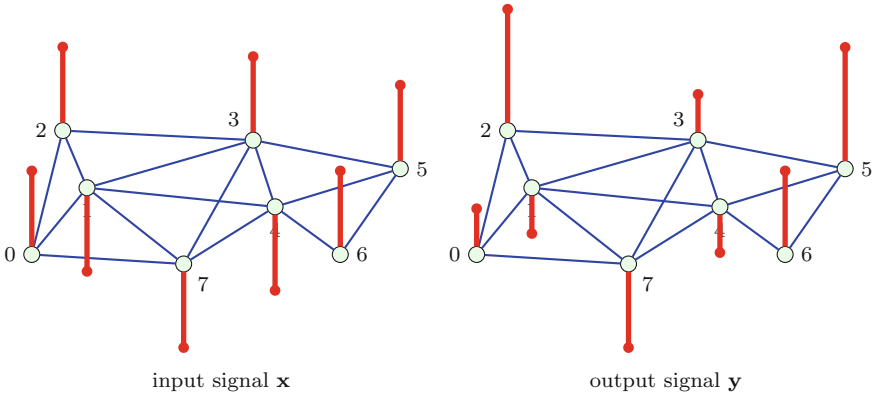


Fig. 23 A vertex domain signal filtering example. An input graph signal (left) and the output signal obtained as $\mathbf{y} = 1.0\,\mathbf{x} + 0.5\,\mathbf{Ax}$ (right)

Having in mind that the matrix $\mathbf{A}^m$ describes walks of the length m in a graph, the output graph signal $y(n)$ is calculated as a linear combination of the input graph signal values within $M-1$ neighborhood of the considered vertex n.

It is obvious that the system order $M-1$ should be lower than the number of vertices N in the case when the minimal and characteristic polynomial are of the same degree. In general, the system order $M-1$ should be lower than the degree N_m of the minimal polynomial of the adjacency matrix $\mathbf{A}$.

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

If the system order is higher or equal to the degree of minimal polynomial $M-1 \geq N_m$ then there exist more than one system producing the same output signal for a given input signal. All of these systems are called equivalent. This topic will be addressed in Sect. 3.4.4 dealing with the filter design in the spectral domain.

As an example consider a graph signal Fig. 23(left) and a linear system on this graph with coefficients $h_0 = 1$, $h_1 = 0.5$. The output graph signal is presented in Fig. 23(right).

In general, a system on a graph is defined in the vertex domain by

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

This system is linear since

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

A system on a graph is shift invariant if

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

A system on a graph defined by

$$H(\mathbf{A}) = h_0\mathbf{A}^0 + h_1\mathbf{A}^1 + \cdots + h_{M-1}\mathbf{A}^{M-1} \tag{17}$$

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

3.3 Graph Fourier Transform Based on the Adjacency Matrix

In the classical signal analysis the signals are often analyzed and processed in the spectral (Fourier) domain. The spectral domain approach to signal processing has lead to many simple and efficient algorithms in classical signal processing.

The spectral analysis and processing approach can be extended to the graph signals as well. Spectral domain representations of the graph signals can be based on the adjacency matrix or Laplacian spectral decomposition. Both of these approaches will be described in this and the next section, respectively.

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

$$\mathbf{X} = \mathbf{U}^{-1}\mathbf{x} \tag{18}$$

where $\mathbf{U}$ is a matrix with eigenvectors of the adjacency matrix in its columns. Denote elements of vector $\mathbf{X}$ as $X(k)$, for $k = 0, 1, \ldots, N-1$. If $\mathbf{U}^{-1} = \mathbf{U}^T$ the element $X(k)$ is a projection of the considered signal to the kth eigenvector, as a graph signal decomposition basis function,

$$X(k) = \sum_{n=0}^{N-1} x(n)u_k(n). \tag{19}$$

Therefore the graph Fourier transform can be understood as a signal decomposition onto the set of eigenvectors as orthonormal basis functions.

The inverse graph Fourier transform is obtained as

$$\mathbf{x} = \mathbf{U}\,\mathbf{X} \tag{20}$$

or

$$x(n) = \sum_{k=0}^{N-1} X(k)u_k(n). \tag{21}$$

In the case of a circular graph from Fig. 19, this transform reduces to the standard discrete Fourier transform (DFT), Eq. (11). It is the reason why the transform (19) and its inverse (21) are referred to as the graph discrete Fourier transform (GDFT) and inverse graph discrete Fourier transform (IGDFT).

Consider a system on a graph (15). If we use the spectral representation of the adjacency matrix $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ we will get

$$\mathbf{y} = h_0\mathbf{U}\mathbf{\Lambda}^0\mathbf{U}^{-1}\mathbf{x} + h_1\mathbf{U}\mathbf{\Lambda}^1\mathbf{U}^{-1}\mathbf{x} + \cdots + h_{M-1}\mathbf{U}\mathbf{\Lambda}^{M-1}\mathbf{U}^{-1}\mathbf{x} \tag{22}$$

or

$$\mathbf{y} = \mathbf{U}\big(h_0\mathbf{\Lambda}^0 + h_1\mathbf{\Lambda}^1 + \cdots + h_{M-1}\mathbf{\Lambda}^{M-1}\big)\mathbf{U}^{-1}\mathbf{x} = \mathbf{U}\,H(\mathbf{\Lambda})\mathbf{U}^{-1}\mathbf{x}. \tag{23}$$

By left multiplying this relation with $\mathbf{U}^{-1}$ we get

$$\mathbf{U}^{-1}\mathbf{y} = H(\mathbf{\Lambda})\mathbf{U}^{-1}\mathbf{x} \tag{24}$$

If the GDFTs of the input and output graph signal

$$\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}, \qquad \mathbf{Y} = \mathbf{U}^{-1}\mathbf{y}$$

are used, we will obtain the spectral domain system relation as

$$\mathbf{Y} = H(\mathbf{\Lambda})\,\mathbf{X} \tag{25}$$

or

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

The transfer function of a system on a graph is defined by

$$H(\lambda_k) = \frac{Y(k)}{X(k)} = (h_0 + h_1\lambda_k + \cdots + h_{M-1}\lambda_k^{M-1}). \tag{26}$$

The classical signal analysis system (16) is obtained with the adjacency matrix whose eigenvalues are $\lambda_k = e^{-j2\pi k/N}$, defined by (11). Note that any classical system whose transfer function can be described using the DFT with periodicity N can be written in this form with $M = N$.

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

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

$$H(z^{-1}) = \mathcal{Z}\{h_n\} = h_0 + h_1z^{-1} + \cdots + h_{M-1}z^{-(M-1)}. \tag{27}$$

Obviously

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

and we can use results defined for the classical z-domain transfer function.

Definition of the z-transform for arbitrary graph signals $x(n)$ and $y(n)$ that would satisfy the relation $Y(z^{-1}) = H(z^{-1})X(z^{-1})$ is not straightforward. It will be discussed later in Sect. 3.9.

3.4 Filtering in the Adjacency Matrix Spectral Domain

3.4.1 Normalization

The energy of a graph shifted signal is $\|\mathbf{x}_1\|_2^2 = \|\mathbf{A}\mathbf{x}\|_2^2$. The graph shift does not satisfy isometry property. In general, the energy of shifted signal is not the same as the energy of the original signal, $\|\mathbf{A}\mathbf{x}\|_2^2 \neq \|\mathbf{x}\|_2^2$. In processing graph signals it is commonly desirable that a graph shift does not increase the signal energy.

Using the matrix norm-two it can be easily shown that the ratio of energies of the graph shifted and the original signal satisfies the relation

$$\max\left\{\frac{\|\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right\} = \max\left\{\frac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\|\mathbf{x}\|_2^2}\right\} = \lambda_{\max}^2. \tag{28}$$

where $\lambda_{\max} = \max_k |\lambda_k|$.

If we do not want that the energy of a graph shifted signal $\|\mathbf{A}\mathbf{x}\|_2^2$ exceeds the energy of the original graph signal $\|\mathbf{x}\|_2^2$ then we should use the normalized adjacency matrix

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

in the graph shift operation and in any system on a graph. This normalization does not make the shift on graph operation isometric. The energy of the shifted signal is less than or equal to the energy of the original graph signal. The equality is achieved only for a very specific signal proportional to the eigenvector that corresponds to $\lambda_{\max}$.

The basic shift on a graph is then defined by using normalized adjacency matrix as

$$\mathbf{x}_1 = \mathbf{A}_{norm}\mathbf{x}. \tag{29}$$

A system on a graph with the normalized adjacency matrix is of the form

$$\mathbf{y} = \sum_{m=0}^{M-1} h_m \mathbf{A}_{norm}^m \mathbf{x}. \tag{30}$$

3.4.2 Spectral Domain Filtering

The filtering relation, as a special kind of a system on a graph, can be written as

$$\mathbf{y} = \sum_{m=0}^{M-1} h_m \mathbf{A}_{norm}^m \mathbf{x} = H(\mathbf{A}_{norm})\mathbf{x}.$$

Its spectral domain form follows from the decomposition of $H(\mathbf{A}_{norm})$ as

$$\mathbf{y} = H(\mathbf{A}_{norm})\mathbf{x} = \mathbf{U}H(\mathbf{\Lambda})\mathbf{U}^{-1}\mathbf{x}$$

with

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

as

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

where $\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}$ and $\mathbf{Y} = \mathbf{U}^{-1}\mathbf{y}$ are the graph Fourier transforms of the input and output signal, respectively. The transfer function of a filter on a graph is again

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

where λ_k are the eigenvalues of the normalized adjacency matrix $\mathbf{A}_{norm}$.

3.4.3 Spectral Ordering of the Adjacency Matrix Eigenvectors

For proper low-pass and high-pass filtering we have to establish the spectral order. This means that we have to establish a criterion to classify the eigenvectors, corresponding to the basis functions, as slow varying or fast varying. In classical Fourier analysis, the basis functions are ordered according to the frequency. Low-pass (slow varying) basis functions are the functions with small frequencies. The frequencies of the graph eigenvectors, as functions for signal decomposition, are not defined. We have to find another criterion to classify the eigenvectors. Again, an inspiration will be found in the classical Fourier analysis. In that case, instead of the frequency, energy of the signal change can be used as an indicator of the speed of a signal change in time.

The energy of a signal (a basis function) $u(n)$ change in classical analysis can be defined as the energy of the first difference

$$E_{\Delta u} = \sum_{n=0}^{N-1} |u(n) - u(n-1)|^2.$$

Lower values of $E_{\Delta u}$ means that $u(n)$ is slow-varying. Value $E_{\Delta u} = 0$ indicates, in classical signal analysis, that the signal is constant. Large values of $E_{\Delta u}$ are associated with fast signal changes in time. This form is also called the norm-two total variation of a signal. If the energy of a basis function $u(n)$ change is large it means that this eigenvector can be considered as the one belonging to the higher spectral content of the signal.

In graph signals the first graph difference can be defined as a difference of the graph signal and its graph shift. For an eigenvector $\mathbf{u}$, its form is

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

The energy of signal change is the energy of the first graph difference of signal $\mathbf{u}$

$$E_{\Delta u} = \|\mathbf{u} - \mathbf{A}_{norm}\mathbf{u}\|_2^2 \tag{31}$$

$$= \left\|\mathbf{u} - \frac{1}{\lambda_{\max}}\mathbf{A}\mathbf{u}\right\|_2^2 = \left\|\mathbf{u} - \frac{1}{\lambda_{\max}}\lambda\mathbf{u}\right\|_2^2 = |1 - \frac{\lambda}{\lambda_{\max}}|^2 \tag{32}$$

For eigenvectors $\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$ and $\|\mathbf{u}\|_2^2 = 1$ hold.

The energy of signal change is minimal for $\lambda = \lambda_{\max}$ and increases as λ decreases, Fig. 10.

After we have established a criterion for the eigenvector ordering, based on the corresponding eigenvalues, we will define an ideal low-pass filter. This filter should pass unchanged all signal components (eigenvectors) whose changes are slower than the one defined by the cut-off eigenvalue λ_c. It should stop all signal components (eigenvectors) whose variations are faster than the one defined by the cut-off eigenvalue. The ideal low-pass filter is defined as

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

As an example, consider a signal on a graph presented in Fig. 2a. The graph signal is obtained as a linear combination of two adjacency matrix eigenvectors $\mathbf{x} = 3.2\mathbf{u}_7 + 2\mathbf{u}_6$ (adjacency matrix eigenvectors of the considered graph are presented in Fig. 10). The signal is presented in Fig. 24a. The signal is corrupted by a white Gaussian noise with signal-to-noise (SNR) ratio $SNR_{in} = 2.7\,\text{dB}$. The noisy graph signal is presented in Fig. 24b. The noisy signal is filtered by using an ideal spectral domain graph filter with a cut-off eigenvalue $\lambda_c = 1$. The output signal, presented in Fig. 24c, is obtained. The output SNR is $SNR_{out} = 18.8\,\text{dB}$.

The energy of signal change criterion is consistent with the classical DFT based filtering when $\lambda_k = \exp(-j2\pi k/N)$ and $\lambda_{\max} = 1$. In that case the decision on the low-pass and high-pass basis functions is made based on $E_{\Delta u}$ value and a given threshold.

3.4.4 Spectral Domain Filter Design

Let the desired graph transfer function be $G(\mathbf{\Lambda})$. A system with this transfer function can be implemented either in the spectral domain or in the vertex domain.

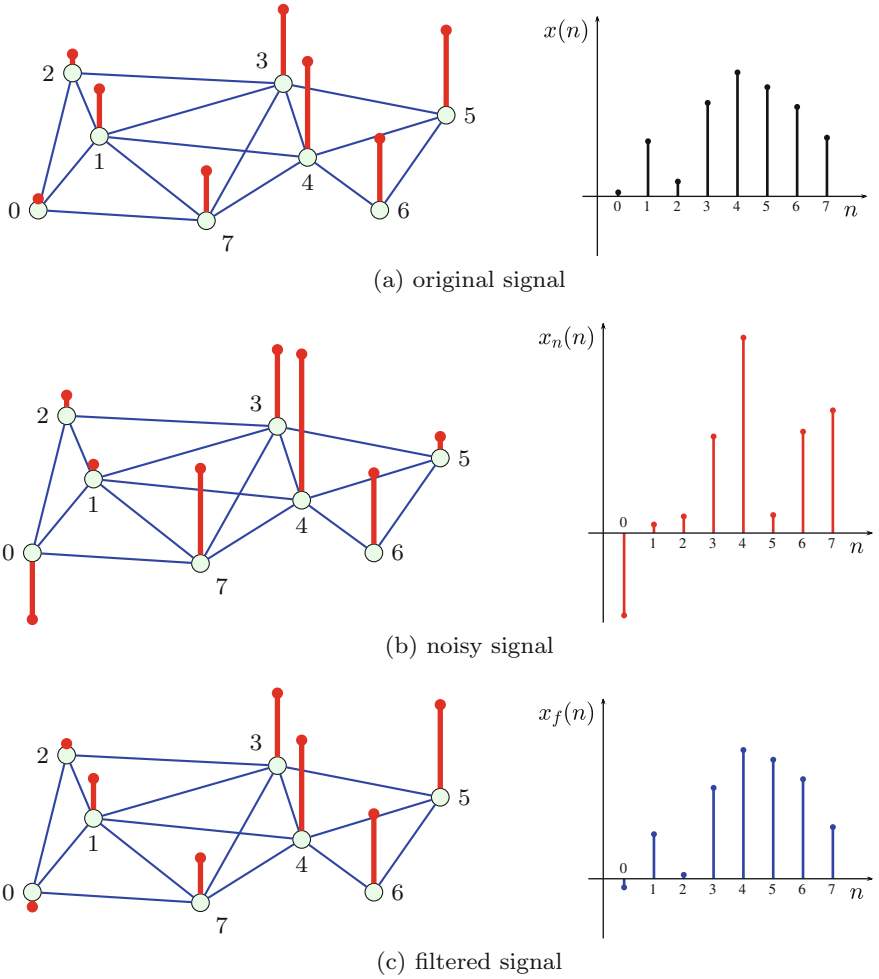


Fig. 24 Signal filtering example. Original signal (**a**), noisy signal (**b**) and filtered signal (**c**). An ideal low-pass filtering with two highest eigenvalues in the pass-band is applied

In the spectral domain the implementation is straightforward. It can be performed in the following three steps:

1. Calculate the GDFT of the input graph signal $\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}$,
2. Multiply the GDFT of the input graph signal by $G(\mathbf{\Lambda})$ to get $\mathbf{Y} = G(\mathbf{\Lambda})\mathbf{X}$, and
3. Calculate the output graph signal as the inverse GDFT, $\mathbf{y} = \mathbf{U}\mathbf{Y}$.

This procedure may computationally be very demanding for large graphs. In the case of a large graph it would be easier to implement the desired filter (or its close approximation) in the vertex domain.

For the implementation in the vertex domain, we have to find the coefficients $h_0, h_1, \ldots, h_{M-1}$ in (15) such that its spectral representation $H(\mathbf{\Lambda})$ is equal (or approximately equal) to $G(\mathbf{\Lambda})$. This is done in the following way. The transfer function of the vertex domain system is given by (26) as $H(\lambda_k) = h_0 + h_1\lambda_k^1 + \cdots h_{M-1}\lambda_k^{M-1}$. It should be equal to the desired transfer function $G(\lambda_k)$, for $k = 0, 1, \ldots, N-1$. This condition leads to a system of linear equations

$$\begin{aligned} h_0 + h_1\lambda_0^1 + \cdots h_{M-1}\lambda_0^{M-1} &= G(\lambda_0) \\ h_0 + h_1\lambda_1^1 + \cdots h_{M-1}\lambda_1^{M-1} &= G(\lambda_1) \\ &\vdots \\ h_0 + h_1\lambda_{N-1}^1 + \cdots h_{M-1}\lambda_{N-1}^{M-1} &= G(\lambda_{N-1}). \end{aligned} \tag{33}$$

The matrix form of this system is

$$\mathbf{V}_\lambda \mathbf{h} = \mathbf{g}, \tag{34}$$

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

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

is the vector of the system coefficients that we want to calculate, and

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

Now we will comment on the solution of system (33), (34).

Comments on the System of Equations Solution

1. Consider the case when all eigenvalues are distinct (minimal polynomial is equal to characteristic polynomial, $P_{min}(\lambda) = P(\lambda)$).
   (a) If the filter order is such that $M = N$, then the solution of (33) is unique, since the Vandermonde determinant is always nonzero.
   (b) If the filter order is such that $M < N$, then system (33) is overdetermined. The solution of (33) is obtained in the mean squared sense only (as it will be described later in this section).
2. If some of the eigenvalues are of a degree higher than one (minimal polynomial order N_m is lower than N) system (33) reduces to a system of N_m linear equations (by removing multiple equations for the repeated eigenvalues λ).
   (a) If the filter order is such that $N_m < M \leq N$ the system is underdetermined. In that case $M - N_m$ filter coefficients are free variables. The system has an infinite number of solutions. All obtained filters are equivalent.
   (b) If the filter order is such that $M = N_m$ the solution of system (33) is unique.
   (c) If the filter order is such that $M < N_m$ the system (33) is overdetermined and the solution is obtained in the mean squared sense.

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

Solution of the System

For $M = N = N_m$ the solution of system (33) or (34) is

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

For the overdetermined case (when $M < N_m$) the mean-square approximation of $\mathbf{h} = [h_0, h_1, \ldots, h_M]^T$ is obtained by minimizing the squared error

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

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

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

In the cases when $M < N_m$ the obtained solution $\hat{\mathbf{h}}$ is the mean square solution for $\mathbf{V}_\lambda\mathbf{h} = \mathbf{g}$. Since this solution may not satisfy $\mathbf{V}_\lambda\mathbf{h} = \mathbf{g}$ then the designed coefficients $\hat{\mathbf{g}}$ (its spectrum $\hat{G}(\mathbf{\Lambda})$)

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

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

As an example consider the graph from Fig. 2a and the synthesis of a desired filter whose frequency response would be

$$\mathbf{g} = [1, 1, 0.5, 0.4, 0.1, 0, 0, 0]^T.$$

Consider the following cases: $M = 0, 1, 2, 3$. The filter is designed for various M using (33). The solution is obtained according the presented procedure. The results are shown in Fig. 25. The vertex domain realization of the filter with $M = 3$ is

$$\mathbf{y} = 0.4456\mathbf{A}^0\,\mathbf{x} + 0.2298\mathbf{A}^1\,\mathbf{x} - 0.0188\mathbf{A}^2\,\mathbf{x}. \tag{35}$$

For $M = N = 8$ the exact frequency response $\hat{\mathbf{g}} = \mathbf{g}$ is obtained.

Inverse System

An inverse filter $H(\mathbf{\Lambda})$ to $G(\mathbf{\Lambda})$ is obtained from

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

It means that $H(\lambda_k) = 1/G(\lambda_k)$ for each k if all $G(\lambda_k) \neq 0$ and $P(\lambda) = P_{min}(\lambda)$.

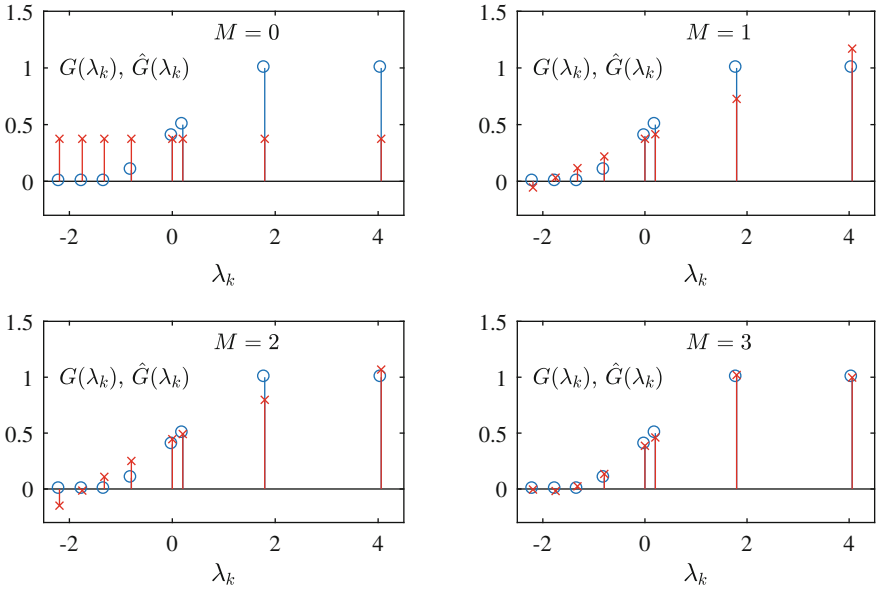


Fig. 25 Designing a filter with the specified transfer function in the spectral domain. The desired spectral response $G(\lambda_k)$ is presented with blue circles. The designed graph system response $\hat{G}(\lambda_k)$, obtained with $M+1$ filter coefficients $h_0, h_1, \ldots, h_M$ in the vertex domain, is presented with red asterisks

3.5 Graph Fourier Transform Based on the Laplacian

Like in the case of an adjacency matrix, the spectral decomposition of a graph signal can be done using the eigenvalue decomposition of the Laplacian $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ or $\mathbf{L}\mathbf{U} = \mathbf{U}\mathbf{\Lambda}$. Although the analysis can be done in a unified way for both the adjacency matrix and the Laplacian based spectral decomposition, due to their different behavior and importance they will be considered separately.

The graph Fourier transform of a signal $\mathbf{x}$, using the Laplacian eigenvalue decomposition, is defined as

$$\mathbf{X} = \mathbf{U}^{-1}\mathbf{x}, \tag{36}$$

where $\mathbf{U}$ is a matrix with the Laplacian eigenvectors. The inverse graph Fourier transform is

$$\mathbf{x} = \mathbf{U}\,\mathbf{X}. \tag{37}$$

In the case of circular unweighted graph this spectral analysis also reduces to the standard Fourier transform, but with real-valued basis functions (13).

3.6 Ordering and Filtering in the Laplacian Spectral Domain

The graph shift and adjacency matrix are related to the first finite difference in the vertex domain. The eigenvectors (basis functions) variations are related to the energy of the graph signal change, Sect. 3.4.3. A similar approach can be used for the Laplacian based decomposition.

In the case of classical time domain signals, the Laplacian on a circle graph represents the second order finite difference $y(n) = -u(n-1) + 2u(n) - u(n+1)$. This difference can be written in a matrix form as $\mathbf{y} = \mathbf{L}\mathbf{u}$. It is obvious that the eigenvectors $u(n)$ with small changes should have small cumulative energy of the second order difference $E_u = \sum_n ((u(n) - u(n-1))^2 + (u(n) - u(n+1))^2)/2$. This value corresponds to the quadratic form of eigenvector $\mathbf{u}$ defined by $E_u = \mathbf{u}^T\mathbf{L}\mathbf{u}$. This reasoning can be used in the graph signals as well. As a default case for the Laplacian analysis we will use weighted undirected graphs.

By definition

$$\mathbf{L}\mathbf{u} = \lambda\mathbf{u}$$

or

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

since $\mathbf{u}^T\mathbf{u} = 1$ by definition. It means that the quadratic form of an eigenvector is equal to the corresponding eigenvalue. Next we will show that it can be used as a measure of the signal smoothness. By definition

$$\mathbf{u}^T\mathbf{L}\mathbf{u} = \sum_{n=0}^{N-1} u(n) \sum_{m=0}^{N-1} W_{nm}(u(n) - u(m)) = \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} W_{nm}(u^2(n) - u(n)u(m)).$$

In full summations over n and m we can replace the summation of $u^2(n)$ by a half of the summations of both $u^2(n)$ and $u^2(m)$ over n and m, since $W_{nm} = W_{mn}$. The same can be done for $u(n)u(m)$. Then we can write

$$\mathbf{u}^T\mathbf{L}\mathbf{u} = \frac{1}{2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} W_{nm}(u^2(n) - u(n)u(m) + u^2(m) - u(m)u(n)) = \frac{1}{2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} W_{nm}(u(n) - u(m))^2 \geq 0. \tag{38}$$

Obviously small $\mathbf{u}^T\mathbf{L}\mathbf{u} = \lambda$ means small variations $W_{nm}(u(n) - u(m))^2$ in the eigenvector for each vertex n. The eigenvectors corresponding to small λ belong to the low-pass part of a graph signal.

From the previous analysis we may also conclude that the eigenvalues of a Laplacian are nonnegative (positive semi-definite matrix). At least one Laplacian eigenvalue is equal to 0. According to the Laplacian definition we have that the sum of

each row (column) is equal to 0. It means that for $\mathbf{x} = \mathbf{1}$ we have $\mathbf{Lx} = \mathbf{0} = 0 \cdot \mathbf{x}$. We can conclude that there is eigenvalue $\lambda_0 = 0$ with corresponding eigenvector $\mathbf{u}_0 = \mathbf{1}/\sqrt{N}$. A vector with all values equal to 1 is denoted by $\mathbf{1}$.

In general, the smoothness of a graph signal $\mathbf{x}$ is defined by the quadratic form

$$E_x = \mathbf{x}^{\mathbf{T}}\mathbf{Lx}.$$

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

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

As an example consider a signal on the graph presented in Fig. 3. The graph signal is obtained as a linear combination of two Laplacian eigenvectors $\mathbf{x} = 2\mathbf{u}_0 + 1.5\mathbf{u}_1$. (Laplacian eigenvectors of the considered graph are presented in Fig. 12.) This signal is presented in Fig. 26a. The signal is corrupted by a white Gaussian noise. The noisy graph signal is described by a signal-to-noise (SNR) ratio $SNR_{in} = -1.76$ dB. Noisy graph signal is presented in Fig. 26b. Using an ideal spectral domain graph filter, with a cut-off eigenvalue $\lambda_c = 2$, the noisy graph signal is filtered. The output signal, presented in Fig. 26c, is obtained. The output SNR, for this signal, is $SNR_{out} = 21.29$ dB.

A direct relation between the adjacency and Laplacian spectral decomposition can be established for K-regular unweighted graphs. For these graphs holds

$$\mathbf{L} = K\mathbf{I} - \mathbf{A}$$

resulting in

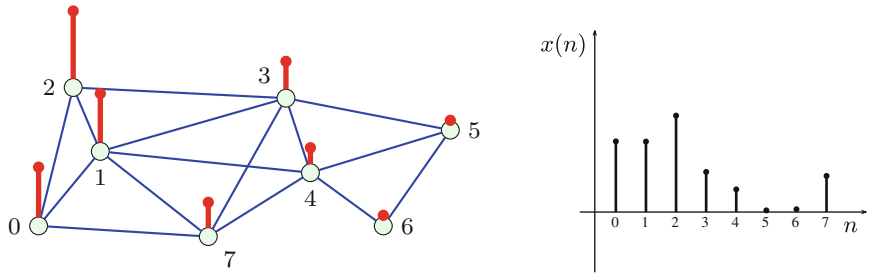
$$\lambda_A = K - \lambda_L,$$

where the adjacency matrix and the Laplacian eigenvalues are denoted by λ_A and λ_L, respectively. The eigenvectors are the same. Ordering with respect to λ from the low-pass to the high-pass part is just opposite for these two graph spectral decompositions.

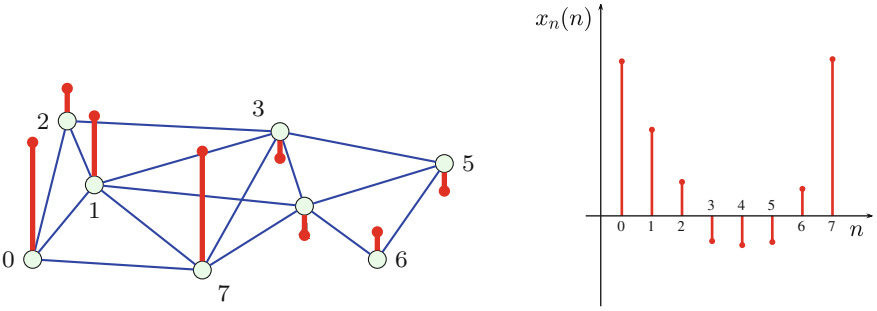
3.7 Systems on a Graph Defined Using the Laplacian

A system on a graph can be defined using the Laplacian as well

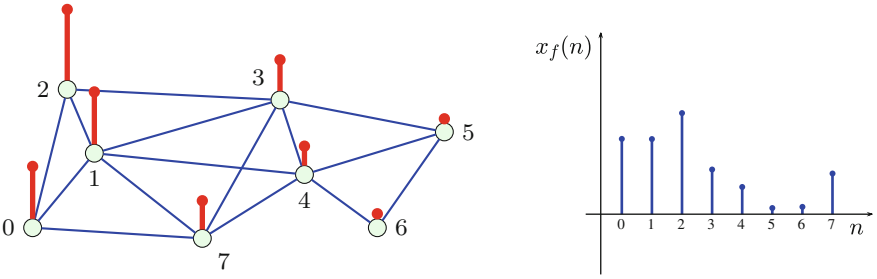
$$\mathbf{y} = h_0\mathbf{L}^0\mathbf{x} + h_1\mathbf{L}^1\mathbf{x} + \cdots + h_{M-1}\mathbf{L}^{M-1}\mathbf{x} = \sum_{m=0}^{M-1} h_m\mathbf{L}^m\mathbf{x}. \tag{39}$$



(a) original signal



(b) noisy signal



(c) filtered signal

Fig. 26 Signal filtering example. Original signal (**a**), noisy signal (**b**) and filtered signal (**c**). Low pass filtering with two smallest eigenvalues is applied

For an unweighted graph this system form can be related to the adjacency matrix form using $\mathbf{L} = \mathbf{D} - \mathbf{A}$.

The spectral domain description of a graph system is obtained using the Laplacian eigenvalue decomposition,

$$\mathbf{y} = \sum_{m=0}^{M-1} h_m \mathbf{L}^m \mathbf{x} = H(\mathbf{L})\mathbf{x} = \mathbf{U}H(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x} = \mathbf{U}H(\mathbf{\Lambda})\mathbf{X} = \mathbf{U}\mathbf{Y}, \tag{40}$$

where

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

or

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

In the vertex domain the nth element of $\mathbf{y} = \mathbf{U}H(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x}$ is

$$y(n) = \sum_{k=0}^{N-1}\sum_{i=0}^{N-1} x(i)u_k(i)H(\lambda_k)u_k(n) = \sum_{i=0}^{N-1} x(i)h_n(i), \tag{41}$$

where

$$H(\lambda_k) = h_0 + h_1\lambda_k + \cdots + h_{M-1}\lambda_k^{M-1} \tag{42}$$

and

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

The value of $y(n)$ can be interpreted as a generalized convolution, using a generalized shift of impulse response in the vertex domain. It can be described by using responses to the unite delta pulses. Let us consider the delta function located at a graph vertex m and its spectrum. The delta function at the vertex m is defined as

$$\delta_m(n) = \begin{cases} 1 & \text{for } n = m \\ 0 & \text{for } n \neq m, \end{cases} \tag{43}$$

and the corresponding spectrum is given by

$$\Delta(\lambda_k) = \sum_{n=0}^{N-1} \delta_m(n)u_k(n) = u_k(m). \tag{44}$$

Any graph signal can be written as

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

or in a vector form

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

where $\boldsymbol{\delta}_i$ is a vector with elements $\delta(n-i)$. Then

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

with elements

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

Calculation of this form of convolution for a vertex n, given by (41), is localized to the $(M-1)$ neighborhood of vertex n, according to (40). This is an important property for large graphs. A generalized convolution for two arbitrary graph signals will be explained next.

3.8 Convolution of Signals on a Graph

Consider two graph signals $x(n)$ and $h(n)$. A generalized convolution operator of these two signals on a graph is defined using their spectra [39]. The assumption is that the spectrum of a convolution

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

on a graph is equal to the product of the graph signal spectra

$$Y(k) = X(k) H(k). \tag{45}$$

The result of the generalized graph convolution operation $x(n) * h(n)$ is equal to the inverse GDFT of $Y(k)$,

$$y(n) = x(n) * h(n) = \sum_{k=0}^{N-1} Y(k) u_k(n) = \sum_{k=0}^{N-1} X(k) H(k) u_k(n).$$

In this case

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

A shift on the graph can be defined within the framework of the generalized convolution. Consider the graph signal $h(n)$ and the delta function located at the vertex m. Here, we will use $h_m(n)$ to denote the shifted version of the graph signal $h(n)$. The signal corresponding to a shift to a vertex m is equal to

$$h_m(n) = h(n) * \delta_m(n) = \sum_{k=0}^{N-1} H(k) u_k(m) u_k(n). \tag{46}$$

The same relation follows from the inverse GDFT of $X(k)H(\lambda_k)$,

$$y(n) = \sum_{k=0}^{N-1} X(k)H(k)u_k(n) = \sum_{k=0}^{N-1}\sum_{m=0}^{N-1} x(m)u_k(m)H(k)u_k(n) = \sum_{m=0}^{N-1} x(m)h_m(n) = x(n) * h(n), \tag{47}$$

where

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

plays the role of a shifted signal. Since the definition of $H(k)$ as a GDFT of a signal $h(n)$ differs from (42) it produces different shift operation. These two shift operations are denoted by $T_m\{h(n)\}$ and $\mathcal{T}_m\{h(n)\}$, respectively.

Consider, for example, the signal with Laplacian GDFT

$$H(k) = \exp(-k/4).$$

Shifted signals $h(n)$ obtained using $h_m(n) = T_m\{h(n)\}$ are presented in Fig. 27.

3.9 Graph z-Transform of a Signal

The relation between $T_m\{h(n)\}$ and $\mathcal{T}_m\{h(n)\}$ can be established based on the definitions of $H(\lambda_k)$ and $H(k)$. For $H(\lambda_k)$ defined by (42) the corresponding IGDFT coefficients $h(n)$

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

and the coefficients h_n in

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

are not the same, $h(n) \neq h_n$.

Vector form of the last two relations is

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

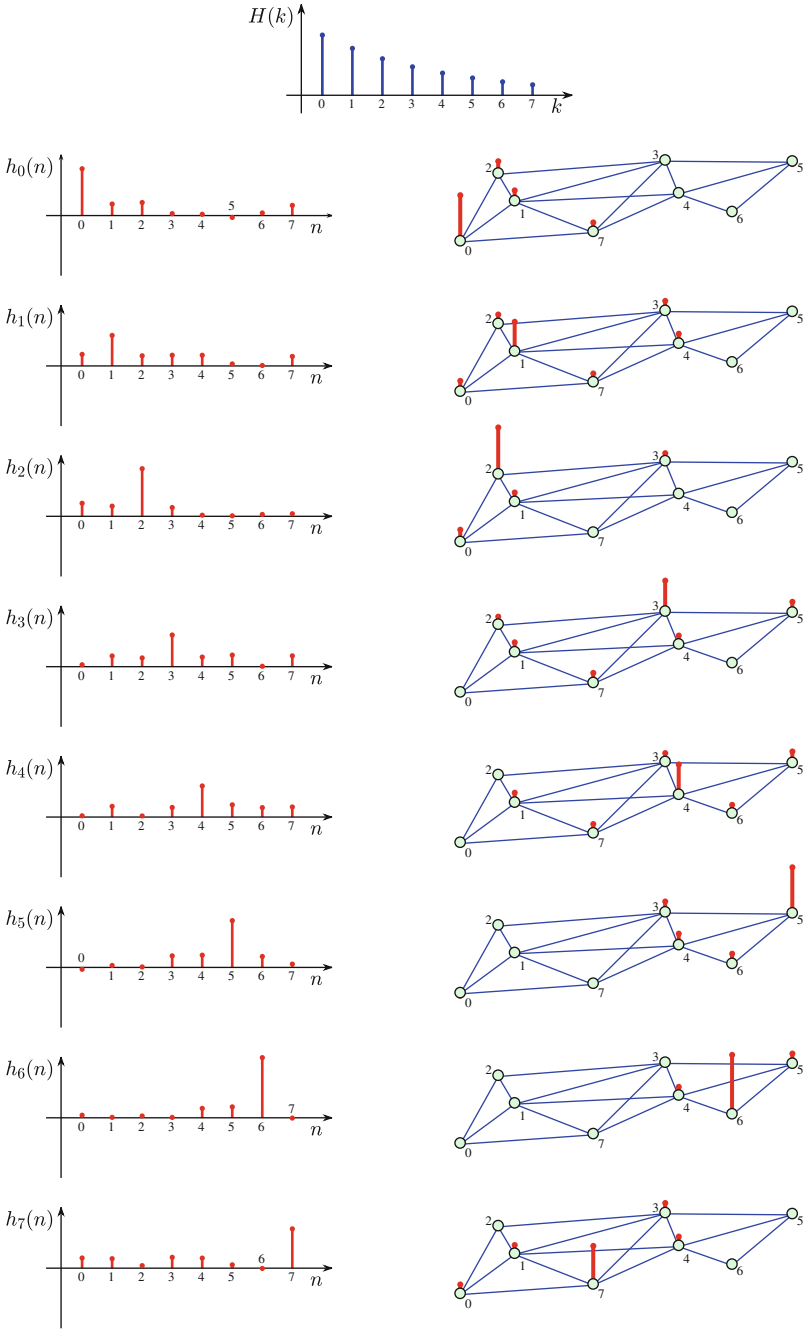


Fig. 27 Graph signal shifts based on the Laplacian eigendecomposition

The signal $h(n)$ and the coefficients h_n can easily be related via

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

These coefficients would be the same in the classical DFT (with directed adjacency matrix) when $\lambda_k = \exp(-j2\pi k/N)$ and $u_k(n) = \exp(j2\pi nk/N)/\sqrt{N} = \lambda_k^{-n}/\sqrt{N}$, with $h_n = h(n)$ and $H(\lambda_k) = \sum_{n=0}^{N-1} h(n)u_k^*(n)$.

The previous relation will be used to define the z-transform of a graph signal. For given signal $\mathbf{x} = [x(0)\ x(1)\dots x(N-1)]^T$ the signal corresponding to a system transfer function that would have the same GDFT is

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

The z-transform of these coefficients is

$$X(z^{-1}) = \mathcal{Z}\{x_n\} = x_0 + x_1 z^{-1} + \dots + x_{N-1}z^{-(N-1)}. \tag{48}$$

For this z-transform holds

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

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

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

where the output graph signal $y(n)$ is obtained from the inverse z-transform of the coefficients y_n of $Y(z^{-1}) = H(z^{-1})X(z^{-1})$

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

The z-transform representation may be of interest when the eigenvalues are complex-valued. They may appear in decomposition of adjacency matrices of undirected graphs. For example, for the graph from Fig. 2b and its adjacency matrix the eigenvalues are presented in Fig. 28.

The analytic signal and Hilbert transform are defined as

$$X_a(k) = (1 + \text{sign}(\text{Imag}(\lambda_k))X(k)$$

$$X_h(k) = j\ \text{sign}(\text{Imag}(\lambda_k))X(k)$$

$$X(k) = X_a(k) + jX_h(k)$$

If these relations are applied to the standard DFT with $\lambda_k = \exp(-j2\pi k/N)$ we would get the classical signal processing definitions.

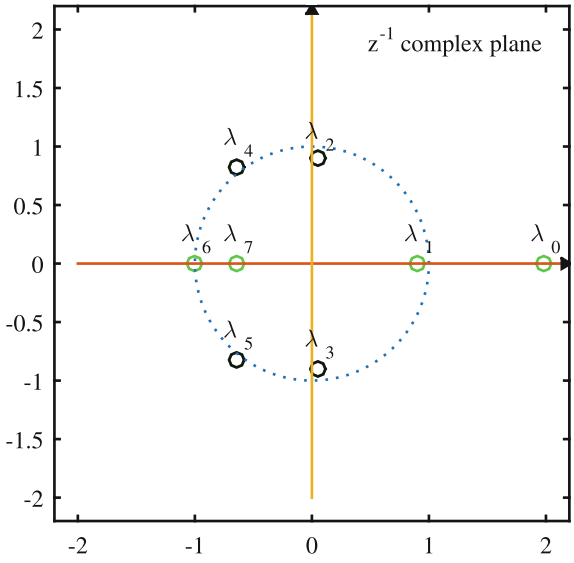


Fig. 28 Eigenvalues of the directed graph adjacency matrix

3.10 Shift in the Spectral Domain

We can define a shift in the spectral domain in the same way as the shift in the vertex domain has been defined. Consider a product of two signals $x(n)y(n)$ on an undirected graph. Its GDFT is

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

where

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

can be considered as a shift of $Y(k)$ for i. Obviously $Y_0(k) = Y(k)$ up to a constant factor. This relation does not hold for the shift in the vertex domain.

3.11 Parseval's Theorem on a Graph

For two signals $x(n)$ and $y(n)$ on an undirected graph and their spectra $X(k)$ and $Y(k)$, Parseval's theorem holds

$$\sum_{n=0}^{N-1} x(n)y(n) = \sum_{k=0}^{N-1} X(k)Y(k). \tag{49}$$

To prove Parseval's theorem on graphs we can write

$$\sum_{n=0}^{N-1} x(n)y(n) = \sum_{n=0}^{N-1} \left[\sum_{k=0}^{N-1} X(k)u_k(n) \right] y(n) = \sum_{k=0}^{N-1} X(k) \sum_{n=0}^{N-1} y(n)u_k(n), \tag{50}$$

producing Parseval's theorem. It has been assumed that $\mathbf{U}^{-1} = \mathbf{U}^T$ for undirected graphs. This theorem holds for both the Laplacian and the adjacency matrix based decompositions on undirected graphs.

3.12 Optimal Denoising

Consider a signal composed of a slow-varying signal $\mathbf{s}$ and a fast changing disturbance ε

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

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

The optimal denoising task could be defined as a minimization of

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

Before we solve this problem we will explain the meaning of terms in the cost function J. Minimization of the first term $\frac{1}{2}\|\mathbf{y} - \mathbf{x}\|_2^2$ forces that the output signal $\mathbf{y}$ is as close to the available observations $\mathbf{x}$ as possible. The second term is a measure of signal $\mathbf{y}$ smoothness. It promotes the solution smoothness (see Sect. 3.6). Parameter α is a balance between output closeness to $\mathbf{x}$ and smoothness of $\mathbf{y}$ criterion.

The solution of the minimization problem is

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

resulting in

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

The Laplacian spectral domain form of this relation follows with $\mathbf{L} = \mathbf{U}^T \mathbf{\Lambda} \mathbf{U}$, $\mathbf{Y} = \mathbf{U}^T \mathbf{y}$, and $\mathbf{X} = \mathbf{U}^T \mathbf{x}$ as

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

The transfer function of this filter is

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

For a small α, $H(\lambda_k) \approx 1$ and $\mathbf{y} \approx \mathbf{x}$. For a large α, $H(\lambda_k) \approx \delta(k)$ and $\mathbf{y} \approx const.$ is maximally smooth (a constant, without any variation).

Here we will state two more cost function forms used for graph signal denoising.

Instead of the resulting signal smoothness, we may add the condition that its deviation from a linear form is as small as possible. Then the cost function is

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

resulting in a closed form solution

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

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

A combination of the previous two cost function forms may provide additional flexibility in the transfer function design. If we use

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

we would get the transfer function

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

We can change the transfer function form by choosing appropriate values of the parameters α and β. For example, if we want the component corresponding to $\lambda_1 \neq 0$ to be unattenuated we would use $\alpha + \beta\lambda_1 = 0$. This cost function can be extended to produce a transfer function for M unattenuated components.

In some applications we would like to promote the sparsity of the resulting signal change, instead of its smoothness. Then the compressive sensing theory requires that the quadratic form or the norm-two in the previous equations is replaced with the forms that promote sparsity. Two possible such cost functions are:

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

and

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

with $0 \leq p \leq 1$. Minimization of these functions can not be done in an analytic way, like in the case of $p = 2$. The norm-zero, with $p = 0$, is the best in promoting sparsity. For $p = 0$, the second term in minimization counts and minimizes the number of nonzero elements in $\mathbf{Ly}$. In the second minimization form the norm-zero promotes the smallest possible number of nonzero elements of the form $\sum_{m=0}^{N-1} W_{nm}(y(n) - y(m))^2$. This is the total variations (TV) approach.

The norm-one with $p = 1$ in previous relations is convex, allowing the gradient descend methods in the solution, while producing the same solution as with $p = 0$, under some conditions.

4 Subsampling, Compressed Sensing, and Reconstruction

Graphs may have a large number of vertices. This number can be of the order of millions or even higher. The fact that the number of vertices and corresponding graph signal values can be extremely large makes the problem of subsampling and compressive sensing crucially important in graph signal processing. The problems of subsampling and compressive sensing are closely related to the reconstruction possibility from a reduced set of measurements (signal samples or their linear combinations). Here we will present several basic approaches to the subsampling, along with their relations to classical signal processing [40–58].

4.1 Subsampling of the Low-Pass Graph Signals

We will start with the simplest case where we can assume that the considered graph signal is of low-pass type. This signal can be written as a linear combination of $K < N$ eigenvectors with the slowest changes. For example, for the Laplacian spectrum of a signal with K nonzero values

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

the signal is of form

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

The smallest number of graph signal samples needed to recover this signal is $M = K < N$. Assume that M signal samples are available, $K \leq M < N$. The vector of available graph signal samples will be referred to as the measurement vector. It will be denoted by $\mathbf{y}$. The set of vertices where the graph signal samples are available is

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

The measurement matrix can be defined using the IGDFT $\mathbf{x} = \mathbf{U}\,\mathbf{X}$ or

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

Keeping the equations corresponding to the available graph signal samples at $n \in \mathcal{M} = \{n_1, n_2, \ldots, n_M\}$ we get

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

The matrix form of this system of equations is

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

where $\mathbf{A}_{MN}$ is the measurement matrix and $\mathbf{y} = [x(n_1), x(n_2), \ldots, x(n_M)]^T$ are the available samples. This system is underdetermined for $M < N$. It cannot be solved uniquely for $\mathbf{X}$ without additional constraints. Since we have assumed that the signal contains a linear combination of only $K \leq M$ the slowest varying eigenvectors, we can exclude the GDFT coefficients $X(K), X(K+1), \ldots, X(N-1)$, since they are zero-valued and do not contribute to the graph signal samples. Then we can write

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

This system in a matrix form reads

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

where the definition of the reduced measurement matrix $\mathbf{A}_{MK}$ and the reduced GDFT vector $\mathbf{X}_K$ is obvious. For $K = M$ this system can be solved. If $M > K$ the system is overdetermined and the solution is found in the mean squared error (MSE) sense. This solution is

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

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

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

The recovery condition is that the inverse $(\mathbf{A}_{MK}^T \mathbf{A}_{MK})^{-1}$ exists. It means that

$$\text{rank}(\mathbf{A}_{MK}^T \mathbf{A}_{MK}) = K. \tag{51}$$

In terms of the matrix condition number, the requirement is

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

In the case of noisy measurements, the noise in the reconstructed GDFT coefficients is directly related to the input noise and the matrix condition number. If we are able to choose the signal sample positions (vertices), then the sampling strategy should be to find the set of measurements producing the condition number as close to one as possible.

As an example of the reconstruction from a reduced set of signal samples, consider a graph's signal values at $M = 3$ vertices

$$\mathbf{y} = [x(0) \;\; x(2) \;\; x(6)]^T = [0.299 \;\; 0.345 \;\; 1.361]^T.$$

Assume that the graph signal is of low-pass type with $K = 2$ lowest nonzero GDFT coefficients $X(0)$ and $X(1)$. The signal GDFT coefficients can be reconstructed from

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

since the rank of matrix $\mathbf{A}_{MK}$ is 2. The matrix condition number value is $\text{cond}(\mathbf{A}_{MK}^T \mathbf{A}_{MK}) = 1.97$. The reconstructed nonzero values of the GDFT are $X(0) = 2$ and $X(1) = 1$. The reconstructed graph signal $\mathbf{x} = \mathbf{U}\mathbf{X}$ is presented in Fig. 29.

The classical signal processing downsampling and interpolation relations are obtained with $u_k(n) = \exp(j2\pi nk/N)/\sqrt{N}$.

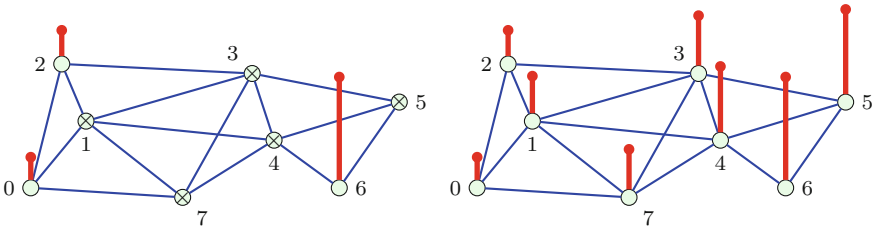


Fig. 29 Subsampling of a lowpass graph signal example

4.2 Subsampling of the Sparse Graph Signals

4.2.1 Known Coefficient Positions

The previous analysis holds not only for a low-pass type of $\mathbf{X}$, but for a general $\mathbf{X}$ with K nonzero values at arbitrary, known, spectral positions,

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

as well. Then the system of equations

$$\begin{bmatrix} x(n_1) \\ x(n_2) \\ \vdots \\ x(n_M) \end{bmatrix} = \begin{bmatrix} u_{k_1}(n_1) & u_{k_2}(n_1) & \ldots & u_{k_K}(n_1) \\ u_{k_1}(n_2) & u_{k_2}(n_2) & \ldots & u_{k_K}(n_2) \\ \vdots & & & \\ u_{k_1}(n_M) & u_{k_2}(n_M) & \ldots & u_{k_K}(n_M) \end{bmatrix} \begin{bmatrix} X(k_1) \\ X(k_2) \\ \vdots \\ X(k_K) \end{bmatrix}, \tag{52}$$

whose matrix form reads $\mathbf{y} = \mathbf{A}_{MK}\mathbf{X}_K$, is solved for the nonzero spectral values $X(k), k \in \mathcal{K}$, in the same way as in the case of low-pass signal presented in Sect. 4.1.

4.2.2 Support Matrices, Subsampling-Upsampling

In graph signal processing literature the subsampling problem is often defined using the support matrices. Assume that the signal is subsampled in such way that it is available on a subset of vertices $n \in \mathcal{M} = \{n_1, n_2, \ldots, n_M\}$ instead on the full set of vertices. For the subsampled signal we can define its upsampled version, obtained by adding zeros at vertices where the signal is not available. The subsampled and upsampled version of the original signal $\mathbf{x}$ is

$$\mathbf{x}_s = \mathbf{B}\mathbf{x}.$$

The support matrix $\mathbf{B}$ is an $N \times N$ diagonal matrix with ones at the diagonal positions corresponding to $\mathcal{M} = \{n_1, n_2, \ldots, n_M\}$ and zeros elsewhere. Signal $\mathbf{x}$ with N independent values cannot be reconstructed from its $M < N$ values in $\mathbf{x}_s$, without additional constraints. For sparse signals the additional constraint is that the signal $\mathbf{x}$ has only $K \leq M$ nonzero coefficients in the GDFT domain $\mathbf{X} = \mathbf{U}^T\mathbf{x}$ at $k \in \mathcal{K} = \{k_1, k_2, \ldots, k_K\}$. Then

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

holds. The support matrix $\mathbf{C}$ is an $N \times N$ diagonal matrix with ones at the diagonal positions corresponding to $\mathcal{K} = \{k_1, k_2, \ldots, k_K\}$ and zeros elsewhere. Note that $\mathbf{X}$ is on both sides of this equation, contrary to $\mathbf{x}_s = \mathbf{B}\mathbf{x}$. The reconstruction formula follows from

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

with $\mathbf{X} = \text{pinv}\left(\mathbf{B}\mathbf{U}^T\mathbf{C}\right)\mathbf{x}_s$. The inversion

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

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

$$\text{rank}(\mathbf{C}) = K = \text{rank}\left(\mathbf{B}\mathbf{U}^T\mathbf{C}\right).$$

This condition is equivalent to (51) since the nonzero part of matrix $\mathbf{B}\mathbf{U}^T\mathbf{C}$ is equal to $\mathbf{A}_{MK}$ in (52).

4.2.3 Unknown Coefficient Positions

The problem is more complex if the positions of nonzero spectral coefficients $\mathcal{K} = \{k_1, k_2, \ldots, k_K\}$ are not known. This problem has been formulated and solved within compressive sensing theory. The reconstruction problem formulation is

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

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

The minimization problem can be solved in many ways. Here we will present a simple, two step solution:

1. Using $M \gg K$ signal samples, the positions $\mathcal{K}$ of nonzero coefficients are estimated.
2. The nonzero coefficients of $\mathbf{X}$ at the estimated positions $\mathcal{K}$ are reconstructed, along with the signal $\mathbf{x}$ at all vertices.

For the estimation of nonzero positions in Step 1 we can use the projection of measurements to the measurement matrix

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

defined as

$$\mathbf{X}_0 = \mathbf{A}_{MN}^T\mathbf{y}. \tag{53}$$

Positions of K largest values in $\mathbf{X}_0$ are used as an estimate of the nonzero positions $\mathcal{K}$. This procedure can also be implemented in an iterative way. In the first iteration we assume $K = 1$ and the largest component is estimated and removed. The position of the second component is then estimated. The first and the second component are then reconstructed and removed. The procedure is iteratively repeated K times.

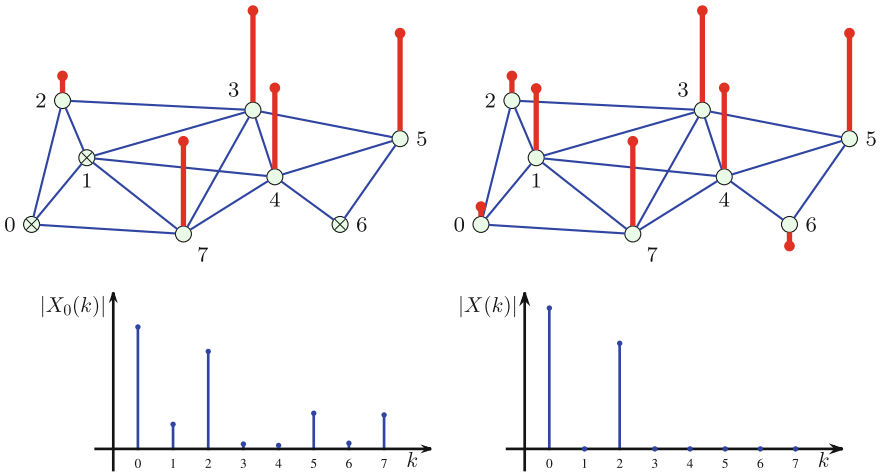


Fig. 30 Compressive sensing on graphs

As an example consider a sparse graph signal with sparsity $K = 2$ measured at vertices 2, 3, 4, 5, and 7

$$\mathbf{y} = [0.224 \;\; 1.206 \;\; 1.067 \;\; 1.285 \;\; 1.116]^T.$$

Measurements (available signal samples) are illustrated in Fig. 30 (upper left subplot). The estimate $\mathbf{X}_0$ is calculated according to (53). The positions of two non-zero coefficients are estimated as positions of two largest values in $\mathbf{X}_0$. In the considered case $\mathcal{K} = \{0, 2\}$, Fig. 30 (lower left subplot). The GDFT coefficients are then reconstructed for sparsity $K = 2$, resulting in $X(0) = 2$, $X(2) = 1.5$, Fig. 30 (lower right). The reconstructed graph signal at all vertices is presented in Fig. 30 (upper right subplot).

4.2.4 On the Unique Reconstruction Conditions

It is easy to show that the initial estimate $\mathbf{X}_0$ will produce correct positions of the nonzero elements $X(k)$ and the reconstruction will be unique if

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

where μ is equal to the maximal value of the inner product of two columns of the measurement matrix $\mathbf{A}_{MN}$ (the coherence index).

A K-sparse signal can be written as $x(n) = \sum_{i=1}^{K} X(k_i)u_{k_i}(n)$. Its initial estimate values are

$$X_0(k) = \sum_{i=1}^{K} X(k_i) \sum_{n \in \mathcal{M}} u_k(n) u_{k_i}(n) = \sum_{i=1}^{K} X(k_i) \mu(k, k_i)$$

where $\mathcal{M} = \{n_1, n_2, \ldots, n_M\}$ and $\mu(k, k_i) = \sum_{n \in \mathcal{M}} u_k(n) u_{k_i}(n)$. If the maximal possible absolute value of $\mu(k, k_i)$ is denoted by $\mu = \max |\mu(k, k_i)|$ then, in the worst case, the amplitude of the strongest component $X(k_i)$ (assumed with the normalized amplitude 1), reduced for the maximal possible influence of other equally strong (unity) components $1 - (K-1)\mu$, should be greater than the maximal possible disturbance at $k \neq k_i$, being $K\mu$. From $1 - (K-1)\mu > K\mu$, the unique reconstruction condition follows.

In order to define other unique reconstruction conditions we will consider again the solution of $\mathbf{y} = \mathbf{A}_{MN}\mathbf{X}$ with a minimal number of nonzero coefficients in $\mathbf{X}$. Assume that the sparsity K is known. Then a set of K measurements would produce a possible solution for any combination of K nonzero coefficients in $\mathbf{X}$. If we assume that we have another set of K measurements, we would get another set of possible solutions. A common solution in these two sets of solutions would be the solution of our problem. There are no two different K sparse solutions $\mathbf{X}_K^{(1)}$ and $\mathbf{X}_K^{(2)}$ (the solution is unique) if all possible $\mathbf{A}_{M2K}^T \mathbf{A}_{M2K}$ matrices are nonsingular. The requirement that all reduced measurement matrices corresponding to a $2K$ sparse $\mathbf{X}$ are nonsingular can be written in several forms,

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

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

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

where d_i are the eigenvalues of $\mathbf{A}_{M2K}^T \mathbf{A}_{M2K}$, $d_{\min}$ is the minimal eigenvalue, $d_{\max}$ is the maximal eigenvalue, and δ_{2K} is the restricted isometry constant.

All these conditions are satisfied if $d_{\min} > 0$ or $0 \leq \delta_{2K} < 1$.

In noisy cases robustness is required, and therefore more strict bounds for $d_{\min}$ and δ_{2K} are required. For example, it has been shown, that $0 \leq \delta_{2K} < 0.41$ will guarantee stable inversion and robust reconstruction of the noisy signals. In addition, this bound will allow convex relaxation of the reconstruction problem.

Namely, the previous problem can be solved using the convex relation of the norm-zero to a norm-one formulation

$$\min \|\mathbf{X}\|_1 \text{ subject to } \mathbf{y} = \mathbf{A}_M \mathbf{X}.$$

The solution of these two problem formulation are the same if the measurement matrix satisfies the previous conditions with $\delta_{2K} < 0.41$. The signal reconstruction problem can now be solved using gradient-based approaches or linear programming methods.

4.3 Linear Combinations of Samples and Aggregated Sampling

If some spectrum coefficients are strongly related to only a few of the signal samples, then the signal samples may not be good candidates for the measurements. In that case linear combinations of all signal samples

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

should be used. The weighting coefficients $\mathbf{B}_{MN}$ for the measurements could be, for example, the Gaussian random numbers. The reconstruction is obtained as the solution of

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

or the solution of the corresponding convex minimization problem.

A specific form of linear combinations of the graph signals are described as aggregate sampling. Sampling in classical signal processing can be interpreted on a directed circular graph (Fig. 19) in the following way. Consider the signal at an instant n. If we sense this signal at this vertex/instant only, we get its value $y_0(n) = x(n)$. If we apply the shift operator we get $\mathbf{y}_1 = \mathbf{A}\mathbf{x}$. If this signal is sampled at the same vertex n we get $y_1(n) = x(n-1)$. If we continue this shift and sample operation N times we will get all signal values $x(n), x(n-1), \ldots, x(n-N+1)$. If we stop the shifts after $M < N$ steps the signal can still be recovered using the compressive sensing based reconstruction methods, if the reconstruction conditions are met.

The same procedure can be used on a signal on an arbitrary graph. Assume that we sample the graph signal at a vertex n and get

$$y_0(n) = x(n).$$

If the signal is now graph shifted $\mathbf{y}_1 = \mathbf{A}\mathbf{x}$ we will get a new measurement as a shifted signal value at the considered vertex,

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

If we continue with one more shift we will get

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

where $A^{(2)}_{nm}$ are the elements of matrix $\mathbf{A}^2 = \mathbf{A}\mathbf{A}$. If we continue $M = N$ times we would get a system of N linear equations $\mathbf{y} = \mathbf{B}_{MN}\mathbf{x}$. From these equations we can calculate all signal values $x(n)$. If we stop at $M < N$ the signal can still be recovered

using the compressive sensing based reconstruction methods if the signal is sparse and the reconstruction conditions are met.

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

4.4 On the Sampling Strategies

Signal sampling strategy is a process that will result in an optimal set of signal samples which will guarantee a unique reconstruction of a sparse signal. In classical signal processing, the sampling strategies are based on the minimization of parameters defining the solution uniqueness. In the case when the positions of nonzero GDFT coefficients are known (including the low-pass filtering as a special case) the requirement is that the rank of the measurement matrix is equal to the signal sparsity. In more general cases when the nonzero coefficient positions are not known, the restricted isometry property is the most commonly used uniqueness criterion. However, its application in practice is almost impossible, since it is an NP hard combinatorial problem (requiring $2K$ class combinations of N elements). A sampling strategy that will minimize the coherence index will guarantee the maximal number of nonzero coefficients reconstruction with a given set of signal samples. This process is less computationally intensive. However, the coherence index based criterion for signal sampling strategy is quite pessimistic.

In graph signals, the application of these sampling strategy criteria is even more difficult. The number of vertices may be extremely large. Large dimensionality of the graphs makes these approaches almost unsuitable for graph signal processing.

Subsampling of graphs is of crucial importance in the cases of extremely large graphs. Several approaches are possible in the graph analysis and graph signal processing.

We have already described a possible graph signal oriented subsampling strategy under the assumption that the GDFT is sparse, with a few nonzero coefficients. Then the graph signal can be reconstructed with a very reduced number of signal samples or their linear combinations.

Another class of approaches is graph oriented. The problem is defined as follows. Given a large, in general directed, graph $\mathcal{G}$ with N vertices, the goal is to find a much simpler graph with similar properties. In this so called down-scale method, the goal is to define a smaller size graph $\mathcal{S}$ with a very reduced number of vertices $M \ll N$ that will behave in a similar way as the original large graph. Similarity is defined with respect to the parameters of interest, like for example, the connectivity or distribution on graph. The criteria may be related to the spectral behavior of graphs as well.

Several methods are used for graph down-scaling. Some of them will be listed next:

- The simplest method for graph down-sampling is a random vertex or random node (RN) selection method. In this method, a random subset of vertices is used for analysis and representation of large graphs and signals on them. Vertices are selected with equal probabilities. This method will produce good results in many applications. Random selection has been preferred in classical compressive signal analysis as well.
- Different from the RN method, where the vertices are selected with a uniform probability, is the random degree vertex/node (RDN) selection, in which the probability that a vertex is selected is proportional to the vertex degree. Vertices with more connections, having larger $d_n = \sum_m W_{nm}$, are selected with a higher probability. This approach is biased with respect to highly connected vertices.
- A similar method to the RDN is based on the vertex rank (PageRank). The PageRank is defined by the importance of the vertices connected to the considered vertex n (see Sect. 7.8.5). Then the probability that a vertex n will be used in a down-scaled graph is proportional to the PageRank of this vertex. This method is called random PageRank vertex (RPN) selection. It is also biased with respect to the highly connected vertices with a high PageRank.
- A method based on the random selection of edges, that will be left in the simpler graph, is called the random edge (RE) method. This method may lead to graphs that are not well connected, with large diameters.
- The RE method may be combined with the random vertex selection to get a combined RNE method. It consists in a random vertex selection followed by a random selection of one of the edges corresponding to the selected vertex.
- In addition to these methods, more sophisticated methods based on the random vertex selection and the random walks (RW) analysis may be defined. For example, we can randomly select small subset of vertices and form several random walks starting from each selected vertex. In this way Random Walk (RW), Random Jump (RJ) and Forest Fire graph down-scaling strategies are defined.

4.5 Filter Bank on a Graph

Subsampling of a signal by a factor of 2, followed by the corresponding upsampling, can be described in classical signal processing by

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

This is the basic operation used in the filter-banks based signal processing. We can extend this concept to the graph signals. Consider a graph with the set of vertices $\mathcal{V}$. The set of vertices can be considered as a union of two disjoint sets $\mathcal{V} = \mathcal{E} \cup \mathcal{H}$ and $\mathcal{E} \cap \mathcal{H} = \emptyset$. The subsampling-upsampling procedure can be done in two steps:

1. Downsample the signal on graph by keeping the signal $x(n)$ values on the vertices $n \in \mathcal{E}$.

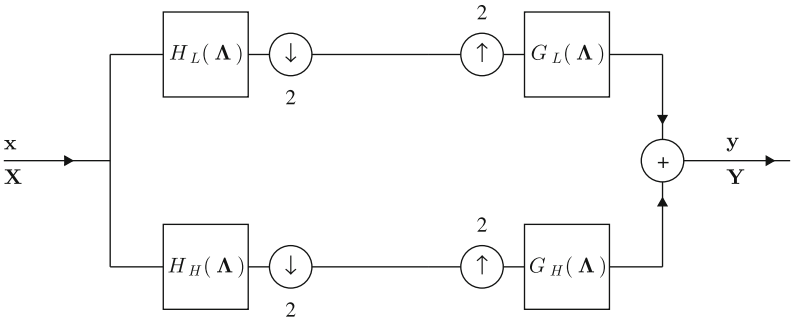


Fig. 31 A Filter bank on a graph signal

2. Upsample the graph signal by setting the values on $n \in \mathcal{H}$ to zero.

This combined downsampling-upsampling operation produces a graph signal

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

or in vector form

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

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

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

The spectral representation of this signal in the Laplacian basis functions domain is

$$\mathbf{F} = \mathbf{U}^T\mathbf{f} = \frac{1}{2}(\mathbf{U}^T + \mathbf{J}_{\mathcal{E}}\mathbf{U}^T)\mathbf{x} = \frac{1}{2}(\mathbf{I} + \mathbf{J}_{\mathcal{E}})\mathbf{X}.$$

If the downsampled-upsampled signal $\mathbf{F} = (\mathbf{I} + \mathbf{J}_{\mathcal{E}})\mathbf{X}/2$ passes through a system of low-pass analysis and syntheses filters $H_L(\mathbf{\Lambda})$ and $G_L(\mathbf{\Lambda})$, Fig. 31, the output is

$$\mathbf{F}_L = \frac{1}{2}H_L(\mathbf{\Lambda})(\mathbf{I} + \mathbf{J}_{\mathcal{E}})G_L(\mathbf{\Lambda})\mathbf{X}.$$

The same holds for the high-pass part

$$\mathbf{F}_H = \frac{1}{2}H_H(\mathbf{\Lambda})(\mathbf{I} + \mathbf{J}_{\mathcal{H}})G_H(\mathbf{\Lambda})\mathbf{X}.$$

The total output is a sum of these two signals, as illustrated in Fig. 31. After rearranging its terms we get

$$\mathbf{Y} = \mathbf{F}_L + \mathbf{F}_H = \tfrac{1}{2}(H_L(\mathbf{\Lambda})G_L(\mathbf{\Lambda}) + H_H(\mathbf{\Lambda})G_H(\mathbf{\Lambda}))\mathbf{X} + \tfrac{1}{2}(H_L(\mathbf{\Lambda})\mathbf{J}_{\mathcal{E}}G_L(\mathbf{\Lambda}) + H_H(\mathbf{\Lambda})\mathbf{J}_{\mathcal{H}}G_H(\mathbf{\Lambda}))\mathbf{X}. \tag{54}$$

The prefect reconstruction condition $\mathbf{Y} = \mathbf{X}$ is achieved if

$$\begin{aligned} H_L(\mathbf{\Lambda})G_L(\mathbf{\Lambda}) + H_H(\mathbf{\Lambda})G_H(\mathbf{\Lambda}) &= 2\mathbf{I} \\ H_L(\mathbf{\Lambda})\mathbf{J}_{\mathcal{E}}G_L(\mathbf{\Lambda}) + H_H(\mathbf{\Lambda})\mathbf{J}_{\mathcal{H}}G_H(\mathbf{\Lambda}) &= 0 \end{aligned} \tag{55}$$

A solution of this system is simple for bipartite graphs. It reduces to

$$G_L(\lambda)H_L(\lambda) + G_H(\lambda)H_H(\lambda) = 2 \tag{56}$$

$$G_L(\lambda)H_L(2-\lambda) - G_H(\lambda)H_H(2-\lambda) = 0 \tag{57}$$

A quadratic mirror filter solution is such that for the designed $H_L(\lambda)$, other filters are $G_L(\lambda) = H_L(\lambda)$, $H_H(\lambda) = H_L(2-\lambda)$, and $G_H(\lambda) = H_H(\lambda)$. For this solution the design equation is

$$H_L^2(\lambda) + H_L^2(2-\lambda) = 2.$$

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

5 Time-Varying Signals on Graphs

If we assume that the signal on graph changes in time at each vertex, then we have a signal $x_p(n)$ where n indicates the vertex index and p corresponds to the discrete-time index. If the signal sampling in time is uniform then the index p corresponds to $p\Delta t$ time instant.

In general, this kind of data can be considered within the Cartesian product framework, as shown in Fig. 9. The resulting graph $\mathcal{G} = (\mathcal{V}, \mathcal{B})$ follows as a Cartesian product of the given graph $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{B}_1)$ and a simple line (or circular) graph $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{B}_2)$ that corresponds to the classical time-domain signal processing.

The adjacency matrix of a Cartesian product of two graphs is

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

where $\mathbf{A}_1$ is the adjacency matrix of the given graph $\mathcal{G}_1$ and $\mathbf{A}_2$ is the adjacency matrix for the line or circle graph. The numbers of vertices in $\mathcal{G}_1$ and $\mathcal{G}_2$ are denoted by N_1 and N_2.

Next we will consider a simple and important example of a time-varying signal defined in an iterative way.

5.1 Diffusion on Graph and Low Pass Filtering

Consider the diffusion equation $\partial\mathbf{x}/\partial t = -\mathbf{L}\mathbf{x}$. Its discrete-time form obtained by using the backward difference approximation of the partial derivative is

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

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

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

The forward difference approximation of the diffusion equation results in

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

or

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

It is interesting to note that these iterative forms lead to the quadratic form of graph signal minimization. The signal quadratic form on a graph is $E_x = \mathbf{x}\mathbf{L}\mathbf{x}^T$, (see Sect. 3.6). If we want to find its minimum we can use the steepest descent method. Then the signal value at an instant p is changed in the opposite direction of the gradient, toward the energy minimum position. The gradient of quadratic form is $\partial E_x/\partial\mathbf{x}^T = 2\mathbf{x}\mathbf{L}$, resulting in the iterative procedure

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

This relation can be used for simple and efficient filtering of graph signals (with the aim to minimize E_x as the measure of the signal smoothness). If we assume that in one iteration the input graph signal is $\mathbf{x}_p$ and the output graph signal is $\mathbf{x}_{p+1}$, then the spectral domain relation of this system is

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

or

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

Obviously, the low varying components pass through this system since $(1-\alpha\lambda_k)$ is close to 1 for small λ_k, while the high varying components with larger λ_k are attenuated. Iterative procedure will converge if $|1-\alpha\lambda_{\max}|<1$. In the stationary case, when

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

all components $X_{p+1}(k)$ tend to 0 except the constant component $X_{p+1}(0)$, since $\lambda_0 = 0$. This component defines the stationary state (maximally smooth) solution. In order to avoid this effect, the iteration process can be used in alternation with

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

When these two iterative processes are used in a successive order the resulting system (Taubin's $\alpha-\beta$ algorithm) is

$$\mathbf{x}_{p+2} = (\mathbf{I}+\beta\mathbf{L})(\mathbf{I}-\alpha\mathbf{L})\mathbf{x}_p. \tag{58}$$

The resulting transfer function in the spectral domain in these two iteration steps is

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

After K iterations the transfer function is

$$H_K(\lambda_k) = ((1+\beta\lambda_k)(1-\alpha\lambda_k))^K. \tag{59}$$

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

The Graph from Fig. 3 and its Laplacian are considered in this example. For the parameters $\alpha = 0.1545$, $\beta = 0.1875$, the spectral transfer function (59) is presented in Fig. 32 for the considered graph filter. The results for the following numbers of iterations $K = 1, 5, 10, 20$ are shown. We have filtered the noisy signal from Fig. 26b. The initial noisy signal is denoted as $\mathbf{x}_0$. Then $\mathbf{x}_1 = (\mathbf{I}-0.1545\mathbf{L})\mathbf{x}_0$ is calculated with the Laplacian defined by (6). Next $\mathbf{x}_2 = (\mathbf{I}+0.1875\mathbf{L})\mathbf{x}_1$ is obtained. In the third and fourth iteration, the signal values $\mathbf{x}_3 = (\mathbf{I}-0.1545\mathbf{L})\mathbf{x}_2$ and $\mathbf{x}_4 = (\mathbf{I}+0.1875\mathbf{L})\mathbf{x}_3$ are calculated. This two-step iteration cycle is repeated $K = 20$ times. The resulting signal is almost the same as an output of the ideal low-pass filter presented in Fig. 26c.

6 Random Graph Signals

In this section we will present basic definitions of the random signals on graphs [59–64]. The conditions for wide-sense stationarity (WSS) will be considered. The stationarity is related to the signal shift on a graph. We have presented two approaches

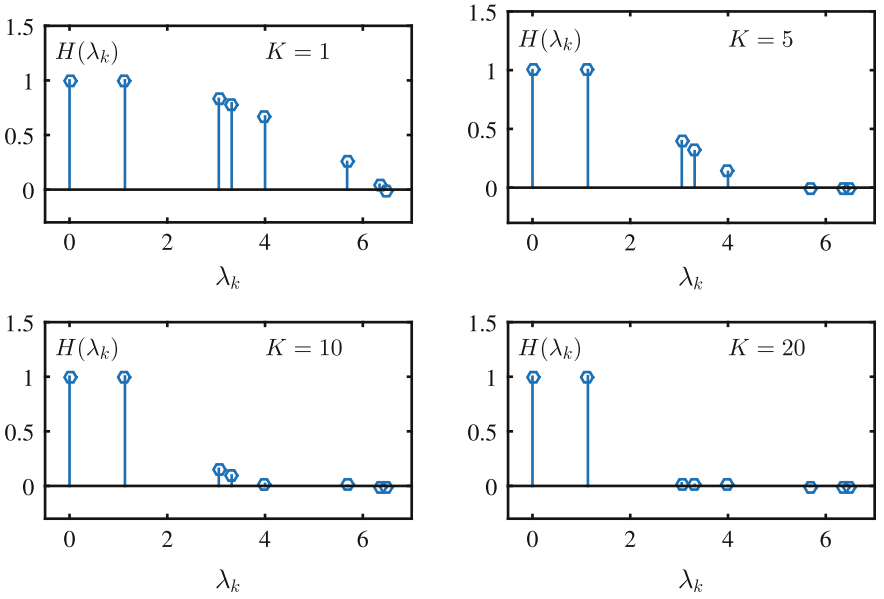


Fig. 32 Filter approximation in the spectral domain for various number of iterations K. The Graph from Fig. 3 and its Laplacian are considered

to define the shift on a graph (using the adjacency matrix or Laplacian and their spectral decompositions). The main problem is that the shift on a graph does not preserve energy (isometry property) $\|\mathbf{Ax}\|_2^2 \neq \|\mathbf{x}\|_2^2$. In order to define an equivalent of the WSS on graphs (GWSS) other properties of the WSS signals in the classical time domain processing are used. This is the reason for providing a short review of the classic signal processing WSS definitions and properties first.

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

A WSS random time-domain signal $x(n)$ can be considered as an output of a liner shift invariant system with impulse response $h(n)$ to a white noise input $\varepsilon(n)$ with $r_\varepsilon(n, m) = \delta(n-m)$.

In the time domain, the eigenvectors $\mathbf{u}_k$ of the shift operator $y(n) = x(n-1)$ are the DFT basis functions, $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H$. For a random signal, its DFT $\mathbf{X} = \mathbf{U}^H\mathbf{x}$ is a random signal with the autocorrelation matrix $\mathbf{P}_x = \mathrm{E}\{\mathbf{X}\mathbf{X}^H\}$, where $\mathbf{U}^H$ is the DFT transformation matrix. For WSS signals, the matrix $\mathbf{P}_x$ is diagonal with the power spectral density (PSD) values

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

on diagonal.

For WSS random signals $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^H\}$ is diagonalizable with the same transform matrix $\mathbf{U}$ as in $\mathbf{X} = \mathbf{U}^H\mathbf{x}$,

$$\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^H\} = \mathrm{E}\{\mathbf{U}\mathbf{X}(\mathbf{U}\mathbf{X})^H\} = \mathbf{U}\mathrm{E}\{\mathbf{X}\mathbf{X}^H\}\mathbf{U}^H = \mathbf{U}\mathbf{P}_x\mathbf{U}^H \tag{60}$$

since $\mathbf{P}_x$ is a diagonal matrix for WSS signals.

These properties of the WSS signals in classical analysis will be used for the graph signals next.

6.1 Adjacency Matrix Based Definition

Consider a white noise signal ε on a graph with samples $\varepsilon(n)$. A signal $\mathbf{x}$ on the graph is graph wide sense stationary (GWSS) if it can be considered an output of a linear and shift invariant graph system $H(\mathbf{A}) = \sum_{m=0}^{M-1} h_m\mathbf{A}^m$ to the white noise input ε,

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

The autocorrelation matrix $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^H\}$ of a GWSS signal is diagonalizable with the matrix of the adjacency matrix $\mathbf{A}$ eigenvectors

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H \tag{61}$$

$$\mathrm{E}\{\mathbf{x}\mathbf{x}^H\} = \mathbf{U}\mathbf{P}_x\mathbf{U}^H, \tag{62}$$

where $\mathbf{P}_x$ is a diagonal matrix. The values on the diagonal of matrix $\mathbf{P}_x$ denoted by $\mathbf{p}_x$ represent the PSD of a graph signal $p_x(k) = \mathrm{E}\{|X(k)|^2\}$.

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

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

Using $H(\mathbf{A}) = \mathbf{U}^T H(\mathbf{\Lambda})\mathbf{U}$ we get

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

what concludes the proof. The diagonal matrix is

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

with the PSD of this signal

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

Periodogram on the graph can be estimated using K realizations of the random signal denoted by $\mathbf{x}_i$. It is equal to the diagonal elements of matrix

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

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

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

Wienner Filter on a Graph

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

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

In the spectral domain the system is described by

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

Assume that the signal and the noise are statistically independent. The noise is a zero-mean random signal. The aim is to find a filter $G(\mathbf{\Lambda})$ such that $\mathbf{Y} = G(\mathbf{\Lambda})\mathbf{X}$ estimates $\mathbf{S}$ in the mean squared sense. We will minimize

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

The zero value of the derivative with respect to the elements of $G(\mathbf{\Lambda})$ produces

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

Since all matrices are diagonal we may use symbolic matrix division resulting in

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

or

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

In a non-noisy case, $E(k) = 0$ for all k, the inverse filter follows, as expected.

6.2 Spectral Domain Shift Based Definition of GWSS

This approach is based on the shift on a graph defined using the graph filter response

$$\mathcal{T}_m\{h(n)\} = h_m(n) = \sum_{k=0}^{N-1} H(\lambda_k) u_k(m) u_k(n). \tag{63}$$

The matrix form of this relation is

$$\mathcal{T}_h = H(\mathbf{L}) = \sum_{m=0}^{M-1} h_m \mathbf{L}^m = \mathbf{U} H(\mathbf{\Lambda}) \mathbf{U}^H, \tag{64}$$

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

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

The definition of the GWSS within this framework reads: The signal is GWSS if its autocorrelation function is invariant with respect to the shift $\mathcal{T}_m\{r_x(n)\}$

$$r_x(m) = \mathcal{T}_m\{r_x(n)\}.$$

For a GWSS signal the autocorrelation matrix $\mathbf{R}_x$ is diagonalizable with the matrix of eigenvectors of the Laplacian $\mathbf{L}$

$$\mathbf{L} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T. \tag{65}$$

For the basic autocorrelation we can assume that

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

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

where

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

is a diagonal matrix.

Finally, we will mention one more approach based on the shift operator defined as $\mathcal{T}_m = \exp(j\pi\sqrt{\mathbf{L}/\rho})$. It maps the eigenvalues of the Laplacian $\mathbf{L}$ on a unit circle, preserving the isometry.

7 Examples of Graphs Topologies

In the previous section, we have assumed that the graph is defined and the signal processing on the given graph topology is considered. However, the graph topology is very often a part of the processing problem rather than it being given within the problem definition. In this case, we may assume that the vertices are given, while the edges and their weights are part of the problem solution [65–82].

In general, we will consider three possible scenarios for the graph edges:

- The first group of problems is related to the geometry of the vertex positions. In the cases of various sensing setups (temperature, pressure, transportation, …) the locations of the sensing positions (vertices) are known and the vertex distances play a crucial role in data relations.
- The second group consists of problems where the relation among the sensing positions are physically well defined. Examples of such problems are electric circuit networks, linear heat transfer, social and computer networks, spring-mass systems, …. In these cases the edge weights are well defined based on the physics of the considered problem.
- In the third group we will include the problems where the data similarity plays a crucial role for the underlying graph topology. Various approaches are used to define the data similarity.

7.1 Geometric Graph Topologies

For a graph corresponding to a geometric network, the edge weights are related to the vertices distance. Consider vertices n and m whose positions in space are defined by position vectors $\mathbf{r}_n$ and $\mathbf{r}_m$. The Euclidean distance between these two vertices is

$$r_{nm} = \text{distance}(m, n) = \|\mathbf{r}_n - \mathbf{r}_m\|_2 .$$

A common way to define the graph weights in such networks is

$$W_{nm} = \begin{cases} e^{-r_{nm}^2/\tau^2} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa, \end{cases} \tag{66}$$

where r_{nm} is the Euclidean distance between the vertices n and m, and τ and κ are constants. The weights tend to 1 for close vertices. The weights are 0 or close to 0 for distant vertices.

The basic idea behind this definition of the edge weights is the assumption that the signal value measured at vertex n is similar to signal values measured at the neighboring vertices. According to this definition, the processing of a signal at vertex n should include close vertices with higher weights (close to 1) while the signal values

sensed at farther vertices would be less relevant. They are included with smaller weighting coefficients or not included at all.

While the Gaussian function, used in (66), is the most appropriate in many applications, other forms to penalize signal values at the vertices far from the considered vertex can be used. Examples of such functions are

$$W_{nm} = \begin{cases} e^{-r_{nm}/\tau} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa \end{cases} \tag{67}$$

or

$$W_{nm} = \begin{cases} 1/r_{nm} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa. \end{cases} \tag{68}$$

The simplest form of the weighting coefficients would be

$$W_{nm} = \begin{cases} 1 & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa. \end{cases} \tag{69}$$

This form would correspond to an unweighted graph with $\mathbf{W} = \mathbf{A}$.

As an example, consider the Minnesota roadmap graph. The edges of the given adjacency matrix are weighted according to distances using (67) with $\tau = 25$ km and κ is not used since the connectivity is already determined by the given adjacency matrix.

A simulated signal and its noisy version are given in Fig. 33a, b. The noisy signal is filtered by low-pass filter in the Laplacian spectral domain using the 50 lowest changing spectral components. Filtering is also done using the two-step iterative procedure (58) with 200 iterations using $\alpha = 0.1$ and $\beta = 0.15$. Filtered signals are presented in Fig. 33c, d. The input SNR is 8.7 dB and the output SNR of 23.0 and 23.3 dB is achieved.

A more general form of smoothing would be obtained using weights with appropriate low-pass filter coefficients in the adjacency or Laplacian spectrum, like for example the one presented by (15).

In classical signal analysis, using

$$\text{distance}(m, n) = r_{nm} = \|n - m\|_2 = |n - m|,$$

and $W_{nm} = e^{-r_{nm}^2/\tau^2}$ for $r_{nm} \leq \kappa$ and $W_{nm} = 0$ for $r_{nm} > \kappa$, the value

$$\mathbf{y} = \mathbf{A}^0\mathbf{x} + \mathbf{A}^1\mathbf{x}$$

produces a Gaussian weighted mean, localized around the vertex (time instant) n (moving average). For example, for $\tau = 4$ and $\kappa = 8$ it would produce the time domain weighted moving average

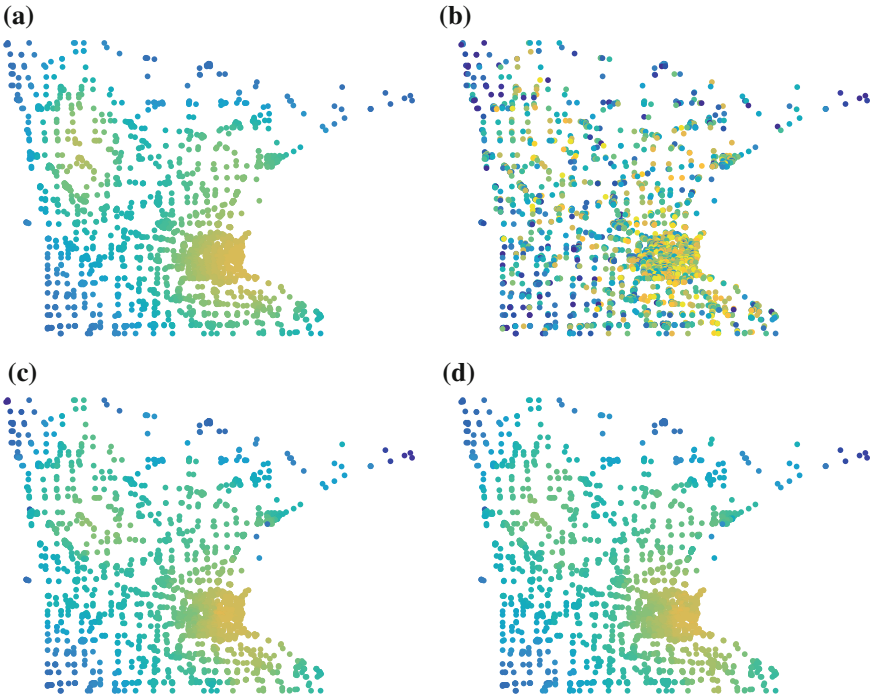


Fig. 33 Minnesota roadmap graph: **a** simulated signal, **b** noisy signal, **c** low-pass filtering, and **d** iterative filtering example

$$y(n) = x(n) + \sum_{m} W_{nm} x(n-m) = \sum_{m=-8}^{8} e^{-\frac{(n-(n-m))^2}{16}} x(n-m).$$

Unweighted moving average within $-\kappa \leq m \leq \kappa$ would be obtained with a large τ. If the Euclidean distance between pixels is used in an image, we would get a moving average filtered image with a radial Gaussian window.

7.2 Topology Based on the Signal Similarity

In the previous section the graph weights are defined assuming that the geometric distance of vertices, where the signal is sensed, is a good and reliable indicator of the data similarity. That may be the case in some applications like, for example, the measurements of atmospheric temperature and pressure when the terrain configuration has no influence to the similarity of measured data. However, in general the geometric distance of vertices may not be a good indicator of data similarity.

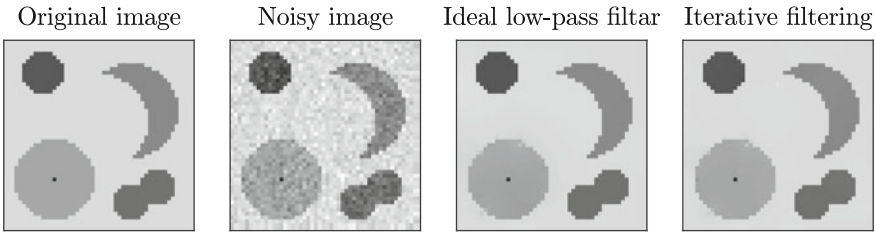


Fig. 34 Original and noisy images (left) and filtered images (right) using frequency domain low-pass filter and iterative vertex domain filtering. The image is 8bit grayscale. Edge weights are calculated with $\kappa = \sqrt{2}$ and $\tau = 20$. Low-pass filter in frequency domain is done using an ideal low-pass filter with 10 lowest spectral components. Iterative filtering (58) is performed with 200 iterations using $\alpha = 0.1$ and $\beta = 0.15$

In some applications like, for example, image processing, the signal value by themselves can be used as an indicator of the signal similarity, often in combination with the pixel/vertex distances. If the image intensity values at pixels indexed by n and m are denoted by $x(n)$ and $x(m)$ then the difference of intensities is defined by

$$\text{Intensity distance}(m, n) = s_{nm} = |x(n) - x(m)|.$$

Then the weights can be defined as

$$W_{nm} = \begin{cases} e^{-(x(n)-x(m))^2/\tau^2} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa \end{cases}$$

where r_{nm} is a geometric distance of the considered pixels/vertices.

An example with these kinds of weights applied to simple image graph filtering is presented in Fig. 34.

In some applications we are able to collect more than one data set for a given set of sensing points/vertices. In that case a more reliable measure of data similarity can be defined. Assume that at each vertex $n = 0, 1, \ldots, N-1$ we have acquired P signal values denoted by $x_p(n)$. This data set may be a set of multivariate data or signal measurements in a sequence. Then a good similarity measure function for real-valued signal at vertices n and m is

$$s_{nm}^2 = \frac{\sum_{p=1}^{P} \left(x_p(n) - x_p(m)\right)^2}{\sqrt{\sum_{p=1}^{P} x_p^2(n) \sum_{p=1}^{P} x_p^2(m)}}.$$

The graph weights can again be defined using any of the previous penalty functions, for example,

$$W_{nm} = \begin{cases} e^{-s_{nm}^2/\tau^2} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa. \end{cases}$$

The function of the form

$$W_{nm} = \begin{cases} e^{-s_{nm}/\tau} & \text{for } r_{nm} \leq \kappa \\ 0 & \text{for } r_{nm} > \kappa. \end{cases}$$

is also used quite often.

As an example assume that $x_p(n)$ in P observations, $p = 1, 2, \ldots, P$ at N vertices $n = 0, 1, \ldots, N-1$ are zero-mean random noises with equal variances $\sigma_x^2 = 1$. Then

$$s_{n,m}^2 = \frac{\sum_{p=1}^{P} \left(x_p(n) - x_p(m)\right)^2}{\sqrt{\sum_{p=1}^{P} x_p^2(n) \sum_{p=1}^{P} x_p^2(m)}} = 2(1 - R_x(n,m))$$

where

$$R_x(n,m) = \frac{1}{P} \sum_{p=1}^{P} x_p(n) x_p(m)$$

is the normalized autocorrelation function.

The same structure can be used for image classification, handwriting recognition or in establishing block similarity in large images. In these cases the distance between image/block n and image/block m is equal to

$$s_{nm} = \text{Image/Block distance}(m,n) = \|\mathbf{x}_n - \mathbf{x}_m\|_F,$$

where $\|\mathbf{x}\|_F$ is the Frobenius norm of an image or block matrix $\mathbf{x}$ (that is, the square root of the sum of all its squared elements).

7.3 Correlation Based Graph Laplacian learning

Consider a graph signal with P independent observations. Denote the observed signal at vertex n and observation p as $x_p(n)$. The column vector with graph signal samples from the pth observation is denoted by $\mathbf{x}_p$. All observations of this graph signal can be arranged into an $N \times P$ matrix

$$\mathbf{X}_P = \left[\mathbf{x}_1 \;\; \mathbf{x}_2 \;\; \ldots \;\; \mathbf{x}_P \right].$$

Denote the nth row of this matrix as a row vector $\mathbf{y}_n$

$$\mathbf{y}_n = \left[x_1(n-1) \;\; x_2(n-1) \;\; \ldots \;\; x_P(n-1) \right]$$

Then the matrix of observations can also be written as

$$\mathbf{X}_P = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_N \end{bmatrix}.$$

The correlation coefficients estimated by averaging over the observations are

$$R_x(n,m) = \frac{1}{P}\sum_{p=1}^{P} x_p(n)x_p(m) = \frac{1}{P}\mathbf{y}_n\mathbf{y}_m^T$$

or in a matrix form

$$\mathbf{R}_x = \frac{1}{P}\mathbf{X}_P\mathbf{X}_P^T.$$

Since the correlation includes signal from all vertices, this matrix accumulates all correlations obtained through all possible walks from the current vertex to any other vertex. It means that the correlation coefficient for two vertices will produce misleading results if there exit one or more other vertices where the signal is strongly related to both of the considered vertices. That is the reason why the correlation overestimates the direct connections. It is not a good parameter for establishing direct links (edges) between vertices. Additional conditions should be imposed on the correlation matrix or other statistical parameters should be be used for edge weights estimation. Next we will present a method for the connectivity (edge weights) estimation based on the correlations, imposing the sparsity constraint on the weight coefficients.

Consider the vertex 0 with $n = 1$, Eq. (7.3). We can estimate the edge weights from this vertex to all other vertices β_{1m}, $m = 2, 3, \ldots, N$ by minimizing

$$J_1 = \|\mathbf{y}_1 - \sum_{m=2}^{N} \beta_{1m}\mathbf{y}_m\|_2^2 + \lambda \sum_{m=2}^{N} |\beta_{1m}|$$

The first term promotes the correlation between the observations $\mathbf{y}_1$ at the considered vertex with $n = 1$ and the observations $\mathbf{y}_m$ at all other vertices, $m = 2, 3, \ldots, N$. The second term promotes sparsity in coefficients β_{1m} (number of nonzero coefficients β_{1m}). Parameter λ balances these two conditions. Full matrix form of the previous cost function is

$$J_1 = \|\mathbf{y}_1^T - \mathbf{Y}_1^T\boldsymbol{\beta}_1\|_2^2 + \lambda\|\boldsymbol{\beta}_1\|_1$$

where $\mathbf{Y}_1$ is obtained from the matrix $\mathbf{X}_P$ with the first row being removed and

$$\boldsymbol{\beta}_1 = \begin{bmatrix} \beta_{12} & \beta_{13} & \ldots & \beta_{1N} \end{bmatrix}^T.$$

This problem can be solved using commonly defined LASSO minimization as $\boldsymbol{\beta}_1 = \text{lasso}(\mathbf{Y}_1^T, \mathbf{y}_1^T, \lambda)$, see the appendix.

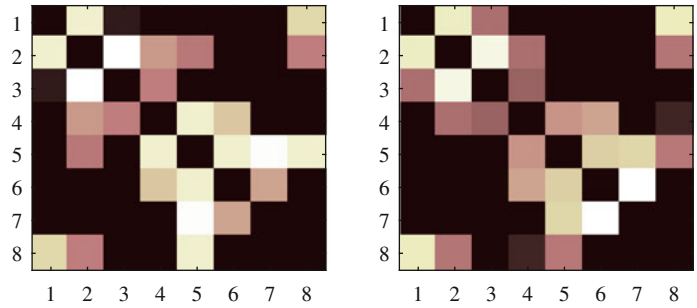


Fig. 35 Groundtruth (left) and estimated weighting matrix (right). The axis index n corresponds to the vertex $n-1$

The minimization is repeated for all vertices with $n = 1, 2, \ldots, N$

$$J_n = \|\mathbf{y}_n^T - \mathbf{Y}_n^T \boldsymbol{\beta}_n\|_2^2 + \lambda \|\boldsymbol{\beta}_n\|_1.$$

Since this procedure does not guarantee symmetry $\beta_{nm} = \beta_{mn}$ the edge weights could be calculated as $W_{nm} = \sqrt{\beta_{nm}\beta_{mn}}$.

As an example, consider the graph from Fig. 3 and $P = 10000$ observations. Observations are simulated by assuming white Gaussian external sources with zero-mean and variance 1 located at two randomly chosen vertices (see Appendix 8.1 and Fig. 45). An $N \times P$ matrix of signal $\mathbf{X}_P$ is formed. Using its rows the vector $\mathbf{y}_n$ and matrix $\mathbf{Y}_n$ are obtained. The matrix of coefficients β_{nm} follows from lasso($\mathbf{Y}_n^T$, $\mathbf{y}_n^T$, λ) with $n = 1, 2, \ldots, 8$ and $\lambda = 1.7$ as

$$\boldsymbol{\beta} = \begin{bmatrix} 0 & 0.24 & 0 & 0 & 0 & 0 & 0 & 0.36 \\ 0.55 & 0 & 0.86 & 0.22 & 0 & 0 & 0 & 0.18 \\ 0 & 0.31 & 0 & 0.13 & 0 & 0 & 0 & 0 \\ 0 & 0.19 & 0 & 0 & 0.28 & 0.18 & 0 & 0 \\ 0 & 0.01 & 0 & 0.39 & 0 & 0.42 & 0.43 & 0.36 \\ 0 & 0 & 0 & 0.18 & 0.19 & 0 & 0.44 & 0 \\ 0 & 0 & 0 & 0 & 0.23 & 0.30 & 0 & 0 \\ 0.32 & 0.16 & 0 & 0 & 0.21 & 0 & 0 & 0 \end{bmatrix}.$$

The groundtruth and estimated weights are presented in Fig. 35.

The same experiment is repeated for the unweighted graph from Fig. 2a. The result is presented in Fig. 36. In this case the obtained values of $\boldsymbol{\beta}$ are used to decide weather $A_{mn} = 1$ or $A_{mn} = 0$.

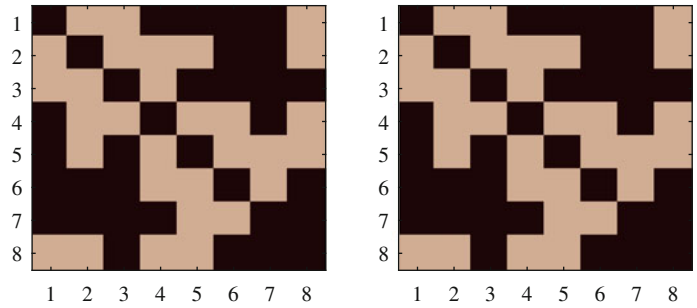


Fig. 36 Groundtruth (left) and estimated adjacency matrix (right) for unweighted graph

7.4 Graph Laplacian Learning with the Smoothness Constraint

Consider a set of noisy signal values $x_p(n)$ in P observations, $p = 1, 2, \ldots, P$ on N vertices $n = 0, 1 \ldots, N-1$ of an undirected graph. The goal is to get the graph connectivity (its Laplacian). To this aim we will calculate a signal $y_p(n)$ that is close to the observations $x_p(n)$ under the condition that $y_p(n)$ is also as smooth as possible on a graph. This formulation is similar to the one described and explained in Sect. 3.12. The signal $y_p(n)$ can be found by minimizing the cost function

$$J_p = \frac{1}{2}\|\mathbf{y}_p - \mathbf{x}_p\|_2^2 + \alpha \mathbf{y}_p^T \mathbf{L} \mathbf{y}_p, \text{ for } p = 1, 2, \ldots, P.$$

The difference here is that the Laplacian (graph edges and their weights) is unknown as well. It has to be determined along with the output signal $\mathbf{y}_p$.

Since we have P observations we can form $N \times P$ matrices

$$\mathbf{X}_P = \left[\mathbf{x}_1 \;\; \mathbf{x}_2 \;\; \ldots \; \mathbf{x}_P \right] \text{ and } \mathbf{Y}_P = \left[\mathbf{y}_1 \;\; \mathbf{y}_2 \;\; \ldots \; \mathbf{y}_P \right].$$

The cost function for the whole set of observations is

$$J = \sum_{p=1}^{P} J_p = \frac{1}{2}\|\mathbf{Y}_P - \mathbf{X}_P\|_F^2 + \alpha \text{Trace}\{\mathbf{Y}_P^T \mathbf{L} \mathbf{Y}_P\} + \beta \|\mathbf{L}\|_F^2,$$

where $\text{Trace}\{\mathbf{Y}_P^T \mathbf{L} \mathbf{Y}_P\}$ is the matrix form of the term $\mathbf{y}_p^T \mathbf{L} \mathbf{y}_p$ and the constraint about the energy of Laplacian $\|\mathbf{L}\|_F^2 = \sum_n \sum_m L_{mn}^2$ is added in order to keep its values as low as possible.

It has been assumed that the the Laplacian is normalized. In order to avoid trivial solutions, the condition

$$\text{Trace}\{\mathbf{L}\} = N$$

is used as well, along with

$$L_{mn} = L_{nm} \leq 0 \text{ for } n \neq m, \text{ and } \sum_{m=0}^{N-1} L_{nm} = 0.$$

The problem is jointly convex with respect to the signal and Laplacian. It is solved in an iterative two-step procedure:

(1) Assume that

$$\mathbf{Y}_P = \mathbf{X}_P.$$

(2) Estimate Laplacian $\mathbf{L}$ by minimizing

$$J_1 = \alpha \text{Trace}\{\mathbf{Y}_P^T \mathbf{L} \mathbf{Y}_P\} + \|\mathbf{L}\|_F^2$$

with given conditions for the Laplacian form.

(3) For the obtained Laplacian in step (2), the signal $\mathbf{Y}_P$ is calculated minimizing

$$J_2 = \frac{1}{2}\|\mathbf{Y}_P - \mathbf{X}_P\|_F^2 + \alpha \text{Trace}\{\mathbf{Y}_P^T \mathbf{L} \mathbf{Y}_P\}.$$

Steps (2) and (3) are iteratively repeated. Step (3) has a closed form solution as presented in Sect. 3.12.

7.5 Generalized Laplacian Learning

The generalized Laplacian $\mathbf{Q}$ is defined as

$$\mathbf{Q} = \alpha \mathbf{I} - \mathbf{N},$$

where $\mathbf{N}$ is a nonnegative symmetric matrix and $\mathbf{Q}$ is a symmetric positive semidefinite matrix. Any generalized Laplacian can be written as a sum of a standard Laplacian $\mathbf{L}$ and a diagonal matrix $\mathbf{P}$

$$\mathbf{Q} = \mathbf{L} + \mathbf{P}.$$

The generalized Laplacian allows self-loops on the vertices. They are defined by $\mathbf{P}$.

Consider a set of noisy signals $x_p(n)$ and their P observations, $p = 1, 2, \ldots, P$ on N vertices $n = 0, 1 \ldots, N-1$ of an undirected graph. The main goal is again to get the graph connectivity (its Laplacian) from the condition that the observed signal is as smooth as possible on the graph defined by a generalized Laplacian $\mathbf{Q}$. The cost function to achieve this goal is defined by the signal smoothness function

$$J_p = \mathbf{x}_p^T \mathbf{Q} \mathbf{x}_p, \text{ for } p = 1, 2, \ldots, P.$$

The correlation matrix of the considered observations is $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} \approx \frac{1}{P}\sum_{p=1}^{P}\mathbf{x}_p\mathbf{x}_p^T$. The smoothness promoting cost function for all data is

$$J = \sum_{p=1}^{P}\mathbf{x}_p^T\mathbf{Q}\mathbf{x}_p = \mathrm{Trace}\{\mathbf{X}_P\mathbf{Q}\mathbf{X}_P^T\} = \mathrm{Trace}\{\mathbf{R}_x\mathbf{Q}\}.$$

Next the conditions for the generalized Laplacian should be added (otherwise a trivial solution would be obtained). For symmetric positive definite matrices, all eigenvalues must be positive. Since the product of the eigenvalues is equal to $\det(\mathbf{Q})$, this condition is included by adding the term $\log(\det(\mathbf{Q}))$ to the cost function. Then the cost function is of the form

$$J = \log(\det(\mathbf{Q})) + \mathrm{Trace}\{\mathbf{R}_x\mathbf{Q}\}.$$

The interpretation of this cost function within the Gaussian random signal and maximum likelihood estimate is given in Sect. 7.9. This cost function minimizes the logarithm of the joint probability density function of a graph signal $\mathbf{x}_p$ under the Gaussianity assumption,

$$P(\mathbf{x}_p) \sim \det(\mathbf{Q})\exp\left(-\frac{1}{2}\mathbf{x}_p\mathbf{Q}\mathbf{x}_p\right).$$

Minimization of the cost function J with respect to $\mathbf{Q}$, with $\partial J/\partial\mathbf{Q} = 0$, produces

$$\mathbf{Q} = \mathbf{R}_x^{-1}.$$

Here we have used the relation between the trace of a positive semidefinite matrix and the trace of its eigenvalue matrix

$$\log(\det(\mathbf{Q})) = \sum_{i=k}^{N}\log(\lambda_k) = \mathrm{Trace}(\log(\mathbf{\Lambda})) = \mathrm{Trace}(\log(\mathbf{Q})).$$

The solution of the previous equation $\mathbf{Q} = \mathbf{R}_x^{-1}$, can be used as the generalized Laplacian estimate and the graph is obtained.

The weighting matrix corresponding to the inverse correlation matrix $\mathbf{R}_x$, with positive and small off diagonal values set to zero is shown in Fig. 37 (right). Here we consider the graph from Fig. 3 and $P = 10000$ observations. The observations are simulated by assuming white Gaussian external sources with zero-mean and variance 1 located at two randomly chosen vertices (as described in more details in Sect. 7.3).

The correlation function matrix $\mathbf{R}_x$ may be singular. It is always singular when $N > P$. Also, this form will not produce a matrix satisfying the conditions to be a generalized Laplacian. The inverse correlation function may have positive off-diagonal

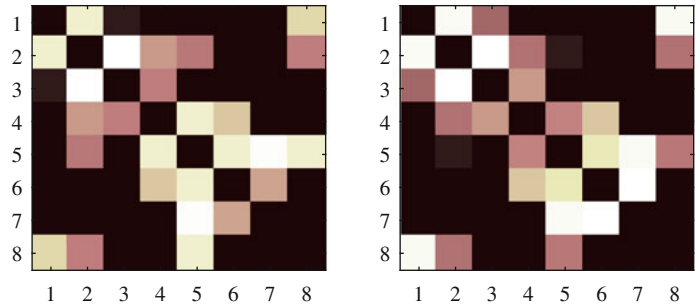


Fig. 37 Groundtruth (left) and weighting matrix estimated using inverse correlation (right). The axis index n corresponds to the vertex $n-1$

values. Therefore the previous cost function should have additional constraints. Here we will present two of them.

In the first approach, a term corresponding to the Lagrange multipliers $\mathbf{B}$ is added so that these values do not change the diagonal elements of $\mathbf{Q}$ and provide that all

$$Q_{mn} = Q_{nm} \leq 0$$

for $n \neq m$, with $B_{nm} = B_{mn} \geq 0$. The diagonal elements of matrix $\mathbf{B}$ are $B_{nn} = 0$. Finally, $B_{nm}Q_{nm} = 0$ for all n and m. In this case the minimization solution for the generalized Laplacian is obtained as

$$\mathbf{Q} = (\mathbf{R}_x + \mathbf{B})^{-1}$$

using the cost function

$$J = \log(\det(\mathbf{Q})) + \text{Trace}\{\mathbf{R}_x\mathbf{Q}\} + \text{Trace}\{\mathbf{BQ}\}.$$

Another possible approach corresponds to the classical reconstruction formulation of a sparse signal. In this case the sparsity constraint on the generalized Laplacian is added. The cost function is defined as

$$J = \log(\det(\mathbf{Q})) + \text{Trace}\{\mathbf{R}_x\mathbf{Q}\} + \beta\|\mathbf{Q}\|_1.$$

This minimization problem can be solved using various methods. One of them is the graphical LASSO algorithm, an extension of the standard LASSO algorithm to the graph problems (see appendix). For the same signal as in the previous examples, the weighting matrix obtained using the graphical LASSO, with positive and small values set to zero, is shown in Fig. 38 (right).

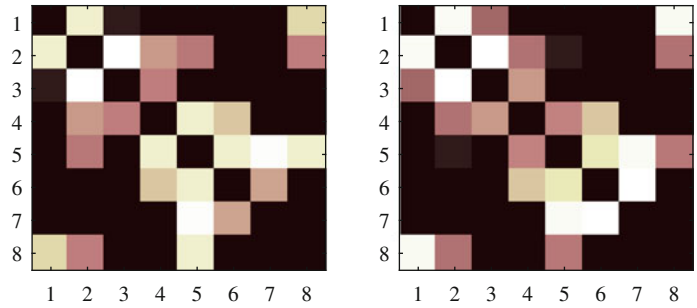


Fig. 38 Groundtruth (left) and weighting matrix estimated using graphical LASSO (right). The axis index n corresponds to the vertex $n-1$

7.6 Graph Topology with Some a Priori Knowledge

Consider a random signal $x(n)$ and its P realizations on a graph. Assume that a random graph signal is a result of white noise signals ε as external sources in each vertex (in the sense as presented in Fig. 40). Then the external source to the vertex signal relation is

$$\mathbf{L}\mathbf{x} = \boldsymbol{\varepsilon}$$

or $\mathrm{E}\{\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T\} = \mathbf{L}\,\mathrm{E}\{\mathbf{x}\mathbf{x}^T\}\mathbf{L}^T$ resulting in $\mathbf{I} = \mathbf{L}\mathbf{R}_x\mathbf{L}^T$. The solution is

$$\mathbf{L}^2 = \mathbf{R}_x^{-1} \text{ or } \mathbf{L} = \mathbf{R}_x^{-1/2}.$$

Assume now that the graph signal is considered with a reference vertex. For the notation simplicity assume, without loss of generality, that the reference vertex is denoted by $N-1$ (the last vertex). Assume also that the value of the graph signal at this reference vertex is $x_p(N-1) = 0$ for any p. If this is not the case then we can achieve this by subtracting $x_p(N-1)$ from each $x_p(n)$. This operation will not change the properties of the graph signal. In this case the correlation matrix is singular and assumes the form

$$\mathbf{R}_x = \begin{bmatrix} \mathbf{R}_{N-1} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix},$$

where $\mathbf{R}_{N-1}$ is the correlation of the vertices $n = 0, 1, \ldots, N-2$. Assuming non-singular $\mathbf{R}_{N-1}$ the Laplacian can be calculated as

$$\mathbf{L} = \mathbf{R}_x^{-1/2} = \begin{bmatrix} \sqrt{\mathbf{R}_{N-1}^{-1}} & \mathbf{a} \\ \mathbf{a}^T & -\sum_{m=0}^{N-2} a_m \end{bmatrix},$$

where the column vector $\mathbf{a} = [a_0\ a_1\ \ldots\ a_{N-2}]^T$ elements are added so that

$$\sum_{m=0}^{N-2} L_{nm} + a_n = 0$$

holds for each row and column. The matrix square root operation is used.

For the graph from Fig. 7 and random graph signal generated using random white external sources the Laplacian is obtained from

$$\mathbf{R}_x = \begin{bmatrix} 2.92 & 2.91 & 3.53 & 2.98 & 2.50 & 3.08 & 3.05 & 0 \\ 2.91 & 3.78 & 4.61 & 4.14 & 3.55 & 4.43 & 4.42 & 0 \\ 3.53 & 4.61 & 6.27 & 5.13 & 4.27 & 5.33 & 5.26 & 0 \\ 2.98 & 4.14 & 5.13 & 5.43 & 4.71 & 6.05 & 6.05 & 0 \\ 2.50 & 3.55 & 4.27 & 4.71 & 4.52 & 5.73 & 6.01 & 0 \\ 3.08 & 4.43 & 5.33 & 6.05 & 5.73 & 7.76 & 7.77 & 0 \\ 3.05 & 4.42 & 5.26 & 6.05 & 6.01 & 7.77 & 8.81 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This approach can be generalized to any known

$$H(\mathbf{L})\mathbf{x} = \varepsilon$$

or $H(\mathbf{L})\varepsilon = \mathbf{x}$. In this case a matrix polynomial equation has to be solved for the Laplacian.

7.7 Graph Topology Based on the Eigenvectors

Assume that the available observations of a graph signal $x_p(n)$ are graph wide sense stationary (GWSS). A graph signal's observations $\mathbf{x}_p$ are GWSS if they can be considered as the outputs of a linear and shift invariant system $H(\mathbf{A})$ to the white noise inputs ε_p, with $\mathbf{x} = H(\mathbf{A})\varepsilon$, see Sect. 6.1. The correlation matrix of the observed signal can be written as

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

where $\mathbf{U}$ is the matrix of graph eigenvectors $\mathbf{L} = \mathbf{U}^T \mathbf{\Lambda} \mathbf{U}$. The autocorrelation matrix estimation is done using all available observations $\mathbf{R}_x = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} \approx \frac{1}{P} \sum_{p=1}^{P} \mathbf{x}_p \mathbf{x}_p^T$. From the autocorrelation matrix decomposition we can learn about the graph eigenvectors. The same holds for the precision matrix $\mathbf{\Theta} = \mathbf{R}_x^{-1}$ since the inverse matrix has the same eigenvectors as the original matrix.

In order to estimate the graph connectivity (estimate its Laplacian or adjacency matrix) we can use the autocorrelation matrix eigenvectors. Since we do not know $H(\mathbf{\Lambda})$, it will be assumed that the graph is defined by the eigenvalues that produces the smallest number of edges. This can be done by minimizing the number of nonzero values in $\mathbf{L}$ with the given eigenvectors.

The minimization problem is

$$\min_{\lambda_k} \|\mathbf{L}\|_0 \text{ subject to } \mathbf{L} = \sum_{k=0}^{N-1} \lambda_k \mathbf{u}_k \mathbf{u}_k^T .$$

The convex form of this minimization problem is

$$\min_{\lambda_k} \|\mathbf{L}\|_1 \text{ subject to } \mathbf{L} = \sum_{k=0}^{N-1} \lambda_k \mathbf{u}_k \mathbf{u}_k^T .$$

The convex norm-one based form can produce the same solution as the original norm-zero form if the Laplacian sparsity is low and satisfies some conditions (in the sense discussed within Sect. 4.2).

Since the eigenvectors are obtained from the correlation matrix decomposition, the spectral analysis obtained in this way is related to the principal value decomposition (PVD), where the signal is decomposed onto the set of correlation matrix eigenvectors.

For the examples with classical signal processing, we have used the Fourier analysis. The problem formulation presented in this section can be used to define a graph such that the spectral analysis on this graph leads to some other well known transforms. We will illustrate this method on the Hadamard transform with $N = 8$ with eigenvectors

$$\mathbf{U} = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}.$$

If the eigenvalues are found to minimize the number of nonzero elements in the Laplacian, we get the graphs for $N = 8$ and $N = 16$ as shown in Fig. 39.

7.8 Physically Well Defined Graphs

7.8.1 Resistive Electrical Circuits

One of the oldest applications of graph theory in engineering was in electrical circuits theory. Graph theory based methods for analysis and transformations of electrical

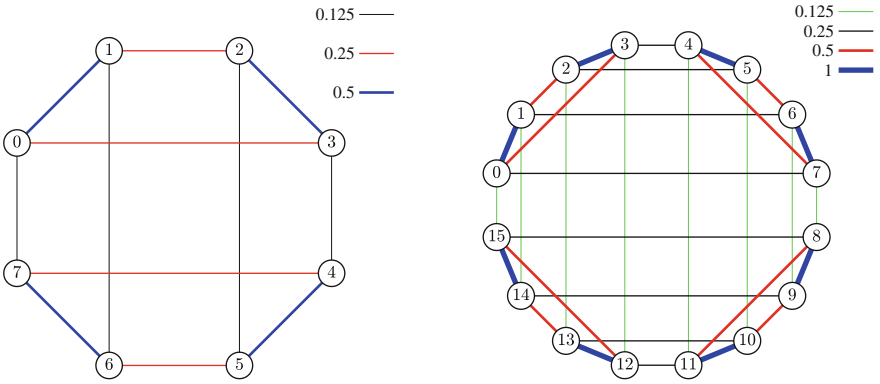


Fig. 39 Graph whose eigenvectors are the Hadamard transform basis functions for $N = 8$ and $N = 16$

circuits are already part of the classical electrical circuits courses and textbooks. This approach can directly be applied to other engineering fields, like heat transfer or mechanical mass strings. It is interesting that some general information theory problems can be interpreted and solved within the graph approach to the basic electrical circuits framework. In these cases the underlying graph topology is well defined and is a part of the problem statement.

The Laplacian can also be considered within the basic electric circuit theory. Since it can be derived based on the Kirchhoff's laws, the Laplacian is also known as the Kirchhoff matrix in electric circuit theory.

Consider a resistive electric circuit. Denote the electric potential in the circuit vertices (nodes) by $x(n)$. The vertices in an electrical circuit are connected with edges. The weight of an edge connecting the vertices n and m is defined by the edge conductance W_{nm}. The conductances are the reciprocal values to the edge resistances $W_{nm} = 1/R_{nm}$. The current in the edge from vertex n to vertex m is equal to

$$i_{nm} = \frac{x(n) - x(m)}{R_{nm}} = W_{nm}(x(n) - x(m)).$$

In addition to the edge currents, an external current generator is attached to each vertex. It can be considered as a source of the signal change in the vertices. The external current at a vertex n is denoted by i_n.

The sum of all currents going from a vertex n, $n = 0, 1, \ldots, N-1$, must be 0,

$$-i_n + \sum_m i_{nm} = 0.$$

The current of the external generator in vertex n must be equal to the sum of all edge currents going from this vertex,

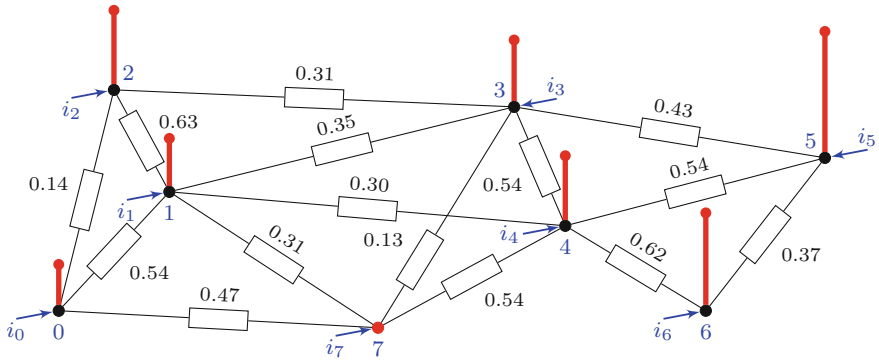


Fig. 40 Electric potential $x(n)$ as a signal on an electric circuit graph

$$i_n = \sum_m W_{nm}(x(n) - x(m)) = d_n x(n) - \sum_m W_{nm} x(m),$$
$$n = 0, 1, \ldots, N-1,$$

where

$$d_n = \sum_m W_{nm} = \sum_{m=0}^{N-1} W_{nm}$$

is the degree of vertex n. The summation over m can be extended to all vertices $m = 0, 1, \ldots, N-1$ since $W_{nm} = 0$ if there is not an edge between vertices n and m.

The previous equations can be written in a matrix form as

$$\mathbf{i} = \mathbf{D}\mathbf{x} - \mathbf{W}\mathbf{x}$$

or

$$\mathbf{L}\mathbf{x} = \mathbf{i}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the Laplacian of graph.

If the Laplacian matrix is decomposed as $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ we get $\boldsymbol{\Lambda}\mathbf{U}^T\mathbf{x} = \mathbf{U}^T\mathbf{i}$ or

$$\boldsymbol{\Lambda}\mathbf{X} = \mathbf{I}$$

where $\mathbf{X} = \mathbf{U}^T\mathbf{x}$ and $\mathbf{I} = \mathbf{U}^T\mathbf{i}$ are GDFT of graph signals $\mathbf{x}$ and $\mathbf{i}$.

Components of the spectral transform vector $\mathbf{X}$ are such that

$$\lambda_k X(k) = I(k)$$

for each k.

Signal on an electrical circuit graph can be related to the presented theory in several ways. For example, potentials on all vertices could be measured with some measurement noise. In that case, filtering on a graph should be applied. Another possible case is when the external conditions are imposed, for example sources are applied to some vertices. We are then interested in potential values at all vertices. This problem corresponds to the graph signal reconstruction.

7.8.2 Heat Transfer

The same model as in the resistive electrical circuit case can be used for a heat transfer network. In this case the signal values are the measured temperatures $x(n) = T(n)$. The heat flux is defined as

$$q_{nm} = (T(n) - T(m))C_{nm} = W_{nm}(x(n) - x(m)),$$

where C_{nm} are the heat transfer constants, representing edge weights in the underlying graph. Then the input heat flux in the vertex n is

$$q_n = \sum_m W_{nm}(x(n) - x(m)) = d_n x(n) - \sum_{m=0}^{N-1} W_{nm} x(m),$$

with

$$\mathbf{q} = \mathbf{L}\mathbf{x}$$

Active vertices are those where there is an external heat flux, while the passive vertices are those where all heat flux coming to a vertex is forwarded to other vertices through the edges. An example of a heat transfer graph is given in Fig. 41.

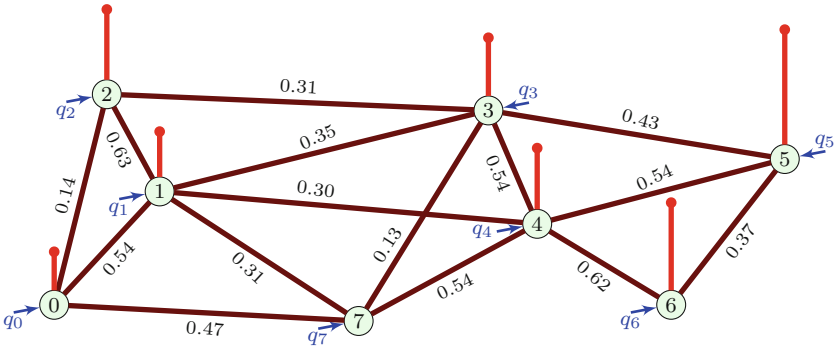


Fig. 41 Temperature $x(n) = T(n)$ as a signal on a heat transfer graph

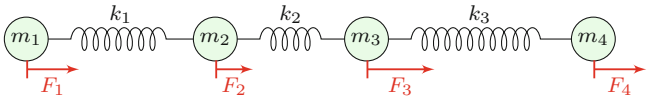


Fig. 42 Spring-mass system

7.8.3 Spring-Mass Systems

A spring mass system is used as a model of a graph and graph signal simulations. Consider a system of $N = 4$ masses corresponding to the line graph, Fig. 42. Assume that all displacements and forces are in the direction of the system line. The displacements $x(n)$ and the forces F_n, according to Hook's law in a steady state, are related as

$$\begin{aligned} k_1(x(1) - x(2)) &= F_1 \\ k_1(x(2) - x(1)) + k_2(x(2) - x(3)) &= F_2 \\ k_2(x(3) - x(2)) + k_3(x(3) - x(4)) &= F_3 \\ k_3(x(4) - x(3)) &= F_4 \end{aligned}$$

In matrix form

$$\begin{bmatrix} k_1 & -k_1 & 0 & 0 \\ -k_1 & k_1 + k_2 & -k_2 & \\ 0 & -k_2 & k_2 + k_3 & -k_3 \\ 0 & 0 & -k_3 & k_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}$$

$$\mathbf{Lx} = \mathbf{F}$$

These equations define a weighted graph and its corresponding Laplacian.

The Laplacian is singular matrix. In order to solve this system for unknown displacements (graph signal) we should introduce a reference vertex with a fixed position (zero displacement). Then the system $\mathbf{Lx} = \mathbf{F}$ can be solved.

7.8.4 Social Networks and Linked Pages

Social networks are also examples of well defined graphs. The vertices are network members and the edges define their relationships in a network. If two members are related, corresponding edge weight is 1. In this case weight matrix is equal to the adjacency matrix. An example of a simple social network with $N = 8$ members is shown in Fig. 43.

Pages with hyper-links can also be considered as a well defined directed graph. An example of links between $N = 8$ pages is given in Fig. 44. An interesting parameter for this kind of graphs is the PageRank.

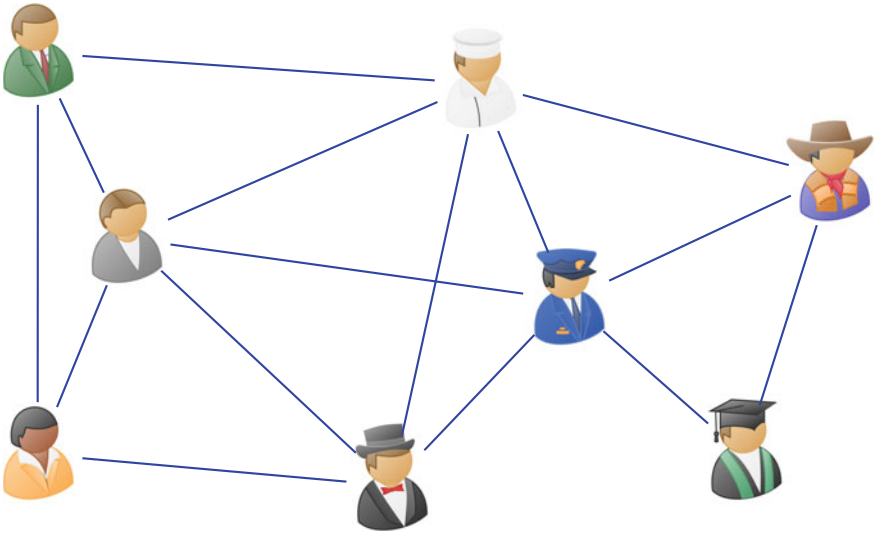


Fig. 43 Social network graph example

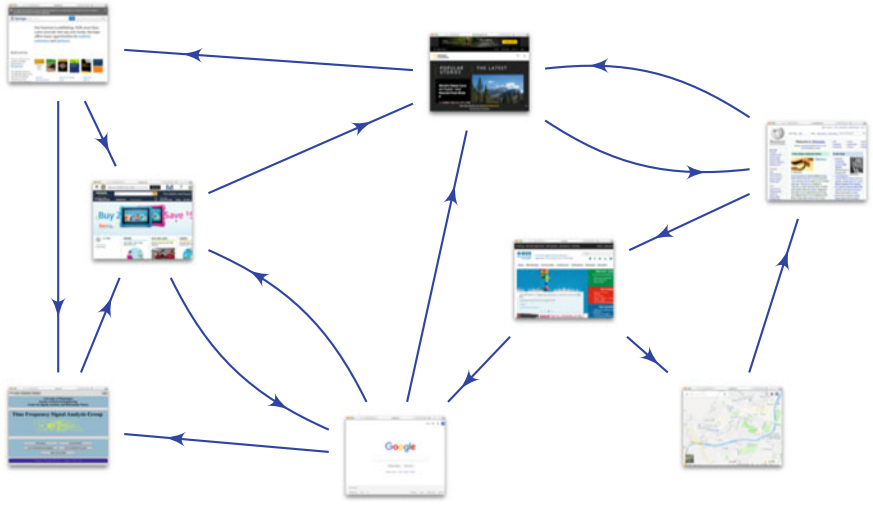


Fig. 44 Linked pages graph example

7.8.5 PageRank

For a directed graph, PageRank of vertex n is defined as a graph signal satisfying the relation

$$x(n) = \sum_m \frac{1}{d_m} W_{mn} x(m),$$

where W_{mn} are weights of the directed edges connecting the vertex m to vertex n and d_m is the outgoing degree of the vertex m. This means that the PageRank of each vertex is related to the PageRank of connected vertices.

The PageRank is commonly calculated using an iterative procedure defined by

$$x_{k+1}(n) = \sum_m \frac{1}{d_m} W_{mn} x_k(m),$$

starting from arbitrary page ranks, for example $x_0(n) = 1$.

The PageRank was defined by Google to rank the web pages. In the original definition a scaling factor was added,

$$x_{k+1}(n) = 0.15 + 0.85 \sum_m \frac{1}{d_m} W_{mn} x_k(m),$$

As an example, consider the graph from Fig. 44 (it is the same graph as in Fig. 2b). We will calculate the PageRank for all vertices in this graph. The weight/adjacency matrix of this graph $\mathbf{W} = \mathbf{A}$ is given by (1), right. The outgoing vertex degrees are calculated as the sum of the matrix columns, $d_m = \sum_{n=0}^{7} A_{nm}$. Their values are $\mathbf{d} = [2\ 3\ 1\ 3\ 1\ 2\ 1\ 2]$. Now the PageRank values for vertices can be obtained through the iterative procedure starting with initial page ranks $\mathbf{x}_0 = [1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1]$. The results for PageRank in a few iterations are

$$\begin{bmatrix} \mathbf{x}_0^T \\ \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_5^T \\ \vdots \\ \mathbf{x}_{11}^T \end{bmatrix} = \begin{bmatrix} 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 & 1.00 \\ 0.83 & 1.83 & 0.50 & 1.33 & 0.50 & 1.50 & 0.50 & 1.00 \\ 0.58 & 1.42 & 0.67 & 2.00 & 0.75 & 1.67 & 0.25 & 1.67 \\ \vdots & & & & & & & \\ 0.73 & 1.60 & 0.82 & 1.71 & 0.61 & 1.11 & 0.31 & 1.10 \\ \vdots & & & & & & & \\ 0.79 & 1.57 & 0.86 & 1.71 & 0.57 & 1.14 & 0.29 & 1.07 \end{bmatrix}.$$

The matrix form of the iterations is

$$\mathbf{x}_{k+1} = \mathbf{W}_N \mathbf{x}_k,$$

where $\mathbf{W}_N$ is obtained from $\mathbf{W}$ by dividing all elements of the mth column, $m = 0, 1, \ldots, N-1$, by d_m. The mean-values of matrix $\mathbf{W}_N$ columns are normalized.

In the considered example, the normalized adjacency/weighing matrix is

$$\mathbf{W}_N = \begin{bmatrix} 0 & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{3} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The final, stationary state, page rank can be obtained from

$$\mathbf{x} = \mathbf{W}_N \mathbf{x}.$$

The final PageRank $\mathbf{x}$ is the eigenvector of matrix $\mathbf{W}_N$ corresponding to the eigenvalue equal to 1.

For the considered example, the eigenvalue decomposition of the matrix $\mathbf{W}_N$ results in the eigenvector, corresponding to eigenvalue 1,

$$\mathbf{x}^T = [0.79 \;\; 1.57 \;\; 0.86 \;\; 1.71 \;\; 0.57 \;\; 1.14 \;\; 0.29 \;\; 1.07].$$

The eigenvector is normalized with its mean value. It corresponds to the iterative solution obtained after 11 iterations.

7.8.6 Random Walk

Assume that the signal $x(n)$ represents probabilities that a random walker is present in vertex n. The random walker will transit from vertex n to one of its neighbouring vertices m with probabilities

$$p_{nm} = \frac{W_{nm}}{\sum_m W_{nm}},$$

where W_{nm} are affinities of the walker to transit from vertex n to vertex m.

The signal $x(n)$ calculation can be considered within the graph framework where W_{nm} are edge weights.

The probabilities in the stage $(p+1)$ are calculated starting from probabilities in the previous stage as

$$\mathbf{x}_{p+1} = \mathbf{D}^{-1}\mathbf{W}\mathbf{x}_p$$

or $\mathbf{D}\mathbf{x}_{p+1} = \mathbf{W}\mathbf{x}_p$. Matrix $\mathbf{W}$ is a matrix of weighting coefficients and $\mathbf{D}$ is the degree matrix.

In stationary state, when $\mathbf{x}_{p+1} = \mathbf{x}_p = \mathbf{x}$ we have $\mathbf{D}\mathbf{x} = \mathbf{W}\mathbf{x}$ or

$$\mathbf{L}\mathbf{x} = \mathbf{0}.$$

Various problem formulations and solutions are now possible within the presented graph theory framework. For example, if we want to find probabilities that the walker reaches vertex 5 before he reaches vertex 7 from any vertex n we have to solve the system $\mathbf{Lx} = \mathbf{0}$ with $x(5) = 1$ and $x(7) = 0$.

7.9 Gaussian Random Signal

Consider a random graph signal $x(n)$ and assume that each sample is Gaussian distributed with mean μ_n and standard deviation σ_n. Assuming that the signal values are correlated, the pdf function of the signal $\mathbf{x}$ is

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^N}} \det(\mathbf{\Sigma}_x^{-1}) \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}_x^{-1}(\mathbf{x}-\boldsymbol{\mu})\right).$$

The inverse value of the autocovariance matrix is the precision matrix $\mathbf{\Theta} = \mathbf{\Sigma}_x^{-1}$. The name precision comes from the one-dimensional case in which the precision is inversely proportional to the variance $\Theta = 1/\sigma^2$.

The maximum likelihood estimate of $\mathbf{x}$ is obtained by minimizing

$$E_x = \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}_x^{-1}(\mathbf{x}-\boldsymbol{\mu}).$$

Its solution is

$$\mathbf{\Sigma}_x^{-1}(\mathbf{x}-\boldsymbol{\mu}) = 0.$$

For zero mean random signal, $\boldsymbol{\mu} = \mathbf{0}$ and $\mathbf{\Sigma}_x^{-1}\mathbf{x} = 0$. This corresponds to the energy of change minimization (maximal smoothness) in the graph.

The Laplacian corresponding to the information matrix is defined by

$$\mathbf{\Sigma}_x^{-1} = \mathbf{L} + \mathbf{P}$$

where $\mathbf{P}$ is a diagonal matrix such that sum of the Laplacian columns is zero. Note that some of non-diagonal elements of $\mathbf{\Sigma}_x^{-1}$ can be positive. In that case, additional conditions should be added to find the best solution avoiding positive coefficients on the Laplacian diagonal.

The edge weights can be extracted from the Laplacian matrix. Since the Laplacian is defined using signal values, this is a point when the presented analysis meets the discussion from the previous section.

8 Appendix

8.1 Graph Signal Calculation Using Laplacian

The graph signal $\mathbf{x}$ can be calculated using this system of linear equations with the vector of external sources. Since the Laplacian is a singular matrix, one graph signal value (potential) should be considered as a free/referent variable.

In the case when there is no external generator, the graph signal $\mathbf{x}$ satisfies the following equation

$$\mathbf{Lx} = \mathbf{0}.$$

This equation corresponds to the minimum of the energy of change in $\mathbf{x}$ defined by the quadratic form

$$E_x = \mathbf{x}^T\mathbf{Lx} = \frac{1}{2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} W_{nm}(x(n)-x(m))^2 = \frac{1}{2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} P_{nm}.$$

where P_{nm} can be considered as a power dissipated in the edge between vertices n and m. It follows from $\partial E_x/\partial \mathbf{x}^T = 2\mathbf{Lx} = \mathbf{0}$ and (38). The solution of this equation $x(n)$ is a constant defined by the signal value at the reference vertex.

For nontrivial solutions, there should be an external source in at least two vertices. Assume that one of them is chosen as the referent vertex. Signal or external source values at these vertices are sufficient to find signal values at all other vertices.

As an example, consider the graph and signal sensed on the graph presented in Fig. 40. The signal values are

$$\mathbf{x} = [0.57, 0.67, 1.03, 0.86, 0.90, 1.68, 1.29, 0]^T$$

and the Laplacian of the signal is

$$\mathbf{Lx} = [0, 0, 1, 0, 0, 2, 0, -3]^T.$$

This means that the vertices denoted by 0, 1, 3, 4, 6 are not active in this case. Their values can be obtained as linear combinations of the signal at neighboring active vertices:

$$\begin{aligned}
-0.47x(7) - 0.14x(2) - 0.54x(1) + 1.15x(0) &= 0\\
-0.31x(7) - 0.30x(4) - 0.35x(3) - 0.63x(2) + 2.13x(1) - 0.54x(0) &= 0\\
-0.43x(5) - 0.54x(4) + 1.82x(3) - 0.31x(2) - 0.54x(1) &= 0\\
-0.62x(6) - 0.54x(5) + 2.00x(4) - 0.54x(3) - 0.30x(1) &= 0\\
0.99x(6) - 0.37x(5) - 0.62x(4) &= 0
\end{aligned}$$

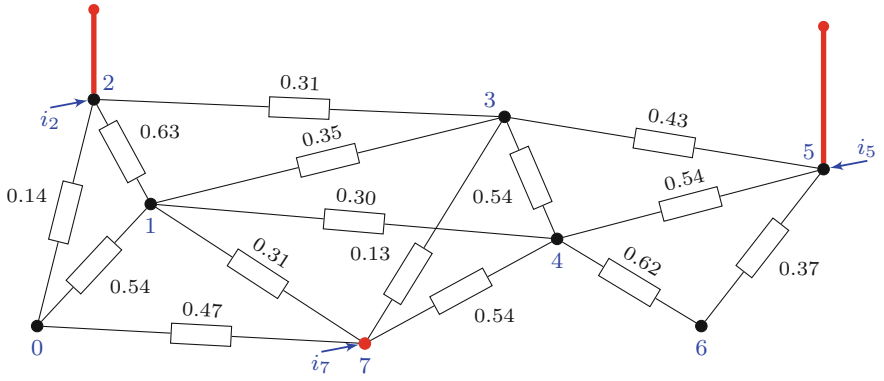


Fig. 45 Electric potential $x(n)$ as a signal on an electric circuit graph at three vertices with nonzero external sources

Solving this system with known signal values $x(2) = 1.03$, $x(5) = 1.68$, and $x(7) = 0$ at the active vertices, Fig. 45, we get the remaining signal values

$$\mathbf{x}_p = [x(0), x(1), x(3), x(4), x(6)]^T = [0.57, 0.67, 0.86, 0.90, 1.29]^T.$$

Assume that only some of the vertices are active, with external sources. Denote the set of these M active vertices by $\mathcal{E}$. Then the vertices without external sources are $\mathcal{H}$, such that $\mathcal{V} = \mathcal{E} \cup \mathcal{H}$ and $\mathcal{E} \cap \mathcal{H} = \emptyset$. For a full reconstruction of this signal we have to know only M signal values $x(n)$ at vertices $n \in \mathcal{E}$ or any other linearly independent M graph signal samples. The remaining $N - M$ graph signal samples are obtained from the equations following from the fact that the Laplacian at $n \in \mathcal{H}$ is zero-valued.

From the circuit theory, it is well known that this kind of graph can be downscaled without any influence to the signal at vertices where Laplacian is nonzero. The vertices with zero Laplacian can be omitted by using so called Y-Δ (star-mesh) transformation. The resulting graph has a reduced number of vertices, while the number of edges may be increased.

This kind of interpolation can be applied to classic, time-domain signals as well. The Laplacian of a time-domain signal at instant (vertex) n is calculated as $2x(n) - x(n-1) - x(n+1)$. Its zero value will indicate that this vertex (instant) is not active (there is no external source). Therefore, this value can be omitted since it can be obtained from the condition that Laplacian is zero from other signal values at instants $n - 1$ and $n + 1$. An illustration of such a signal and reconstruction based on the signal values at the Laplacian nonzero positions is presented in Fig. 46.

In this way, vertices where the signal behavior is changed are detected using the Laplacian. Note that the two-dimensional Laplacian is a classical tool in the image edge detection.

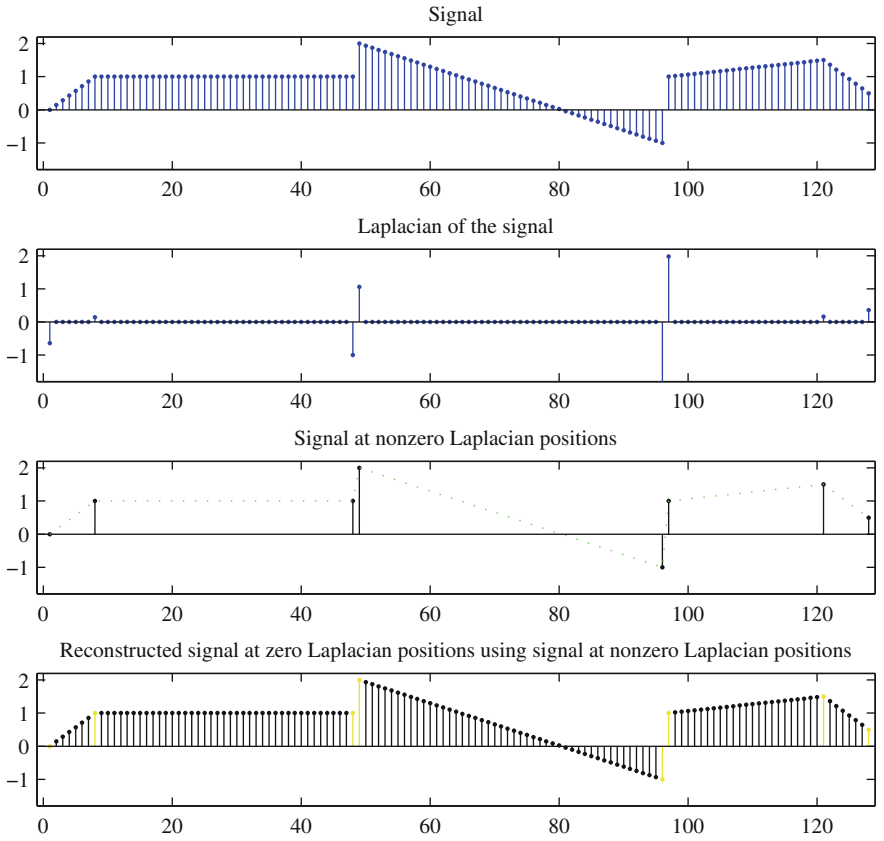


Fig. 46 Reconstructed signal at zero Laplacian positions using signal at nonzero Laplacian positions. Signals are presented as functions of the vertex index n

This kind of subsampling can be considered as a signal processing and graph signal processing equivalence of the Laplace differential equation:

$$\mathbf{L}\{V(x,y)\} = \nabla^2 V(x,y) = \frac{\partial^2 V(x,y)}{\partial x^2} + \frac{\partial^2 V(x,y)}{\partial y^2} = 0$$

with given boundary conditions (Dirichlet problem):

$$V(x,y) = f(x,y) \text{ on boundary } D(x,y) = 0.$$

In graph signal processing, vertices with a nonzero Laplacian define the boundary set and the signal values on these vertices are the boundary conditions.

In general, when the Laplacian of a graph signal is nonzero at all vertices we can apply hard thresholding, keeping large Laplacian values and neglecting the small

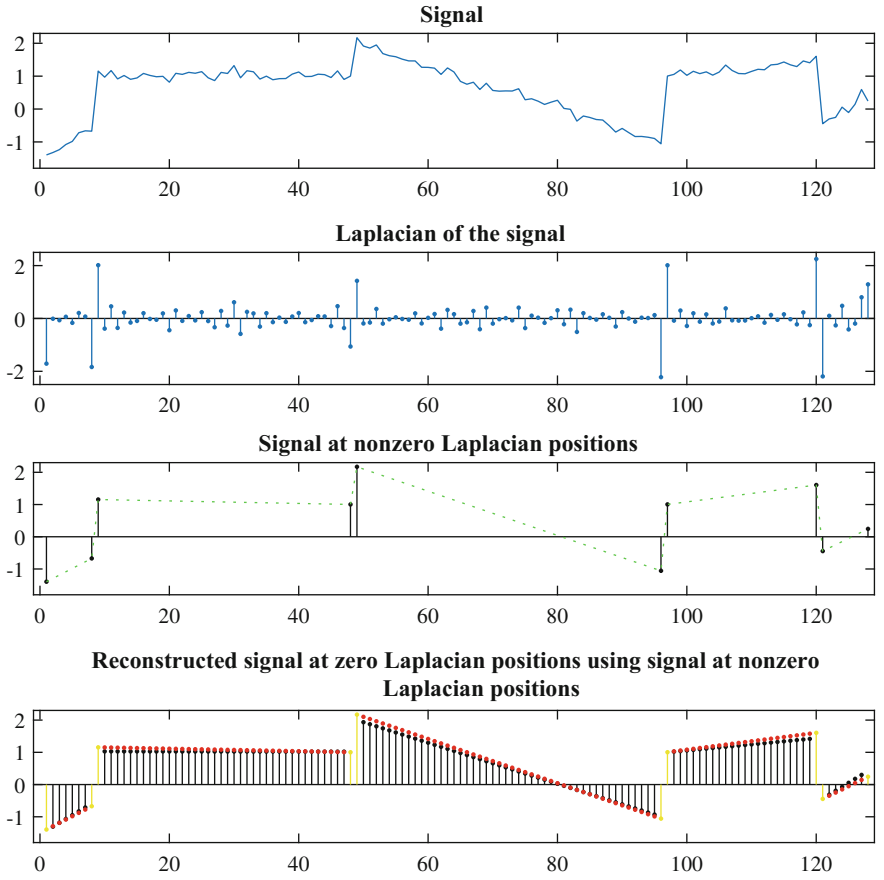


Fig. 47 Reconstructed signal at small Laplacian positions using signal at nonzero Laplacian positions and linear approximation

ones in a linear approximation of the graph signal. In this way we can keep just the signal values on the vertices

$$n \in \mathcal{E} \text{ if } |\mathbf{Lx}| \geq \tau \text{ at vertex } n.$$

The correction (high pass part) of the signal is equal to the Laplacian of signal on $n \notin \mathcal{E}$. It can be used to adjust the values of linear approximation to the full signal reconstruction. If a graph signal is noisy then the Laplacian can be used for the detection of the active (boundary condition) vertices. The signal values can be obtained by a mean squared linear approximation of the noisy data with the detected positions of the discontinuity of linear behavior Fig. 47.

Some vertices may be active for some signal realizations and not active for other signal realizations. Some vertices may not be active for the whole set of realizations, meaning that the graph can be downscaled not only for one considered signal $x(n)$ but for a class of signals by omitting some of the vertices using star-mesh conversions.

8.2 Random Signal Simulation on a Graph

The presentation of a graph and graph signal within the circuit theory can be used to simulate random signals on graphs. Several approaches are possible. Here we will present few of them.

(1) Assume that the graph is initiated by external sources that are random variables. In that case the pth observation of a random signal on the graph is simulated as a solution of the system of equations

$$\mathbf{L}\mathbf{x}_p = \boldsymbol{\varepsilon}_p$$

using $\mathbf{i}_p = \boldsymbol{\varepsilon}_p$. One of the external sources (randomly chosen for each observation p) should compensate all other sources, to ensure $\sum_{n=0}^{N-1} \varepsilon_p(n) = 0$.

(2) Assume that the graph is initiated at only one of its vertices (and the reference vertex) with random white external zero-mean white source. The position of these vertices is randomly selected for each p. Then the random signal observation on a graph is obtained as a solution of

$$\mathbf{L}\mathbf{x}_p = \mathbf{i}_p$$

where $i_p(n) = \varepsilon_p \delta(n - n_i) - \varepsilon_p \delta(n - n_j)$ and n_i and n_j are two randomly selected vertices in each observation.

(3) A simple random graph signal can be simulated using its values at two randomly positioned vertices and $\mathbf{L}\mathbf{x}_p = 0$. Assuming that $x_p(n) = \varepsilon_p \delta(n - n_i) + \epsilon_p \delta(n - n_j)$ and n_i and n_j are two randomly selected vertices in each observation, we may solve the system for all other signal samples using

$$\mathbf{L}\mathbf{x}_p = \mathbf{0}.$$

With two assumed values $x_p(n)$ at $n = n_i$ and $n = n_j$ we can solve this system for all other signal values. In the case of external sources the values should be compensated. In this case there is no need for compensation, meaning that ε_p and ϵ_p could be independent random variables.

(4) Assume that the signal on a graph is formed using a linear combination of a white noise $\mathbf{x}_p^{(0)} = \boldsymbol{\varepsilon}_p$ and its graph shifted versions. The first iteration is

$$\mathbf{x}_p^{(1)} = \alpha_1 \mathbf{L}\mathbf{x}_p^{(0)} + \mathbf{x}_p^{(0)}.$$

After M iterations we get

$$\mathbf{x}_p^{(M)} = \alpha_M \mathbf{L}\mathbf{x}_p^{(M-1)} + \mathbf{x}_p^{(0)} = (h_M \mathbf{L}^M + h_{M-1}\mathbf{L}^{M-1} + \cdots + h_1 \mathbf{L}^1 + 1)\boldsymbol{\varepsilon}_p,$$

where $h_m = \alpha_m \alpha_{m-1} \ldots \alpha_1$. The final signal

$$\mathbf{x}_p = \mathbf{x}_p^{(M)} = H(\mathbf{L})\boldsymbol{\varepsilon}_p,$$

is a GWSS signal.

(5) In graph signal simulation we may also use the adjacency matrix and graph shifts. Assume that an undirected graph with adjacency matrix $\mathbf{A}$ is initiated at N_a randomly chosen vertices $n_1, n_2, \ldots, n_{N_a}$, $\eta = N_a/N$, with spikes $\delta(n - n_i)$, $i = 1, 2, \ldots, N_a$. If we shift these spikes K times we get

$$\mathbf{x} = \mathbf{A}^K \sum_{i=1}^{N_a} \delta(n - n_i).$$

Parameters K and N_a define the resulting signal smoothness. An example of one realization of such a signal is presented in Fig. 22 for $\eta = 1/8$, $K = 1$ (upper subplots) and $\eta = 2/8$, $K = 1$ (lower subplots) using spikes $a_i \delta(n - n_i)$, where a_i are the spike amplitudes.

(6) In classical Fourier analysis, the signals are commonly simulated as sums of the harmonic basis functions. This kind of simulation may be used in graph signal processing as well. A signal on a graph can be written as

$$\mathbf{x} = \sum_{i=1}^{K} a_{k_i} \mathbf{u}_{k_i}$$

where $\mathbf{u}_k$ are the Laplacian or adjacency matrix eigenvectors, and a_k are random constants. This kind of graph signal simulation, with or without an additive noise, is often used in this chapter.

8.3 From the Newton Minimization to the Graphical LASSO

8.3.1 Newton Method

Fist we will shortly review the Newton iterative algorithm for finding the minimum of a convex function. Consider a function $f(x)$ and assume that it is differentiable. Denote the minimum position of $f(x)$ by x^*. The first derivative of $f(x)$ at $x^* = x + \Delta x$ can be expanded into a Taylor series around a point x, using the linear approximation, as

$$f'(x^*) = f'(x) + f''(x)\Delta x. \tag{70}$$

Since $f'(x^*) = 0$ by definition, with $\Delta x = x^* - x$, relation (70) can be rewritten as

$$x^* - x = -\frac{f'(x)}{f''(x)}.$$

This form is used to define an iterative procedure (Newton's iterative method) for finding x^* starting from an $x = x_0$ as

$$x_{k+1} = x_k - \alpha f'(x_k).$$

Parameter α is commonly used instead of $1/f''(x)$ to control the iteration step. Its value should be $0 < \alpha \leq \max(|1/f''(x)|)$, for the considered interval of x. This is the form of the well-known steepest descend method for convex function minimization.

The value $x^* = x - \alpha f'(x)$, with $\alpha = 1/f''(x)$ would be obtained as result of the minimization of a cost function defined by the quadratic form

$$x^* = \arg\min_z G(z) = \arg\min_z \left(f(x) + f'(x)(z - x) + \frac{1}{2\alpha}(z - x)^2 \right).$$

From $d(f(x) + f'(x)(z - x) + \frac{1}{2\alpha}(z - x)^2)/dz = 0$ we would get $x^* = z$.

Next assume that we want to minimize the cost function

$$J(x) = \frac{1}{2\alpha}(x - y)^2 + \lambda|x|,$$

where λ is a parameter. From $dJ(x)/dx = (x - y)/\alpha + \lambda \text{sign}(x) = 0$ we get

$$x + \lambda\alpha \text{sign}(x) = y.$$

The soft-thresholding is used as a solution of this equation. It is denoted as $\text{soft}(y, \alpha\lambda)$ and defined by

$$x = \text{soft}(y, \alpha\lambda) = \begin{cases} y + \alpha\lambda \text{ for } y < -\alpha\lambda \\ 0 \quad \text{for } |y| \leq \alpha\lambda \\ y - \alpha\lambda \text{ for } y > \alpha\lambda. \end{cases}$$

8.3.2 LASSO

For the lasso minimization we will consider the cost function

$$J(\mathbf{X}) = \|\mathbf{y} - \mathbf{A}\mathbf{X}\|_2^2 + \lambda\|\mathbf{X}\|_1 = \|\mathbf{y}\|_2^2 - 2\mathbf{X}^T\mathbf{A}^T\mathbf{y} + \mathbf{X}^T\mathbf{A}^T\mathbf{A}\mathbf{X} + \lambda\|\mathbf{X}\|_1,$$

where $\mathbf{y}$ is a $M \times 1$ column vector, $\mathbf{X}$ is a $N \times 1$ column vector, and $\mathbf{A}$ is an $M \times N$ matrix.

Minimization of this cost function with respect to $\mathbf{X}$ will produce its value such that $\mathbf{AX}$ is as close to $\mathbf{y}$ as possible, promoting the sparsity of $\mathbf{X}$ at the same time. Balance between these two requirements is defined by the parameter λ.

Consider first the differentiable part of the cost function $J(\mathbf{X})$ denoted by $J_D(\mathbf{X}) = \|\mathbf{y} - \mathbf{AX}\|_2^2 = (\mathbf{y} - \mathbf{AX})^T(\mathbf{y} - \mathbf{AX})$. Its derivatives are

$$\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T} = -2\mathbf{A}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{A}^T\mathbf{A}$$

and

$$\frac{\partial^2 J_D(\mathbf{X})}{(\partial \mathbf{X}^T)^2} = 2\mathbf{A}^T\mathbf{A}.$$

The linear model for the first derivative of $J_D(\mathbf{X})$ around its minimum is

$$\frac{\partial J_D(\mathbf{X}^*)}{\partial \mathbf{X}^T} = \frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T} + (\Delta\mathbf{X})\frac{\partial^2 J_D(\mathbf{X})}{(\partial \mathbf{X}^T)^2}.$$

By replacing the inverse of the second order derivative by a constant diagonal matrix $\alpha\mathbf{I}$ we get

$$\Delta\mathbf{X} = \mathbf{X}^* - \mathbf{X} = -\alpha\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T}$$

or

$$\mathbf{X}^* = \mathbf{X} - \alpha\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T} \tag{71}$$

with $0 < \alpha < 1/\max\|2\mathbf{A}^T\mathbf{A}\| = 1/(2\lambda_{\max})$, where $\lambda_{\max}$ is the maximal eigenvalue of matrix $\mathbf{A}^T\mathbf{A}$.

In order to find $\mathbf{Z} = \mathbf{X}^*$ that minimizes the complete cost function $J(\mathbf{X})$, we can minimize the squared difference $\mathbf{Z} - (\mathbf{X} - \alpha\mathbf{I}\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T})$ and the norm-one of $\mathbf{Z}$, by forming the cost function $G(\mathbf{Z})$ as

$$\mathbf{X}^* = \arg\min_{\mathbf{Z}} G(\mathbf{Z}) = \arg\min_{\mathbf{Z}}\left(\frac{1}{2\alpha}\|\mathbf{Z} - \left(\mathbf{X} - \alpha\mathbf{I}\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T}\right)\|^2 + \lambda\|\mathbf{Z}\|_1\right).$$

This minimization will produce $\mathbf{Z}$ as close as possible to the desired solution (71), minimizing its norm-one at the same time. The balance parameter is λ.

The solution of

$$\mathbf{X}^* = \arg\min_{\mathbf{Z}} G(\mathbf{Z}) = \frac{1}{2\alpha}\|\mathbf{Z} - \mathbf{Y}\|^2 + \lambda\|\mathbf{Z}\|_1$$

is obtained from

$$\frac{1}{\alpha}(\mathbf{X}^* - \mathbf{Y}) + \lambda \text{sign}(\mathbf{X}^*) = 0.$$

Using the soft function we can write

$$\mathbf{X}^* = \text{soft}(\mathbf{Y}, \alpha\lambda).$$

Next we will replace the value of $\mathbf{Y}$ by

$$\begin{aligned}\mathbf{Y} = \left(\mathbf{X} - \alpha\mathbf{I}\frac{\partial J_D(\mathbf{X})}{\partial \mathbf{X}^T}\right) &= \mathbf{X} - \alpha\mathbf{I}(-2\mathbf{A}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{A}^T\mathbf{A}) \\ &= 2\alpha\mathbf{A}^T\mathbf{y} + (\mathbf{I} - 2\alpha\mathbf{A}^T\mathbf{A})\mathbf{X}.\end{aligned}$$

The iterative formula for the solution of the defined minimization problem is obtained by replacing $\mathbf{X}^* = \mathbf{X}_{k+1}$ and $\mathbf{X} = \mathbf{X}_k$ as

$$\mathbf{X}_{k+1} = \text{soft}(2\alpha\mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{X}_k) + \mathbf{X}_k, \alpha\lambda).$$

This formula can easily be written for each element of $\mathbf{X}_k$. This is the LASSO (Least Absolute Shrinkage and Selection Operator) iterative algorithm. As the initial estimate $\mathbf{X}_0 = \mathbf{A}^T\mathbf{y}$ is commonly used.

8.3.3 Graphical LASSO

In graph model learning, the cost function in the form

$$J(\mathbf{Q}) = -\log\det\mathbf{Q} + \text{Trace}(\mathbf{Q}\mathbf{R}_x) + \|\mathbf{Q}\|_1$$

may be obtained. Here $\mathbf{Q}$ is the generalized Laplacian $N \times N$ matrix, while $\mathbf{R}_x$ is the available $N \times N$ correlation matrix. The meaning of these terms is explained within the main part of this chapter.

The derivative of the cost function with respect to the elements of $\mathbf{Q}$ can be written as

$$-\mathbf{Q}^{-1} + \mathbf{R}_x + \text{sign}(\mathbf{Q}) = 0 \tag{72}$$

at $\partial J(\mathbf{Q})/\partial\mathbf{Q} = 0$.

Note that $\log\det\mathbf{Q} = \sum_{i=0}^{N-1}\log\lambda_i = \text{Trace}(\log\mathbf{\Lambda}) = \text{Trace}(\log\mathbf{Q})$, where λ_i are the eigenvalues of $\mathbf{Q}$.

Introducing the notation $\mathbf{W} = \mathbf{Q}^{-1}$ or

$$\mathbf{W}\mathbf{Q} = \mathbf{I}$$

we can write

$$\begin{bmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}_{12}^T & w_{22} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_{11} & \mathbf{q}_{12} \\ \mathbf{q}_{12}^T & q_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{73}$$

Multiplying the first row block of $\mathbf{W}$ with the last column block of $\mathbf{Q}$ we get

$$\mathbf{W}_{11}\mathbf{q}_{12} + \mathbf{w}_{12}q_{22} = \mathbf{0}$$

This means that

$$\mathbf{w}_{12} = -\mathbf{W}_{11}\mathbf{q}_{12}/q_{22} = \mathbf{W}_{11}\boldsymbol{\beta}$$

where

$$\boldsymbol{\beta} = -\mathbf{q}_{12}/q_{22}$$

is normalized with $q_{22} > 0$.

Now, from the derivative equation (72) we may write

$$-\begin{bmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}_{12}^T & w_{22} \end{bmatrix} + \begin{bmatrix} \mathbf{R}_{11} & \mathbf{r}_{12} \\ \mathbf{r}_{12}^T & r_{22} \end{bmatrix} + \text{sign}\left(\begin{bmatrix} \mathbf{Q}_{11} & \mathbf{q}_{12} \\ \mathbf{q}_{12}^T & q_{22} \end{bmatrix}\right) = \mathbf{0}.$$

For the upper right block we can write

$$-\mathbf{w}_{12} + \mathbf{r}_{12} + \text{sign}(\mathbf{q}_{12}) = \mathbf{0}$$

or after replacing $\mathbf{w}_{12} = \mathbf{W}_{11}\boldsymbol{\beta}$ and $\mathbf{q}_{12} = -\boldsymbol{\beta}/q_{22}$ we get

$$-\mathbf{W}_{11}\boldsymbol{\beta} + \mathbf{r}_{12} - \text{sign}(\boldsymbol{\beta}) = \mathbf{0}.$$

The solution of this equation for $\boldsymbol{\beta}$ is already defined within LASSO consideration. It is

$$\beta_i = \text{soft}\left(r_{12}(i) - \sum_{k \neq i} W_{11}(k,i)\beta_k, \lambda\right) / W_{11}(i). \tag{74}$$

Now we may summarize the graphical LASSO (GLASSO) iterative algorithm as:

- In the initial step,

$$\mathbf{W} = \mathbf{R}_x + \lambda\mathbf{I}$$

  is used. For each coordinate $j = 1, 2, \ldots, N$, the matrix equation of form (73) is written. For each j the reduced matrix $\mathbf{W}_{11}$ is formed by omitting the jth row and the jth column. Then the matrix $\mathbf{R}_x$ is rearranged appropriately.
- Equation (74) is solved.
- The matrix $\mathbf{W}$ is updated for each j with

$$\mathbf{w}_{12} = \mathbf{W}_{11}\boldsymbol{\beta}.$$

- In the final iteration, for each j, the value of matrix $\mathbf{Q}$ is updated as

$$\mathbf{q}_{12} = -\boldsymbol{\beta}/q_{22}$$

with $1/q_{22} = w_{22} - \mathbf{w}_{12}^T - \boldsymbol{\beta}$.

9 Conclusion

An introduction to graph signal processing is presented. This chapter consists of three main parts. In the first part, a review of graphs is given. Next, the signal on graph definitions, basic properties, and systems for processing signals on graphs are reviewed. Finally, the graph topologies are discussed. The appendix provides some supplementary material for better understanding of the principles presented in the main part of the chapters. The topic of this book is the spectral localization through vertex-frequency analysis [39, 83–100], which will be presented in the next chapters.

Acknowledgements Ervin Sejdić acknowledges the support of the NSF CAREER grant number 1652203.

References

1. L.J. Grady, J.R. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science* (Springer Science & Business Media, New York, 2010)
2. S.S. Ray, *Graph Theory with Algorithms and Its Applications: in Applied Science and Technology* (Springer Science & Business Media, New York, 2012)
3. A. Marques, A. Ribeiro, S. Segarra, Graph signal processing: fundamentals and applications to diffusion processes, in *International Conference on Acoustics, Speech, and Signal Processing, (ICASSP), 2017* (IEEE, 2017)
4. H. Krim, A.B. Hamza, *Geometric Methods in Signal and Image Analysis* (Cambridge University Press, Cambridge, 2015)
5. A. Bunse-Gerstner, W.B. Gragg, Singular value decompositions of complex symmetric matrices. J. Comput. Appl. Math. **21**(1), 41–54 (1988)
6. D.S. Grebenkov, B.-T. Nguyen, Geometrical structure of Laplacian eigenfunctions. SIAM Rev. **55**(4), 601–667 (2013)
7. R. Bapat, The Laplacian matrix of a graph. Math. Stud.-India **65**(1), 214–223 (1996)
8. S. O'Rourke, V. Vu, K. Wang, Eigenvectors of random matrices: a survey. J. Comb. Theory Ser. A **144**, 361–442 (2016)
9. K. Fujiwara, Eigenvalues of Laplacians on a closed Riemannian manifold and its nets. Proc. Am. Math. Soc. **123**(8), 2585–2594 (1995)
10. S.U. Maheswari, B. Maheswari, Some properties of Cartesian product graphs of Cayley graphs with arithmetic graphs. Int. J. Comput. Appl. **138**(3) (2016)
11. D.M. Cvetković, M. Doob, H. Sachs, *Spectra of Graphs: Theory and Application*, vol. 87 (Academic, New York, 1980)

12. D.M. Cvetković, M. Doob, Developments in the theory of graph spectra. Linear Multilinear Algebra **18**(2), 153–181 (1985)
13. D.M. Cvetković, I. Gutman, Selected topics on applications of graph spectra. Matematicki Institut SANU (2011)
14. A.E. Brouwer, W.H. Haemers, *Spectra of Graphs* (Springer Science & Business Media, New York, 2011)
15. F. Chung, *Spectral Graph Theory* (AMS, Providence, 1997)
16. O. Jones, *Spectra of Simple Graphs*, vol. 13 (Whitman College, Walla Walla, 2013)
17. D. Mejia, O. Ruiz-Salguero, C.A. Cadavid, Spectral-based mesh segmentation. Int. J. Interact. Des. Manuf. (IJIDeM) **11**(3), 503–514 (2017)
18. H. Lu, Z. Fu, X. Shu, Non-negative and sparse spectral clustering. Pattern Recognit. **47**(1), 418–426 (2014)
19. X. Dong, P. Frossard, P. Vandergheynst, N. Nefedov, Clustering with multi-layer graphs: a spectral perspective. IEEE Trans. Signal Process. **60**(11), 5820–5831 (2012)
20. R. Horaud, A short tutorial on graph Laplacians, Laplacian embedding, and spectral clustering (2009)
21. R. Hamon, P. Borgnat, P. Flandrin, C. Robardet, Relabelling vertices according to the network structure by minimizing the cyclic bandwidth sum. J. Complex Netw. **4**(4), 534–560 (2016)
22. M. Masoumi, A.B. Hamza, Spectral shape classification: a deep learning approach. J. Vis. Commun. Image Represent. **43**, 198–211 (2017)
23. M. Masoumi, C. Li, A.B. Hamza, A spectral graph wavelet approach for nonrigid 3d shape retrieval. Pattern Recognit. Lett. **83**, 339–348 (2016)
24. J.M. Mouraaa, Graph signal processing, *Cooperative and Graph Signal Processing* (Elsevier, Amsterdam, 2018), pp. 239–259
25. M. Vetterli, J. Kovačević, V. Goyal, *Foundations of Signal Processing* (Cambridge University Press, Cambridge, 2014)
26. A. Sandryhaila, J.M. Moura, Discrete signal processing on graphs. IEEE Trans. Signal Process. **61**(7), 1644–1656 (2013)
27. V.N. Ekambaram, *Graph-Structured Data Viewed Through a Fourier Lens* (University of California, Berkeley, 2014)
28. A. Sandryhaila, J.M. Moura, Discrete signal processing on graphs: frequency analysis. IEEE Trans. Signal Process. **62**(12), 3042–3054 (2014)
29. A. Sandryhaila, J.M. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process. Mag. **31**(5), 80–90 (2014)
30. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
31. R. Hamon, P. Borgnat, P. Flandrin, C. Robardet, Extraction of temporal network structures from graph-based signals. IEEE Trans. Signal Inf. Process. Netw. **2**(2), 215–226 (2016)
32. S. Chen, A. Sandryhaila, J.M. Moura, J. Kovačević, Signal denoising on graphs via graph filtering, in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (IEEE, 2014), pp. 872–876
33. A. Gavili, X.-P. Zhang, On the shift operator, graph frequency, and optimal filtering in graph signal processing. IEEE Trans. Signal Process. **65**(23), 6303–6318 (2017)
34. A. Venkitaraman, S. Chatterjee, P. Händel, Hilbert transform, analytic signal, and modulation analysis for graph signal processing (2016), arXiv:1611.05269
35. A. Agaskar, Y.M. Lu, A spectral graph uncertainty principle. IEEE Trans. Inf. Theory **59**(7), 4338–4356 (2013)
36. X. Yan, B.M. Sadler, R.J. Drost, P.L. Yu, K. Lerman, Graph filters and the z-Laplacian. IEEE J. Sel. Top. Signal Process. **11**, 774–784 (2017)
37. X. Wang, J. Chen, Y. Gu, Local measurement and reconstruction for noisy bandlimited graph signals. Signal Process. **129**, 119–129 (2016)

38. S. Segarra, A. Ribeiro, Stability and continuity of centrality measures in weighted graphs. IEEE Trans. Signal Process. **64**(3), 543–555 (2016)
39. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**(2), 260–291 (2016)
40. S. Chen, A. Sandryhaila, J. Kovačević, Sampling theory for graph signals, in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2015), pp. 3392–3396
41. S. Chen, R. Varma, A. Sandryhaila, J. Kovačević, Discrete signal processing on graphs: sampling theory. IEEE Trans. Signal Process. **63**(24), 6510–6523 (2015)
42. S. Chen, A. Sandryhaila, J.M. Moura, J. Kovačević, Signal recovery on graphs: variation minimization. IEEE Trans. Signal Process. **63**(17), 4609–4624 (2015)
43. S. Chen, R. Varma, A. Singh, J. Kovačević, Signal recovery on graphs: fundamental limits of sampling strategies. IEEE Trans. Signal Inf. Process. Netw. **2**(4), 539–554 (2016)
44. M. Tsitsvero, S. Barbarossa, P. Di Lorenzo, Signals on graphs: uncertainty principle and sampling, IEEE Trans. Signal Process. (2016). https://doi.org/10.1109/TSP.2016.2573748
45. X. Wang, P. Liu, Y. Gu, Local-set-based graph signal reconstruction. IEEE Trans. Signal Process. **63**(9), 2432–2444 (2015)
46. L. Stanković, E. Sejdić, S. Stanković, M. Daković, I. Orović, A tutorial on sparse signal reconstruction and its applications in signal processing. Circuits Syst. Signal Process. 1–58 (2018)
47. L. Stanković, Digital signal processing with selected topics (2015)
48. S.K. Narang, A. Ortega, Downsampling graphs using spectral theory, in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2011), pp. 4208–4211
49. H.Q. Nguyen, M.N. Do, Downsampling of signals on graphs via maximum spanning trees. IEEE Trans. Signal Process. **63**(1), 182–191 (2015)
50. S.K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Trans. Signal Process. **60**(6), 2786–2799 (2012)
51. S. Segarra, A.G. Marques, G. Leus, A. Ribeiro, Interpolation of graph signals using shift-invariant graph filters, in *EUSIPCO* (2015), pp. 210–214
52. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Sampling of graph signals with successive local aggregations. IEEE Trans. Signal Process. **64**(7), 1832–1843 (2016)
53. A. Anis, A. Gadde, A. Ortega, Efficient sampling set selection for bandlimited graph signals using graph spectral proxies. IEEE Trans. Signal Process. **64**(14), 3775–3789 (2016)
54. H. Behjat, U. Richter, D. Van De Ville, L. Sörnmo, Signal-adapted tight frames on graphs. IEEE Trans. Signal Process. **64**(22), 6017–6029 (2016)
55. Y. Tanaka, A. Sakiyama, M-channel oversampled graph filter banks. IEEE Trans. Signal Process. **62**(14), 3578–3590 (2014)
56. A. Sakiyama, Y. Tanaka, Oversampled graph Laplacian matrix for graph filter banks. IEEE Trans. Signal Process. **62**(24), 6425–6437 (2014)
57. N. Tremblay, P. Borgnat, Subgraph-based filterbanks for graph signals. IEEE Trans. Signal Process. **64**(15), 3827–3840 (2016)
58. J. Leskovec, C. Faloutsos, Sampling from large graphs, in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2006), pp. 631–636
59. N. Perraudin, P. Vandergheynst, Stationary signal processing on graphs. IEEE Trans. Signal Process. **65**(13), 3462–3477 (2017)
60. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Stationary graph processes and spectral estimation. IEEE Trans. Signal Process. **65**(22), 5911–5926 (2017)
61. A. Loukas, N. Perraudin, Stationary time-vertex signal processing (2016), arXiv:1611.00255
62. S.P. Chepuri, G. Leus, Subsampling for graph power spectrum estimation, in *Sensor Array and Multichannel Signal Processing Workshop (SAM), 2016 IEEE* (IEEE, 2016), pp. 1–5
63. G. Puy, N. Tremblay, R. Gribonval, P. Vandergheynst, Random sampling of bandlimited signals on graphs. Appl. Comput. Harmon. Anal. (2016)

64. C. Zhang, D. Florêncio, P.A. Chou, Graph signal processing-a probabilistic framework. Microsoft Research, Redmond, WA, USA, Technical report MSR-TR-2015-31 (2015)
65. J. Friedman, T. Hastie, R. Tibshirani, Sparse inverse covariance estimation with the graphical Lasso. Biostatistics **9**(3), 432–441 (2008)
66. N. Meinshausen, P. Bühlmann et al., High-dimensional graphs and variable selection with the Lasso. Ann. Stat. **34**(3), 1436–1462 (2006)
67. E. Pavez, A. Ortega, Generalized Laplacian precision matrix estimation for graph signal processing, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2016), pp. 6350–6354
68. M. Pourahmadi, Covariance estimation: the GLM and regularization perspectives. Stat. Sci. 369–387 (2011)
69. S. Epskamp, E.I. Fried, A tutorial on regularized partial correlation networks. Psychol. Methods (2018)
70. A. Das, A.L. Sampson, C. Lainscsek, L. Muller, W. Lin, J.C. Doyle, S.S. Cash, E. Halgren, T.J. Sejnowski, Interpretation of the precision matrix and its application in estimating sparse brain connectivity during sleep spindles from human electrocorticography recordings. Neural Comput. **29**(3), 603–642 (2017)
71. X. Dong, D. Thanou, P. Frossard, P. Vandergheynst, Learning Laplacian matrix in smooth graph signal representations. IEEE Trans. Signal Process. **64**(23), 6160–6173 (2016)
72. C.-J. Hsieh, Sparse inverse covariance estimation for a million variables (2014)
73. X. Dong, D. Thanou, P. Frossard, P. Vandergheynst, Learning graphs from signal observations under smoothness prior (2015), arXiv:1406.7842
74. M. Slawski, M. Hein, Estimation of positive definite m-matrices and structure learning for attractive Gaussian Markov random fields. Linear Algebra Appl. **473**, 145–179 (2015)
75. S. Ubaru, J. Chen, Y. Saad, Fast estimation of tr(f(a)) via stochastic Lanczos quadrature. SIAM J. Matrix Anal. Appl. **38**(4), 1075–1099 (2017)
76. T.S. Caetano, J.J. McAuley, L. Cheng, Q.V. Le, A.J. Smola, Learning graph matching. IEEE Trans. Pattern Anal. Mach. Intell. **31**(6), 1048–1058 (2009)
77. D. Thanou, D.I. Shuman, P. Frossard, Learning parametric dictionaries for signals on graphs. IEEE Trans. Signal Process. **62**(15), 3849–3862 (2014)
78. E. Camponogara, L.F. Nazari, Models and algorithms for optimal piecewise-linear function approximation. Math. Probl. Eng. **2015** (2015)
79. T. Zhao, H. Liu, K. Roeder, J. Lafferty, L. Wasserman, The huge package for high-dimensional undirected graph estimation in R. J. Mach. Learn. Res. **13**, 1059–1062 (2012)
80. Y. Yankelevsky, M. Elad, Dual graph regularized dictionary learning. IEEE Trans. Signal Inf. Process. Netw. **2**(4), 611–624 (2016)
81. M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, D. Cai, Graph regularized sparse coding for image representation. IEEE Trans. Image Process. **20**(5), 1327–1336 (2011)
82. S. Segarra, A.G. Marques, G. Mateos, A. Ribeiro, Blind identification of graph filters with multiple sparse inputs, in *ICASSP* (2016), pp. 4099–4103
83. R. Rustamov, L.J. Guibas, Wavelets on graphs via deep learning, in *Advances in Neural Information Processing Systems* (2013), pp. 998–1006
84. L. Stanković, M. Daković, T. Thayaparan, *Time-Frequency Signal Analysis with Applications* (Artech House, Boston, 2014)
85. I. Jestrović, J.L. Coyle, E. Sejdić, A fast algorithm for vertex-frequency representations of signals on graphs. Signal Process. **131**, 483–491 (2017)
86. L. Stanković, M. Daković, E. Sejdić, Vertex-frequency analysis: a way to localize graph spectral components [lecture notes]. IEEE Signal Process. Mag. **34**(4), 176–182 (2017)
87. L. Stanković, E. Sejdić, M. Daković, Vertex-frequency energy distributions. IEEE Signal Process. Lett. (2017)
88. L. Stanković, E. Sejdić, M. Daković, Reduced interference vertex-frequency distributions. IEEE Signal Process. Lett. (2018)

89. J. Lefèvre, D. Germanaud, J. Dubois, F. Rousseau, I. de Macedo Santos, H. Angleys, J.-F. Mangin, P.S. Hüppi, N. Girard, F. De Guio, Are developmental trajectories of cortical folding comparable between cross-sectional datasets of fetuses and preterm newborns? Cereb. Cortex **26**(7), 3023–3035 (2015)
90. R.M. Rustamov, Average interpolating wavelets on point clouds and graphs (2011), arXiv:1110.2227
91. A. Golbabai, H. Rabiei, Hybrid shape parameter strategy for the RBF approximation of vibrating systems. Int. J. Comput. Math. **89**(17), 2410–2427 (2012)
92. D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames. IEEE Trans. Signal Process. **63**(16), 4223–4235 (2015)
93. D. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
94. H. Behjat, N. Leonardi, L. Sörnmo, D. Van De Ville, Anatomically-adapted graph wavelets for improved group-level fMRI activation mapping. NeuroImage **123**, 185–199 (2015)
95. I. Ram, M. Elad, I. Cohen, Redundant wavelets on graphs and high dimensional data clouds. IEEE Signal Process. Lett. **19**(5), 291–294 (2012)
96. A. Sakiyama, K. Watanabe, Y. Tanaka, Spectral graph wavelets and filter banks with low approximation error. IEEE Trans. Signal Inf. Process. Netw. **2**(3), 230–245 (2016)
97. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, Graph-based wavelet representation of multi-variate terrain data. Comput. Graph. Forum **35**(1), 44–58 (2016), Wiley Online Library
98. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, Lazy wavelet simplification using scale-dependent dense geometric variability descriptors. J. Control Eng. Appl. Inform. **19**(1), 15–26 (2017)
99. A. Dal Col, P. Valdivia, F. Petronetto, F. Dias, C.T. Silva, L.G. Nonato, Wavelet-based visual analysis of dynamic networks. IEEE Trans. Vis. Comput. Graph. (2017)
100. P. Valdivia, F. Dias, F. Petronetto, C.T. Silva, L.G. Nonato, Wavelet-based visualization of time-varying data on graphs, in *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)* (IEEE, 2015), pp. 1–8

Part II
Vertex-Frequency Theory

Transformation from Graphs to Signals and Back



Ronan Hamon, Pierre Borgnat, Patrick Flandrin and Céline Robardet

Abstract Network science has been a rapidly evolving field to study systems made of interactions between entities. Studying the structure of such networks reveals indeed the underlying mechanisms of these systems, and has been proven successful in many domains, such as sociology, biology, or geography. Recently, connections between network science and signal processing have emerged, making the use of a wide variety of tools possible to study networks. In this chapter, a focus is made on a methodology introduced to transform a graph into a collection of signals, using a multidimensional scaling technique: by projecting a distance matrix representing relations between vertices of the graph as points in a Euclidean space, it is possible to interpret coordinates of vertices in this space as signals, and take advantage of this dual representation to develop new tools for the study of networks. Deeper considerations of this methodology are proposed, by strengthening the connections between the obtained signals and the common graph structures. A robust inverse transformation method is next described, taking into account possible changes in the signals. Establishing a robust duality between graphs and signals opens up new perspectives, as classical signal processing tools, such as spectral analysis or filtering, are made available for the study of the structure of networks.

R. Hamon (✉) · P. Borgnat · P. Flandrin
Laboratoire de Physique, University of Lyon, Ens de Lyon, Univ Claude Bernard, CNRS, 69342 Lyon, France
e-mail: rhamon@protonmail.com

P. Borgnat
e-mail: pierre.borgnat@ens-lyon.fr

P. Flandrin
e-mail: patrick.flandrin@ens-lyon.fr

C. Robardet
University of Lyon, Insa Lyon, Univ Claude Bernard, CNRS, LIRIS, 69342 Lyon, France
e-mail: celine.robardet@insa-lyon.fr

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_2

1 Introduction

Network science [37] provides powerful tools to analyze systems in which relationships between entities naturally arise, whether passengers in transportation systems, users of a social network, or molecules in a chemical reaction. These tools have been proven very useful to extract relevant information from large amount of data represented as graphs, and mostly rely on ideas developed in other fields: The topic of the detection of communities [15, 16] is a relevant example of successful multi-disciplinary works, leading to many theoretical and applied results, based simultaneously on graph theory, statistical physics, and signal processing.

More recently, the connections between signal processing and network theory have tremendously increased, aiming at revisiting concepts from signal processing with a graph-based perspective, in the so-called graph signal processing field: works on the Fourier Transform operator [48], filtering of signals defined on the vertices of the graph [44, 47], spectral wavelets [21], graph filter banks [36, 38, 43, 51], vertex-frequency analysis [49], or sampling for graph signals [3, 7, 17, 33, 52], have then been proposed these recent years. If some of these works include the study of the network structure, for example using graph wavelets [50] detection of communities, graph signal processing is not primarily focused on the description of the structure of the graph as network science is.

Transforming signals to networks has also been a popular topic of research, which mainly focused on the study of nonlinear time series using network tools. As thoroughly described in [13], various approaches have been proposed, such as correlation networks [59, 61], recurrence networks [11, 12, 34] visibility graph [31, 39], or transition networks [6], to transform time series into networks and take advantage of the various network measures to extract significant properties about the dynamics of the corresponding time series. These works led to significant results in applications such as for heart rate [6] or earthquake [1] analyzes.

Conversely, mapping a graph into time series has been less intensively studied. In [6] is proposed a random walk based algorithm, to map graphs into time series by associating a specific value in the time domain with vertices, with the particularity that the graphs are themselves derived from time series. In [18], this approach is extended to the case where the graph is the object of interest, by using semi-supervised learning to map vertices to signal magnitudes such that the resulting time series are smooth. In [32], networks are constructed from sequences of notes of musical pieces, and then randomly followed to generate music. The resulting audio time series exhibit similar patterns as the original ones, that has been captured by the structure of the intermediary signals. In [58], a method is proposed to explore the structure of scale-free networks using finite-memory random walks, where the values of time series at time t are the degree of the vertex visited by the walker at step t. The resulting time series, obtained from different real-world networks, exhibit correlations, linked with the scale-free property of these networks. Other works have focused on using eigenvectors of the Laplacian matrix in a context of dimensionality reduction and data representation, such as Laplacian eigenmaps [4].

Using an alternative approach, a deterministic method based on multidimensional scaling [5] has been proposed in [25, 46] to represent the vertices of the graph as a set of points in a Euclidean space, where the relations described by the edges are represented by distances between points. The method consists in projecting the vertices in a Euclidean space whose dimension is equal to the number of vertices. The considered distance matrix between vertices is built from the adjacency matrix, conveying the simple presence or not of an edge between all pairs of vertices. Their observation is that in this configuration, the small-worldness of a graph, i.e., the property that the average length of the shortest paths between vertices is small when compared to the number of vertices, is visible through the periodicity of the corresponding time series. The role of eigenvectors in the characterization of the graph topology is well-known, and many algorithms rely on them to study various problems, such as spectral clustering [55]. An attractive feature of the approach considered in [46], as opposed to those previously described, is that it is a lossless transformation, preserving all the information about the relationships between vertices in the signals. These two representations are then completely equivalent, making the use of a wide range of processing techniques that are not applicable in the graph domain possible in the Euclidean space.

With these considerations in mind, a deep exploration of this methodology is reported in this chapter, to highlight the links between these signals and the topology of graphs. Already partially outlined in [22, 23], these contributions focus on unweighted graphs. This chapter intends to strengthen the work proposed in [46] on several points: first of all, some theoretical aspects of the method are discussed, and connections between the signals obtained after transformation and the topology of graphs are heightened. A detailed study of the inverse transformation is also proposed, taking into account possible changes in the signals. Establishing a robust duality between graphs and signals opens up new perspectives, as classical signal processing tools, such as spectral analysis or filtering, are made henceforth available for the study of the structure of networks. These perspectives are briefly discussed through two applications of the characterization of the structure of graphs through frequency patterns, and a short study of the effect of signal filtering on the graph structure.

The content is organized as follows: In Sect. 2, the transformation from graph to signals based on classical multidimensional scaling is described and studied, and a discussion is made on several common graph structures that may appear in networks. A robust inverse transformation for this approach is then discussed in Sect. 3 to transform back a collection of signals into a graph, establishing a comprehensive duality between graphs and multivariate signals. Finally, two applications of this duality are discussed in Sect. 4, to automatically extract significant graph structure using spectral analysis and process the graph using filtering on the corresponding signals.

Notations

Throughout the article, the following notations are adopted. G denotes a simple, undirected, and unweighted graph, with n vertices. Unless otherwise mentioned,

$n = 200$ for all illustrations. We denote by $\mathbf{A}$ the corresponding adjacency matrix, whose element a_{ij} is equal to 1 if there exists an edge between vertices i and j, 0 otherwise. Finally, $\boldsymbol{I}_n$ denotes the identity matrix of size n, and $\mathbf{1}_n\mathbf{1}_n^T$ the $n \times n$ matrix of ones.

2 From Graphs to Signals Using Multidimensional Scaling: Study and Illustrations

2.1 Projection of Vertices into Signals

2.1.1 Methodology

The core of the proposed approach is to project the vertices of the graph, living in the graph domain, in a more gentle Euclidean space, while preserving all the information about the relationships between vertices. In [46], the transformation from a graph with n vertices into a collection of signals of n points indexed by the vertices of the graph is based on multidimensional scaling (MDS) [5].

MDS is a set of mathematical techniques used to represent dissimilarities between pairs of objects as distances between points in a Euclidean space. Even if the resulting points have the same dimension as the original distance matrix, it is possible to dramatically reduce the dimension of the space without losing too much information between the dissimilarities between objects. The case where the dissimilarities are assumed to be Euclidean distances, i.e. the distances between points in the Euclidean space is equal to to the original dissimilarity matrix, is called Classical MDS (CMDS) and will be considered in the following. The algorithm to compute the coordinates $\boldsymbol{X}$ of n points in a n-dimensional Euclidean space, from a $n \times n$ Euclidean distance matrix $\boldsymbol{\Delta}$, is given in Algorithm 1, and amounts to applying Principal Component Analysis on a matrix derived from $\boldsymbol{\Delta}$.

Algorithm 1 Classical Multidimensional Scaling (MDS)

1: **procedure** CMDS($\boldsymbol{\Delta} = (\delta_{ij})_{i,j=1,\dots,n}$: Euclidean distance matrix between n objects)
2: Compute the centering matrix: $\boldsymbol{J} = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$
3: Perform the double centering of $\boldsymbol{\Delta}^{\cdot 2}$, where $\boldsymbol{\Delta}^{\cdot 2}$ denotes the matrix $\boldsymbol{\Delta}$ whose elements are squared: $\boldsymbol{B} = -\frac{1}{2}\boldsymbol{J}\boldsymbol{\Delta}^{\cdot 2}\boldsymbol{J}$
4: Diagonalize the matrix: $\boldsymbol{B} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$
5: Return: $\boldsymbol{X} = \boldsymbol{Q}_+\boldsymbol{\Lambda}_+^{\frac{1}{2}}$, where eigenvectors and eigenvalues are sorted in decreasing order of eigenvalues.

Applying CMDS to transform graphs into signals requires defining a dissimilarity measure between vertices. While common measures on graphs often consider the proximity between vertices in the graph, a simpler distance has been introduced in [46], than can be easily obtained from the adjacency matrix:

$$\boldsymbol{\Delta} = \boldsymbol{A} + w(\mathbf{1}_n \mathbf{1}_n^T - \boldsymbol{I}_n - \boldsymbol{A}) \tag{1}$$

where w is an arbitrary weight strictly greater than 1. This definition does not indeed convey (at least explicitly) the proximity between vertices beyond direct linkage: if two vertices are connected, their distance is equal to 1, otherwise it is equal to w. The remoteness between two vertices, which is commonly measured using the length of the shortest path between the two vertices, is then not taken into account in this definition: two pairs of unlinked vertices will have a distance equal to w, whether they are close or not in the graph. This measure allows nonetheless to characterize the structure of the network, as shown in the next sections.

In the rest of the chapter, we will denote the columns of the matrix $\boldsymbol{X}$ as signals or components: the jth signal, denoted $\boldsymbol{X}^{(j)}$, gives the coordinates of all vertices at dimension j of the embedding.

2.1.2 Existence of a Solution

For a given distance matrix $\boldsymbol{\Delta}$, an exact solution for the CMDS problem, i.e., a configuration of points $\boldsymbol{X}$ in a Euclidean space such that distances between these points are equal to distances defined in $\boldsymbol{\Delta}$, exists if and only if $\boldsymbol{\Delta}$ is a Euclidean distance matrix [8]. In the alternate case, only the closest configuration $\boldsymbol{X}$ is obtained. The choice of the distance measure given in Eq. (1) makes this property easily reachable, unlike a distance measure based on the length of the shortest path between the vertices. The Euclidean property of $\boldsymbol{\Delta}$ is nonetheless dependent on the value of w: if it is obvious that w should be strictly greater than 1, a too high value of w leads to non-Euclidean distances matrices, which can be easily verified in low dimension.

The calculation of an upper bound of w relies on the positive-semi-definiteness of the co-variance matrix of $\boldsymbol{X}$: $\boldsymbol{B} = \boldsymbol{X}\boldsymbol{X}^T$. As mentioned in the work of Gower [8], $\boldsymbol{\Delta}$ is Euclidean if and only if $\boldsymbol{B}$ is positive definite:

$$\langle z, \boldsymbol{B}z \rangle \geq 0 \text{ for all vectors } z \in \mathbb{R}^n \tag{2}$$

or equivalently, if and only if $\boldsymbol{\Delta}^{\cdot 2}$ is conditionally negative definite:

$$\langle z, \boldsymbol{\Delta}^{\cdot 2} z \rangle \leq 0 \quad \forall z \in \mathbb{R}^n \text{ such that } \sum_{i=1}^{n} z_i = 0 \tag{3}$$

From the definition of $\boldsymbol{\Delta}$ in Eq. (1), we have:

$$\begin{aligned} \langle z, \boldsymbol{\Delta}^{\cdot 2} z \rangle &= \langle z, \boldsymbol{A}z \rangle + w^2(\langle z, \mathbf{1}_n \mathbf{1}_n^T z \rangle - \langle z, \boldsymbol{I}_n z \rangle - \langle z, \boldsymbol{A}z \rangle) \\ &= \langle z, \boldsymbol{A}z \rangle - w^2(\langle z, z \rangle + \langle z, \boldsymbol{A}z \rangle) \end{aligned} \tag{4}$$

Two cases can be then distinguished:

1. If $\langle z, \boldsymbol{A}z\rangle > -\langle z, z\rangle$, then

$$w^2 \geq \frac{\langle z, \boldsymbol{A}z\rangle}{\langle z, \boldsymbol{A}z\rangle + \langle z, z\rangle} \tag{5}$$

   that is always verified as $w > 1$.
2. If $\langle z, \boldsymbol{A}z\rangle < -\langle z, z\rangle$, then

$$w^2 \leq \frac{\langle z, \boldsymbol{A}z\rangle}{\langle z, \boldsymbol{A}z\rangle + \langle z, z\rangle} \tag{6}$$

Equation (6) shows that the upper bound of w depends on the adjacency matrix $\boldsymbol{A}$, i.e., on the structure of the graph. A worst-case scenario is considered by choosing $\boldsymbol{A}$ and z such that $\langle z, \boldsymbol{A}z\rangle$ is minimal:

- $\boldsymbol{A}$ is defined as the adjacency matrix of a graph with n vertices, with n even, such that $a_{ij} = 1$ if and only if i and j do not belong in the same subset among $\{1, \ldots, \frac{n}{2}\}$ and $\{\frac{n}{2} + 1, \ldots, n\}$. $\boldsymbol{A}$ is then a 4-block matrix, with the bottom-left block and the top-right block equal to 1.
- As for z, it is equal to -1 for the first half of the vector and 1 for the last half of the vector: $z = [-1, -1, \ldots, 1, 1]$.

$\langle z, \boldsymbol{A}z\rangle$ is then equal to $-\frac{n^2}{2}$, while $\langle z, z\rangle = n$. With this structure, an upper bound of w is obtained:

$$w \leq \sqrt{\frac{n}{n-2}}. \tag{7}$$

This result is consistent with the partial results obtained empirically in [25, 46] on several instances of the Watts–Strogatz model and two real-world networks, stating that w should be close to 1, and depends on the number of vertices in the graph,

2.1.3 Unicity of the Solution

The solution achieved by CMDS is not unique, as any rotation, reflection, or translation of points in the Euclidean space will preserve the distances, and then will be a valid solution. The order in which vertices of the graph are labeled plays also an important part in the resulting signals, as it sets the indexation along the time axis. This order is then strongly related to the smoothness of the signals, and thus their interpretability. As a rule, the numbering of vertices in the graph representation should closely follow the topology of the graph, in a "natural" order: Ordering randomly the vertices does not change the value assigned to each vertex, but lead to abrupt variations when representing signal. One relevant approach to get such a labeling of vertices is to solve a graph labeling problem called cyclic bandwidth sum problem [28], looking for a mapping from vertices to integers, in such a way that the

sum of the distances in the vertex ordering between all pairs of connected vertices is minimized. In [24], a heuristic has been proposed to obtain a labeling based on the resolution of this problem, that has been shown to be compliant with the topology of the graph. In the following, the labeling of all considered graphs are obtained using this heuristic.

2.1.4 Class of Admissible Solutions

As the resulting signals define an exact representation of a given graph, they are restricted to a very limited class of multidimensional signals. Here are some elements to help to characterize this class $\mathscr{C}$.

If $\mathbf{X} \in \mathscr{C}$, then:

- $\mathbf{X} \in \mathbb{R}^{n \times n-1}$
- $\forall 1 \leq i < j \leq n, \|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2 = 1$ or $\|\boldsymbol{X}_i - \boldsymbol{X}_j\|_2 = w$
- $\forall 1 \leq i < j \leq n, \|\boldsymbol{X}_i\|_2 \geq \|\boldsymbol{X}_j\|_2$.

This restricts the solution to a very small set, and limits the capacity of the method to handle collection of signals that would not be obtained from a graph, in particular for an inverse transformation. In Sect. 3, a robust inverse transformation is discussed to transform back collections of signals that have been perturbed into a graph, relaxing the strict conditions of the class $\mathscr{C}$.

2.2 Application on Common Graph Structures

Real-world networks exhibit various kind of structures, that have been studied and discussed in numerous works [37]. In the following, the signals obtained after transformation of instances of such common models are displayed and discussed, highlighting relations between both representations.

2.2.1 Erdös–Rényi Model

A random graph is a graph whose edges between vertices are randomly drawn, according to a given distribution. The most famous and simple random model, whose properties have been extensively studied, is the Erdös–Rényi model [14], where each of the $\frac{n(n-1)}{2}$ possible edges are drawn independently according to a Bernoulli distribution with p the probability of existence. Instances of this model are known to exhibit a structure that is different of real-world networks, the graph being a grouping of vertices, whose cohesion is controlled by the probability p.

Studying the eigenvalues of adjacency matrix of random graphs has been the topic of many works [9, 10, 54], as well as, in a lesser extent, the study of eigenvectors [40]. The asymptotic results of interest are the following:

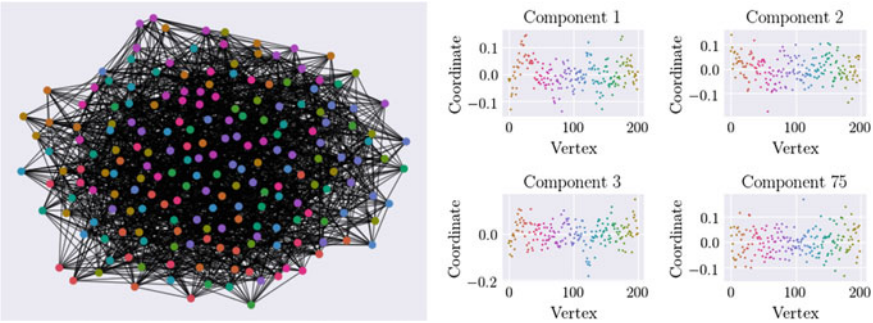


Fig. 1 Transformation of an instance of the Erdös–Rényi model with $p = 0.1$ into a collection of signals (**ER**). For each sub-figure, the left plot shows a two-dimensional representation of the graph, where labels are color-coded. The right plot displays four components of the collection, where the color of points corresponds to the color of the corresponding vertex in the graph

- the largest eigenvalue tends to np;
- other eigenvalues follow a semicircle law on the interval $[-2, 2]$;
- eigenvectors are random i.i.d. Gaussian vectors in $\mathbb{R}^n$.

These results are not sufficient to characterize the resulting signals $\boldsymbol{X}$, as they are based on the eigenvectors and eigenvalues of the matrix $\boldsymbol{B}$ (see Algorithm 1), itself based on the distance matrix $\boldsymbol{\Delta}$. Nonetheless, it is sound to conjecture that they own similar properties, confirmed by empirical simulations. The resulting signals can be then viewed as white noise components, whose amplitudes are given by the eigenvalues of $\mathbf{B}$.

In Fig. 1 is shown a 2-D representation of an instance of the Erdös–Rényi model with $p = 0.1$, and four components of the collection of signals after transformation. Random structure visible in the graph can be found as well as the lack of structure of components, that can be associated to white noise signals.

2.2.2 *k*-Regular Lattices

A k-ring lattice is a graph, in which each vertex, labeled by an integer $i \in \{1, \ldots, n\}$, is connected to the vertices $\{i - \frac{k}{2}, i - \frac{k}{2} + 1, \ldots, i - 1, i + 1, \ldots, i + \frac{k}{2} - 1, i + \frac{k}{2}\}$, for $k \in \{2, 4, \ldots, \frac{n}{2}\}$. As discussed in [46], it is straightforward to find the expression of expected eigenvalues and eigenvectors using circulant matrix theory [19]. The connection between the parameters w, k and the resulting signals are made explicit in the following.

Results from the circulant matrix theory are first recalled. Any circulant matrix $\boldsymbol{C}$ has its vector of eigenvalues $\boldsymbol{\lambda}$ given by:

$$\forall q \in \{0, \ldots, n-1\}, \quad \lambda_q = \sum_{j=0}^{n-1} c_j \zeta^{kj} \tag{8}$$

where $\boldsymbol{c}$ is the circulant vector of $\boldsymbol{C}$ and $\zeta = e^{\frac{2j\pi}{n}}$ is the nth root of the unity. As for eigenvectors, they are given by:

$$\forall q \in \{0, \ldots, n-1\}, \quad \boldsymbol{v}_q = \sqrt{n}[1, \zeta^q, \zeta^{2q}, \ldots, \zeta^{(n-1)q}] \tag{9}$$

corresponding to the columns of the Fourier matrix, denoted $\boldsymbol{F}^{(q)}$. These eigenvalues and eigenvectors appear as complex conjugate pairs, namely $\bar{\lambda}_q = \lambda_{n-q}$ and $\bar{\boldsymbol{v}}_q = \boldsymbol{v}_{n-q}$ for $q \neq 0$. As we consider symmetric matrices, the eigenvalues are real and double ($\lambda_q = \lambda_{n-q}$ for $q > 0$), while the corresponding eigenvectors are the real and imaginary parts of $\boldsymbol{v}_q$, normalized by $\sqrt{2}$ to obtain an orthonormal matrix:

$$\boldsymbol{v}_q = \sqrt{2}\Re(\boldsymbol{F}^{(q)}) = \sqrt{2}\cos\left(\frac{2\pi q}{n}\right) \tag{10}$$

$$\boldsymbol{v}_{n-q} = \sqrt{2}\Im(\boldsymbol{F}^{(q)}) = \sqrt{2}\sin\left(\frac{2\pi q}{n}\right) \tag{11}$$

corresponding to harmonic oscillations. If n is even, $\lambda_{\frac{n}{2}}$ is single and the corresponding eigenvector is not normalized by $\sqrt{2}$.

When applied to the matrix $\boldsymbol{B}$, and recalling that entries of the matrix $\boldsymbol{\Delta}$ can take only three possible values 0, 1, or w, the circulant vector $\boldsymbol{b}$ is defined by

$$b_i = -\frac{1}{2}\left[\delta_i^2 - \frac{\alpha}{n}\right] \tag{12}$$

with $\alpha = k + (n - 1 - k)w^2$. The vector $\boldsymbol{b}$ can then take three possible values:

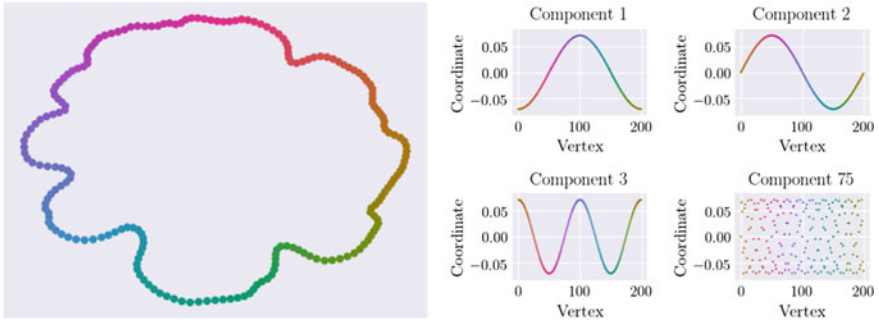
$$b_i = \begin{cases} \frac{\alpha}{2n} \text{ if } i = 0 \\ -\frac{\alpha}{2}(1 - \frac{\alpha}{n}) \text{ if } i \in \{1, \ldots, \frac{k}{2}\} \cup \{n - \frac{k}{2}, \ldots, n-1\} \\ -\frac{\alpha}{2}(w^2 - \frac{\alpha}{n}) \text{ if } i \in \{\frac{k}{2} + 1, \ldots, n - \frac{k}{2} - 1\} \end{cases}. \tag{13}$$

This leads to the following expression of the eigenvalues of $\boldsymbol{B}$:

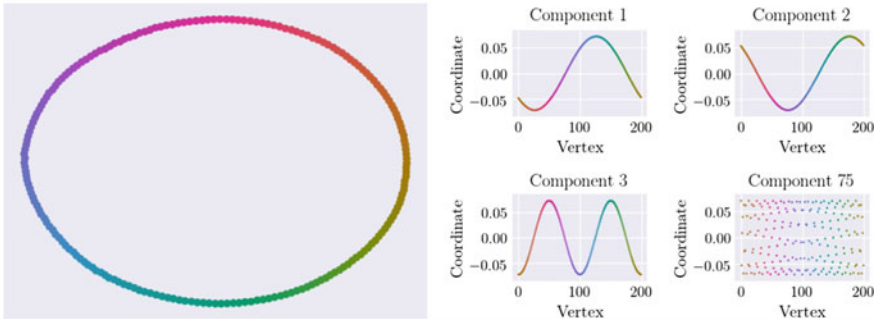
$$\lambda_q = \frac{\alpha}{2n}\sum_{m=0}^{n-1}\zeta^{mq} - \frac{1}{2}\left(\sum_{m=1}^{\frac{k}{2}}\zeta^{mq} + \sum_{m=n-\frac{k}{2}}^{n-1}\zeta^{mq} + w^2\sum_{m=\frac{k}{2}+1}^{n-\frac{k}{2}-1}\zeta^{mq}\right) \tag{14}$$

When the eigenvalues are ordered by the value of q, the eigenvectors are considered in increasing order of frequencies. The components are however sorted according to the energy of eigenvalues λ_k after applying CMDS, which correspond to the sorting according to the value of q only when $k = 2$. In this case, the resulting signals are then harmonic oscillations whose frequencies increase as lower-energy components are considered. When k increases, the components are no longer sorted by frequencies.

Figure 2 highlights this phenomenon, by displaying the obtained signals for a 2-ring lattice (Fig. 2a) and a 6-ring lattice (Fig. 2b). All obtained signals are consistent



(a) 2-ring lattice (**RL-2**)



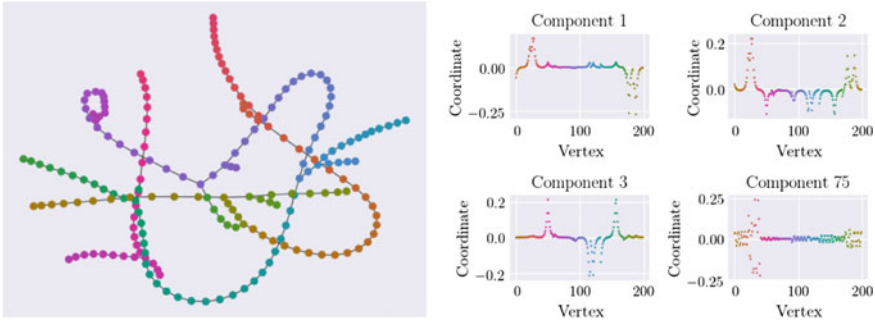
(b) 6-ring lattice (**RL-6**)

Fig. 2 Transformation of two k-ring lattices into a collection of signals. See Fig. 1 for details

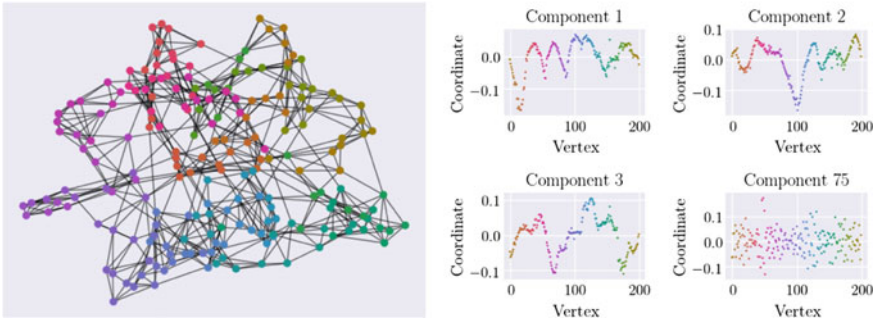
with theoretical results, i.e., the signals are harmonic oscillations, grouped in pairs of signals with the same frequency and a difference of phase equal to $\frac{\pi}{2}$. For the high-energy components (here, components 1, 2, and 3), the value of k does not have any influence. However, the frequency of signals differs, as expected, for lower-energy components (here component 75), as the sorting of components is guided by the eigenvalues.

2.2.3 Watts–Strogatz Model

The Watts–Strogatz model [57] has been developed to build graphs with small-world property, where the average length of the shortest paths between vertices is small compared to the number of vertices, while the clustering coefficient, i.e., the grouping of vertices based on linkage, is high. This property has notably been highlighted in many real-world systems, in particular those including social interactions. The construction of Watts–Strogatz instances starts from a k-ring lattice, and is performed by rewiring each edge with a given probability. Edges may then appear or disappear, all the more so as p is high. In [45], a second-order approximation of the expected



(a) $k = 2$ (**WS-2**)



(b) $k = 6$ (**WS-6**)

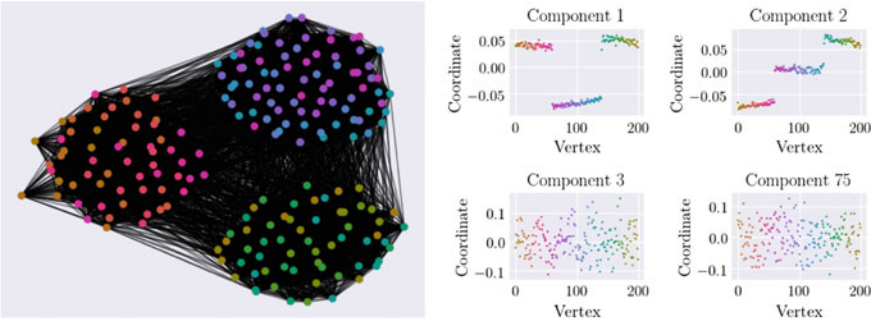
Fig. 3 Transformation of two instances of the Watts–Strogatz model, with $p = 0.05$, into a collection of signals. See Fig. 1 for details

eigenvalues and eigenvectors according to the probability p is derived using perturbation theory [30]. It is shown that the correlation between the approximation and the actual signals is high when p is low, and decreases when p increases, which is consistent with the intuition one can have.

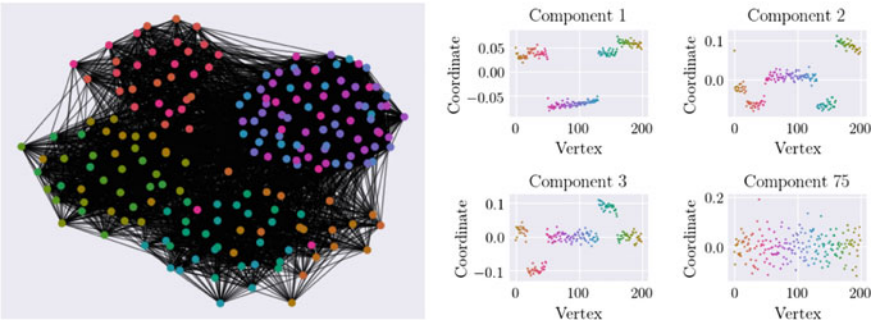
The resulting signals take also into account the noise that is added in their shape, when compared with signals obtained for k-ring lattice. Two instances of the Watts–Strogatz model with a probability of rewiring equal to $p = 0.1$ are considered in Fig. 3. As expected, signals also display a harmonic trend, with a noisy mode whose influence on the shape depends on the value of k: for $k = 6$, the regular structure has more edges than for $k = 2$, and is then more resilient to the change: preserving the ring structure. However, when $k = 2$, one single rewiring may cause the disappearance of the ring, affecting much more the structure and then the shape of the signals.

2.2.4 Stochastic Block Model

A stochastic block model [29] is used to generate a graph with communities. Each of the n vertices is assigned to one of the K communities, and edges between each pair of



(a) 3 communities (**SBM-3**)



(b) 5 communities (**SBM-5**)

Fig. 4 Transformation of an instance of the stochastic block model, with $p_{intra} = 0.8$, and $p_{inter} = 0.1$, into a collection of signals. See Fig. 1 for details

vertices is randomly and independently drawn according to probabilities depending on the group of vertices: if the two vertices belong to the same community, the probability, noted p_w, is close to 1 while otherwise, the probability between vertices of different groups, noted p_b, is lower. The settings of p_w and p_b lead to customized density of edges within and between communities. In [35], intuitions about the shape of the signals are given, in an application of segmentation of images. They suggest that the eigenvectors of block matrices are piece-wise constant with respect to the communities.

Figure 4a, b show two instances of the stochastic block model with $p_{intra} = 0.8$, and $p_{inter} = 0.1$, respectively with three and five communities. An interesting observation is that the high-energy components contain the structure of communities, as conjectured above, with noisy plateaus corresponding to the dense parts of the graph. The number of relevant high-energy components is equal to the number of communities in the graph minus one, as one component is sufficient to discriminate two communities. As for the low-energy components, they are noisy signals, corresponding to the structure inside communities, as it can be observed in Fig. 1.

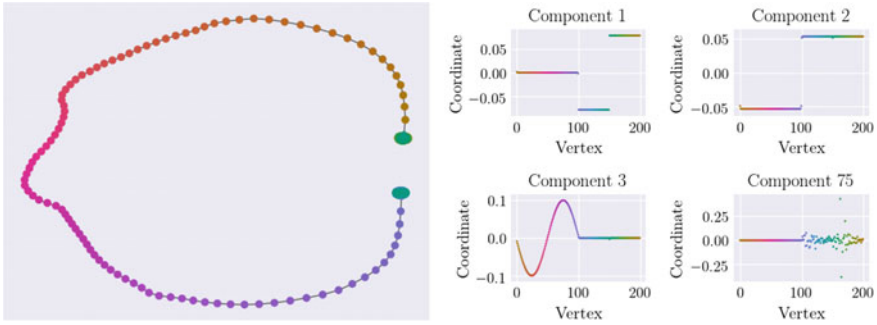


Fig. 5 Transformation of an instance of the Barbell model, with $n_c = 50$ and $n_p = 100$ into a collection of signals (**BAR**). See Fig. 1 for details

2.2.5 Barbell Model

The Barbell model [2] is defined as two cliques, each containing n_c vertices, linked by a path with n_p vertices. It is then a good example of combined structures, as half vertices are divided into two communities, while the other half have a regular path structure.

The shape of the resulting signals displayed in Fig. 5 can be analyzed through the results obtained of the previous examples. First component has indeed plateaus, as for the stochastic block model, describing the cliques as well as the regular part, seen in a first approximation as a clique. Next components describe the regular structure for these vertices with an harmonic oscillation, present on the third component. Finally noisy signal for low-energy components, as seen for the Erdös–Rényi model, represent the dense structure inside cliques.

These illustrations show the connections between graph structure and the resulting signals after transformation, that will be used in Sect. 4 to study the topology of the graph using spectral analysis, and to apply standard techniques, such as filtering, on graphs. Before that, a robust inverse transformation is proposed, to provide a reliable tool to represent altered signals into graph.

3 From Signals to Graph: A Robust Inverse Transformation

3.1 Inverse Transformation: Description of the Problem

The transformation from a graph to a collection of signals defined in Sect. 2 fulfills the objective of a dual representation of graphs as a collection of signals in a Euclidean space. By construction, the transformation described in Algorithm 1 is indeed fully

invertible: computing the distance between any two points gives the distance matrix $\boldsymbol{\Delta}$, from which the retrieval of the adjacency matrix is straightforward (for a suitable value of w). This inverse transformation can only be applied on collection of signals belonging to the class of admissible solutions described in Sect. 2.1.4, that limits it to signals directly obtained from a graph. It is nonetheless worth considering an extension of the inverse transformation to other collection of signals: one may want for instance to process the collection of signals, and study the effect on the corresponding graph, which is the topic of Sect. 4. This case is yet non-trivial, as the distribution of distances is no longer bimodal, but has rather a continuous set of values, spreading around the values of 1 and w all the more that the extent of the disruption is important.

From a perturbed collection of signals, there exist two approaches can be considered to find the best graph represented by these signals:

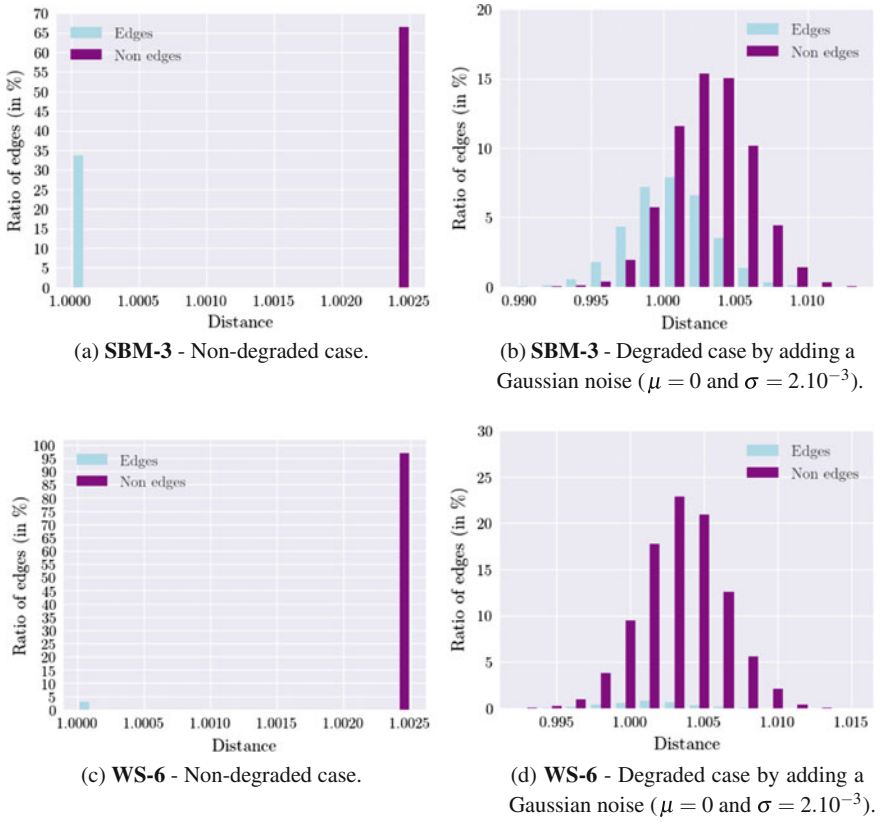
1. Considering the collection of signals, looking for the closest configuration of points that belongs to the class of admissible solutions;
2. Starting from the distribution of Euclidean distances between all pairs of points, finding the distances that correspond to an edge between the corresponding vertices in the graph.

In the rest of this section, the focus is made on the second approach. Two contributions are explored to increase the separability of distances into a bimodal distribution of distances, and to compute a suitable threshold that will extract distances representing edges. These contributions are validated through illustrations, that highlight the good behavior of the inverse transformation with respect to different perturbations of the collection of signals.

3.2 Thresholding of Distances

Transforming back a collection of signals into a graph comes to find a threshold above which a distance between two points in the Euclidean space is sufficiently high to indicate that the two corresponding vertices in the graph are not linked. In the case where the collection of signals is the result of a transformation of a graph, the threshold is straightforward: any value between 1 and w will return the original distance matrix $\boldsymbol{A}$ of the graph, as illustrated by the distribution of distances displayed in Fig. 6a for an instance of the stochastic block model, or in Fig. 6c for an instance of the Watts–Strogatz model.

When the collection of signals is degraded, finding such a threshold is not an easy task, as the distribution of distances may not highlight any natural separation in two distinct distributions. In Fig. 6b, d are displayed the distributions of distances from a collection of signals $\boldsymbol{X}$ on which has been added a Gaussian noise of mean 0 and standard deviation 2.10^{-3}, respectively for the stochastic block model and the Watts–Strogatz instances. The distributions of distances representing edges and the one of distances representing non-edges now overlap, preventing any relevant thresholding



(a) **SBM-3** - Non-degraded case.

(b) **SBM-3** - Degraded case by adding a Gaussian noise ($\mu = 0$ and $\sigma = 2.10^{-3}$).

(c) **WS-6** - Non-degraded case.

(d) **WS-6** - Degraded case by adding a Gaussian noise ($\mu = 0$ and $\sigma = 2.10^{-3}$).

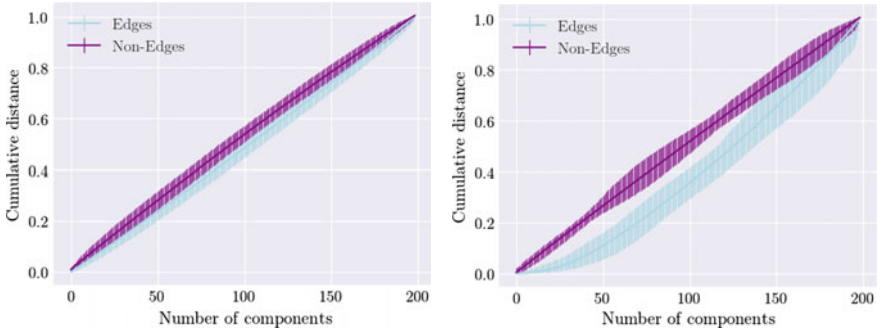
Fig. 6 Distribution of distances between points in the Euclidean space. Light blue bars represent distances corresponding to edges while dark purple bars represent distances corresponding to non-edges

that would separate the distances in two groups. Furthermore, this mixture does not exhibit any bimodal distribution that would describe another graph structure.

The issue is thus how to retrieve anyway a graph structure from a similar distribution of distances. With this objective in mind, two contributions are explored in this section, decomposing the problem into the two following questions:

1. How to separate several modes from the distributions of distances, and increase their separability?
2. From a multi-modal distribution of distances, what is the most suitable threshold that would extract the most significant graph structure?

Each contribution is an address to one of these two questions: the first one takes advantage of the energy of components to compute structure-aware distances, while the second contribution introduces a thresholding technique based on the assumption



(a) **SBM-3** (b) **WS-6**

Fig. 7 Cumulative distance according to the number of retained components. When all components are retained, the distances are either equal to 1 (for distances that represent edges, in light blue) or w (for distances that represent non edges, in purple). Each point gives the mean value of the distance, while the error bar gives the standard deviation above and below the mean

of relatively sparse graphs to discriminate the distances and retrieve the adjacency matrix that describes at best the collection of signals.

3.2.1 Energy-Based Computation of Distances

The distance between two points in a Euclidean space of dimension d is defined as the sum of the squared difference between coordinates of points:

$$d(\boldsymbol{X})_{ij} = \sqrt{\sum_{k=0}^{n-1} (x_{ik} - x_{jk})^2} \tag{15}$$

Each dimension contributes to the total sum, and this contribution varies according to the pair of points considered. In particular, the relationship described by the distance, i.e., the presence or not of an edge between the two corresponding vertices in the graph, has a strong influence on the distribution of contributions over the components. In Fig. 7 is plotted the average cumulative distance according to the number of retained components, for the two kinds of relationships that distances describe. One can observe that the contribution of first components for distances denoting an absence of edge is significantly greater than for distances denoting an edge, to a greater proportion than a uniform contribution of all components would give. This is particularly true for highly-structured networks, as for the Watts–Strogatz instance in Fig. 7b.

This observation is consistent with the one made in Sect. 2.2, that highlighted that high-energy components have a strong influence in the description of the global topology of the graph: If the distance between two vertices i and j in a high-energy

component is high, it means that the two vertices are likely to be distant in the graph. Conversely, if the distance in a high-energy component is low, then the two vertices are likely to be connected in the graph. Neglecting the importance of energies of components comes to forget the hierarchy of components in their description of the global structure of the graph.

The proposed approach takes profit of this specificity to inject the energy in the computation of distances, by defining a distance weighted by the energy of the considered component:

$$d_\alpha(\boldsymbol{X})_{ij} = \sqrt{\frac{1}{\sum_{k=1}^{K} e_k^\alpha} \sum_{k=1}^{K} e_k^\alpha (x_{ik} - x_{jk})^2} \tag{16}$$

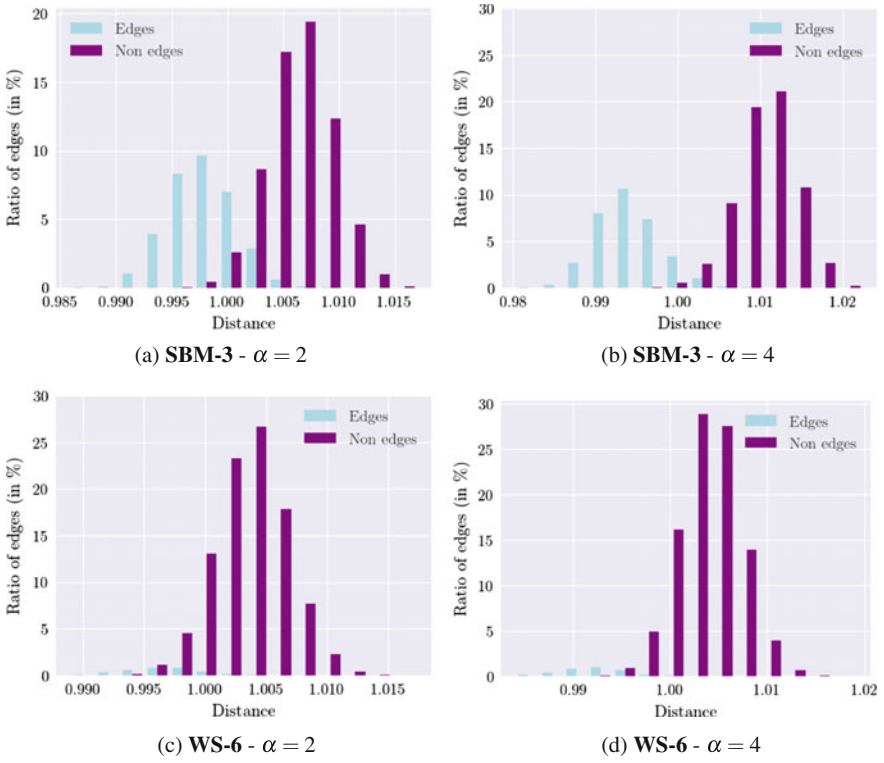
where e_k is the energy of component k, computed as $e_k = \sum_{i=1}^{n} x_{ik}^2$, and $\alpha \geq 0$ is a parameter that controls the importance of the weighting: the higher α is, the higher the contribution of high-energy components in the computation of the distance is.

Figure 8 shows the distribution of distances representing edges and those representing non-edges for a collection of signals after transformation of an instance of the stochastic block model (**SBM-3**) and of the Watts–Strogatz model (**WS-6**), degraded by adding a Gaussian noise of mean 0 and standard deviation 2.10^{-3} using respectively $\alpha = 2$, and $\alpha = 4$. Comparing these histograms with the ones in Fig. 6 (corresponding to the unweighted case $\alpha = 0$), it can be observed that taking into account the energy of components in the computation of distances leads to a better separation of the distribution into two modes, that are gradually split as α increases. These results do not depend on the number of edges in the graph, as the same effect appears for the two instances.

There is no obvious choice for α, which may cause strong changes in the structure of the resulting graph. The higher α is, the greater the distortion of the distribution of distances is. An appropriate way to select α is then to empirically assess the reconstructed structure, setting α proportionally to the intensity of the perturbation of signals.

3.2.2 Sparse Hypotheses for Thresholding Distances

The selection of the threshold is a crucial step to discriminate between the distances that represent edges from those representing non-edges. In [46], the reconstructed graph is constrained to have the same number of edges as the original one: the selection of the threshold is then computed by considering the smallest distances as edges until the number of edges in the original graph is reached. This approach has the major disadvantage that it does not allow for major changes of the structure of the graph. If we assume that it exists a technique that would return a collection of signals that is the exact representation of a graph with a different structure, keeping the same number of edges would not return the expected graph, despite the fact that



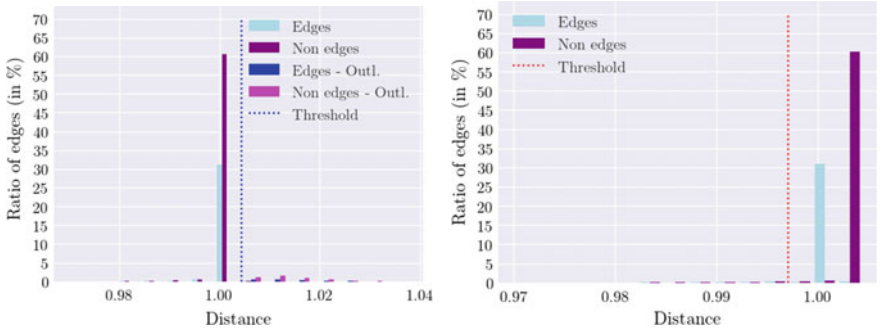
(a) **SBM-3** - $\alpha = 2$ (b) **SBM-3** - $\alpha = 4$

(c) **WS-6** - $\alpha = 2$ (d) **WS-6** - $\alpha = 4$

Fig. 8 Distribution of energy-based distances between points in the Euclidean space for different values of α, for a collection of signals $\boldsymbol{X}$ degraded by adding a Gaussian noise of mean 0 and standard deviation 2.10^{-3}

the discrimination is obvious. In more realistic situations, a strong perturbation of the collection of signals will significantly change the structure of the underlying graph, and then the number of edges.

Thresholding methods have been an active area of research in image computing for image segmentation: in the global approach, a threshold (or a set of threshold in the case of multimodal segmentation) is computed from a gray-scale image, such that the main features of the image are preserved when the number of levels is reduced, typically 2 for black and white images. This led to the well-known algorithm of binarization proposed by Otsu [41]: Considering two classes, the algorithm finds the threshold among all possible thresholds that minimizes the variance within classes. In our context, the distribution is composed of all distances between pairs of points, which differs from gray levels, since the number of possible distances might be equal to the number of pairs of vertices, while in images, the number of levels is fixed (for instance 256 levels for an 8-bit color images, whatever the size of the image). A thresholding approach based on these techniques is nevertheless proposed, by considering a Gaussian mixture model, that can be viewed as a generalization of



(a) With outliers. (b) Without outliers.

Fig. 9 Distribution of distances between points in the Euclidean space for a collection of signals $\boldsymbol{X}$ obtained after transformation of the instance **SBM-3**, degraded by adding a Gaussian noise of mean 0 and standard deviation 10^{-2} to the first ten components

the Otsu's model [26], and returns a suitable threshold using an EM algorithm, more adapted to the distribution of distances than the original Otsu's algorithm.

The distribution of distances is assumed to be composed of two Gaussian distributions, each of them representing the presence or the absence of edge. A first threshold is obtained by estimating and averaging the means of each sub-distribution. This threshold has the major drawback to be very sensitive to outliers, and in particular if the distribution of non-edges distances is composed of several modes, as it is the case for instance if there are only a few points that are moved. In Fig. 9a is plotted the histogram of distances for a collection of signals obtained after transformation of the instance **SBM-3**, degraded by adding a Gaussian noise of mean 0 and standard deviation 10^{-2} to the first ten components. The distribution is clearly shifting to the right, and the obtained threshold is too high and leads to a graph with many edges, blurring the relevant graph structure.

Algorithm 2 Robust thresholding with Gaussian mixtures

procedure ROBUST_THRESHOLDING($\boldsymbol{d}$: distribution of distances)
  Init O the set of outliers as empty.
  Set the threshold τ to $\texttt{Max}(\boldsymbol{d})$.
  while The number of edges is higher than $\frac{n(n-1)}{4}$ **do**
    Find the means of two Gaussian mixtures in $\boldsymbol{d}$ using the EM algorithm on the distances without the outliers.
    Define τ as the average of the means of the two obtained distributions.
  return τ

A robust procedure is given in Algorithm 2 to cope with outliers values. The feature is to get rid of outliers, based on a constraint of sparsity of the reconstructed graph, i.e., a small number of edges compared to the number of vertices. This constraint is ensured by recalculating a suitable threshold if the graph is not sparse enough, the

distances above the threshold being considered as outliers and then excluded of the distribution. Here, a loose constraint is implemented, by setting the maximal number of edges to $\frac{n(n-1)}{4}$, that amounts to saying that the graph should have more pairs of vertices without an edge that pairs with an edge. The outcome of the procedure is displayed on Fig. 9b, that shows the histogram without the outliers, and the obtained threshold.

3.3 Experiments

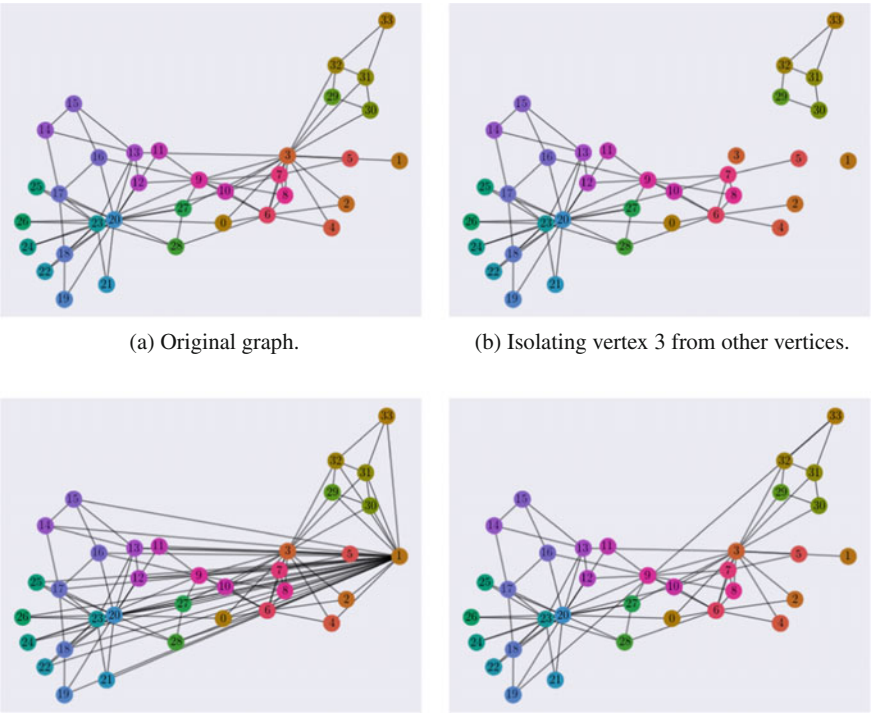
A thorough evaluation of the performance of the robust inverse transformation introduced in the previous section does not have much sense in this context, as on the one hand, if the collection of signals is obtained using the direct transformation, then the inverse transformation is immediate and exact, and on the other hand, if the collection of signals has been altered, whatever the process, it is unlikely to be an exact representation of a graph, and the expected result is unknown. To nonetheless assess the relevance of the proposed inverse transformation, qualitative experiments in specific context are set up to understand how the features help to recover a graph structure behind altered signals. The following experiments are performed on a limited set of instances, that are representative on common structures that appear in networks. It is not intended to provide a rigorous proof of the validity of the inverse transformation, but rather to demonstrate its soundness in the context of this work.

3.3.1 Localized Perturbations

A simple graph, the Zachary's Karate Club graph [60], which is displayed in Fig. 10a, is first considered, on which localized perturbations are performed, to easily visualize the consequences on the structure of the graph. Three modifications of the collection of signals obtained after transformation are considered:

1. isolating the point 3 from other points by multiplying its coordinates by 1.1;
2. bringing the point 1 closer to other by dividing its coordinates by 1.05;
3. bringing the point 33 closer to the point 19 by adding 0.04 times the coordinates of point 19 to the coordinates of point 33.

The resulting graphs are displayed in Fig. 10, showing how modifications made on the collection of signals affect the structure of the underlying graph. In Fig. 10b, moving the point 3 far from other points impact the edges that vertex 3 has with other vertices, that are removed. This has for consequences to split the graph into two main components, with two isolated vertices. At the contrary, when moving a point closer to the other points in the Euclidean space, new edges are added with respect to the original graph, as illustrated in Fig. 10c with the vertex 1. It is also possible to create only one unique edge by moving the point towards the direction of another point,



(a) Original graph.

(b) Isolating vertex 3 from other vertices.

(c) Getting vertex 1 closer to other vertices.

(d) Getting vertex 33 and 19 closer.

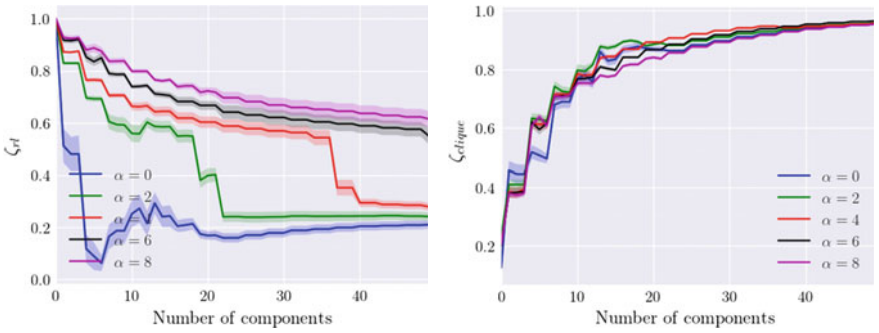
Fig. 10 Graphs obtained after processing the collection of signals obtained after transformation of the Zachary's Karate Club graph [60]. Three kinds of processing are considered on a specific point, that is translated in the resulting graphs by the addition or the deletion of edges

as illustrated in Fig. 10d, where the vertex 33 is moved towards the position of the vertex 19, adding a link between the two vertices in the reconstructed graph.

This experiment shows how the two representations are equivalent, and that processing the signals comes to processing the graph itself. These basic examples also highlight the soundness of the proposed inverse transformation, that is able to consistently deal with non bimodal distributions of distances.

3.3.2 Global Perturbations

Perturbations that affect all vertices of the graph are more difficult to control and to visualize, as the position of all points in the Euclidean space are moved, leading to radical change in the structure of the reconstructed graph. In this experiment, a controlled process is setup, to progressively moving half of the points such that a community emerges from an initial regular structure.



(a) Similarity to a k-ring lattice.

(b) Density of edges between the even vertices.

Fig. 11 Results obtained with respect to the number of altered components, averaged over the value of k. Confidence intervals are obtained using bootstrap

Starting from a k-ring lattice, the coordinates of points with an even label, i.e. half the points, are set to positive values, for every three components. Using previous notations, the new collection of signals $\tilde{\boldsymbol{X}}$ is defined by $\forall i \in \{1, \ldots, n\}, \forall c \in \{1, \ldots, n-1\}$:

$$\tilde{x}_{ic} = \begin{cases} |x_{ic}| & \text{if } i\%2 = 0 \text{ and } c\%3 = 0 \\ x_{ic} & \text{otherwise} \end{cases} \tag{17}$$

where % is the modulo operator. This operation brings half of the vertices closer to each other, as their signs match every three components, and thus tend to form a community. The other points do not move and keep a regular lattice structure.

The obtained graph structure is described through two measures ζ_{rl} and ζ_{clique}, that respectively measure the similarity of the structure with a k-ring lattice and the density of edges between even vertices. This is evaluated for k ranging from even values from 2 to 50 included, and for different values of α.

Figure 11 shows the obtained results: as expected when there is only a few components that are altered, the structure of the graph is close to a k-ring lattice, and the density of edges in the community of even vertices is low. Progressively, these trends are reversed, and the community structure emerges, degrading the regular structure. The level of these trends is worth comparing according to the value of α: for the structure in community, which becomes more and more dominant in the graph, the choice of α does not have a strong influence, as seen in Fig. 11a. It becomes important for the reconstruction of the regular lattice: as shown in Fig. 11b, the higher α is, the better the regular lattice is preserved. More important, even a low non-negative value of α retrieves the lattice from the distances, at the contrary of the regular unweighted distances that completely obfuscate it.

To conclude this section, a robust inverse transformation has been proposed, based on two contributions that significantly improve the retrieval of a graph structure from

a degraded collection of signals. The next step is to take advantage of this robust representation to analyze the graph in the signal domain, which is done in the next section.

4 Signals-Based Processing of Graphs

Previous sections have introduced a comprehensive duality between graphs and signals, using a direct transformation that preserves the relationships between the vertices, and a robust inverse transformation method that allows to transform back this collection of signals into a graph, even if it has been altered through a processing technique. These tools provide another representation of a graph beyond the graph domain, by having access to Euclidean data that can be exploited to explore new graph processing techniques. In this section, two illustrations of these new possibilities are discussed, first by exploiting the shape of signals to characterize the graph, and second to apply basic filtering of signals to retrieve the main structure of networks.

4.1 Connection Between Frequency Patterns of Signals and Graph Structures

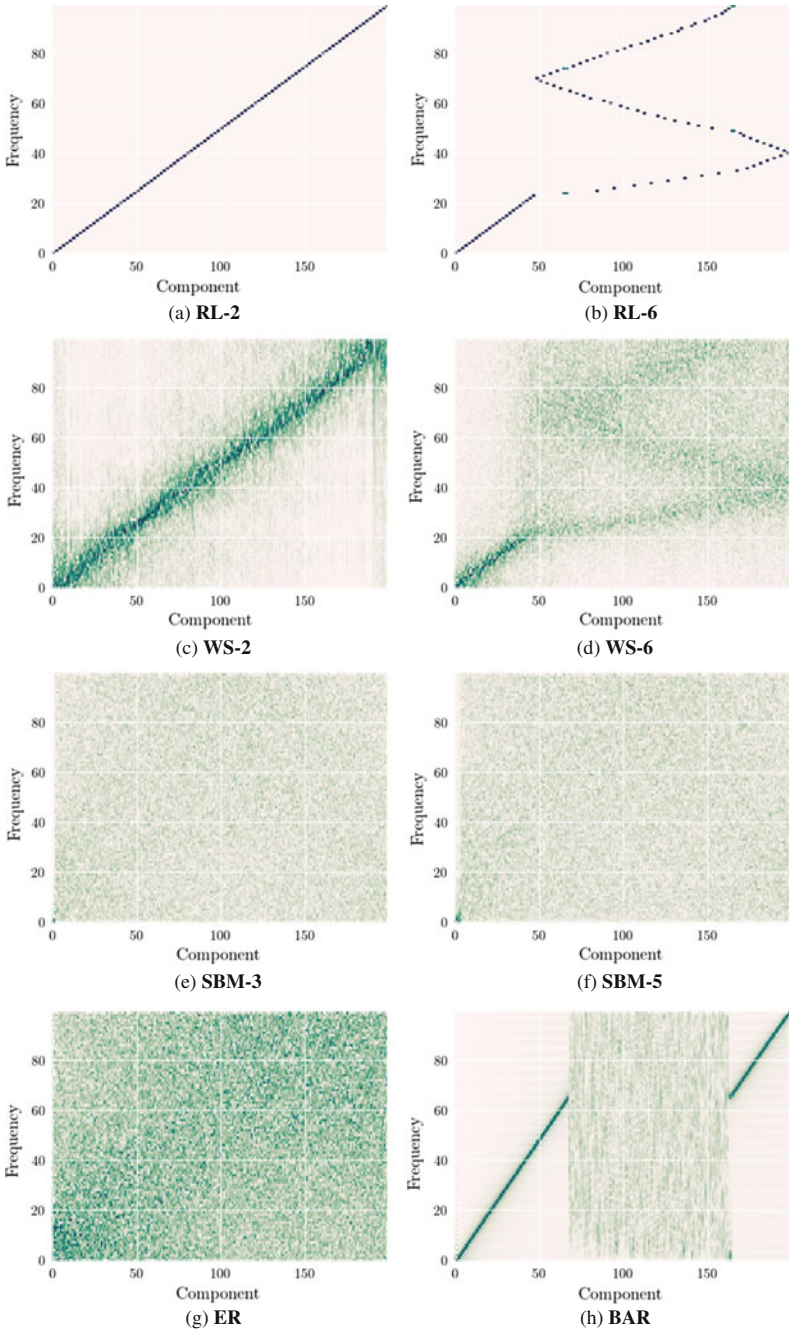
The aim of this part is to show how spectral analysis can be used to identify specific graph structures. As seen in Sect. 2.2, signals present specific shapes which can be linked with the graph structure. Characterizing these shapes using spectral analysis enables us to associate frequency patterns with graph topology.

Spectral analysis is performed using standard signal processing tools: for a given collection $\boldsymbol{X}$ of $n-1$ signals indexed by n vertices, the spectra $\boldsymbol{S}$ give the complex Fourier coefficients whose elements are obtained by applying the Fourier transform on each component of $\boldsymbol{X}$:

$$s_{kf} = \mathscr{F}\boldsymbol{X}^{(k)}(f) \tag{18}$$

estimated, for positive frequencies, on $F = \frac{n}{2} + 1$ bins, $\mathscr{F}$ being the Fourier transform, and $k \in \{1, \ldots, n-1\}$. From $\boldsymbol{S}$, the magnitudes $\boldsymbol{M}$ of each frequency f for each component read as: $m(k, f) = |s_{kf}|$. The matrix $\boldsymbol{M}$ is studied as a frequency-component map, exhibiting patterns in direct relation with the topology of the underlying graph.

Figure 12 shows the frequency-component patterns obtained for the graphs defined in Sect. 2.2. Each pattern highlights a specific graph structure. In Fig. 12a, b, patterns of regular k-lattices display single-frequency components, whose order depends on the value of k: when k is higher than 2, the sorting of components is no longer



(a) **RL-2** (b) **RL-6**

(c) **WS-2** (d) **WS-6**

(e) **SBM-3** (f) **SBM-5**

(g) **ER** (h) **BAR**

Fig. 12 Frequency patterns obtained on instances of graph defined in Sect. 2. Darker color means higher intensity

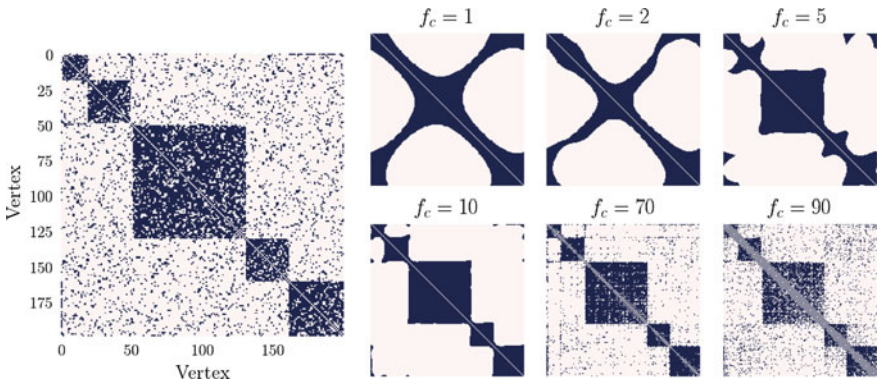
consistent with the increasing order of frequencies, as described in Sect. 2.2.2. Adding noise to the lattices, as done in the Watts–Strogatz instances in Fig. 12c, d, preserves the main frequency content, while adding many noise along the pattern. As described in Sect. 2.2.4, signals associated to graphs with communities are formed by plateaus corresponding to the communities on the first components, leading to the frequency pattern displayed in Fig. 12e, f with high magnitudes for low frequencies. All other parts of the map do not have any kind of structure, representing the structure inside and between communities, or more generally the structure of an instance of the Erdös–Rényi model, whose pattern is displayed in Fig. 12g. Interestingly enough, when the topology of the graph is a combination of several graph structures, as for the Barbell model, the associated patterns are visible, as in Fig. 12h.

Characterizing the structure of networks through frequency patterns has been proven of high interest in [23], where the structure of a time-evolving networks is decomposed using non-negative matrix techniques of the spectra, revealing the underlying structure over time.

4.2 Graph Filtering

The representation of a network as signals allows for taking advantage of all existing concepts in signal processing to perform filtering of a graph, by filtering the corresponding signals. In this basic setup, signals are passed through a low-pass filter with a given cut-off frequency f_c, and reconstructed using the robust inverse transformation defined in Sect. 3.

Figure 13 shows the obtained adjacency matrices for different values of f_c, for an instance of the stochastic block model (**SBM-3**). When only low frequencies are



(a) Original adjacency matrix. (b) Filtered adjacency matrices.

Fig. 13 Adjacency matrix obtained after low-pass filtering for different values of f_c, for an instance of the stochastic block model (**SBM-3**)

retained, the structure in cliques becomes visible, firstly approximately, then more distinctly. When the cut-off frequency increases, the noise inside and between communities is also reconstructed, until recovering the original graph. This example highlights the idea that denoising signals representing the graph acts as a surrogate processing to remove the noise the graph itself. This opens the way to more complicated operations such as decomposition, interpolation or sampling.

5 Conclusion

Signal processing techniques have become more and more standard tools to study real-world networks. In this chapter, a comprehensive framework has been developed to take advantage of these classical tools to analyze and process graphs, using an equivalent signal representation. Starting from the method of transformation from graph to signals proposed in [46], several extensions have been studied and implemented to study the obtained signals, and more particularly by introducing a robust inverse transformation, that is able to transform back the effects of classical signal processing tools of signals on the structure of the graph. The applications of this framework on the characterization of the structure of the graph, and the graph filtering using the signals, suggest that stronger connections can be derived to better exploit this dual representation. Application of this framework on time-evolving graphs [23] have for instance made possible the extraction of the relevant sub-structures over time, using matrix factorization techniques on the corresponding signals. The main interest of this work lies in its ability to benefit from the well-grounded field of signal processing to perform difficult operations on graphs, and in this way it paves the road to new approach for the analysis of real-world networks, using more sophisticated signal processing tools.

Acknowledgements Figures displayed in this article have been created using Python and the following libraries: matplotlib/seaborn [27, 56], networkx [20], numpy/scipy [53], and sklearn [42]. All the code is available at the following url: https://github.com/r-hamon/pygas.

This work was supported by the programs ARC 5 and ARC 6 of the région Rhône-Alpes, and the ANR projects Vél'Innov ANR-12-SOIN-0001-02 and GRAPHSIP ANR-14-CE27-0001.

References

1. B. Aguilar-San Juan, L. Guzmán-Vargas, Earthquake magnitude time series: scaling behavior of visibility networks. Eur. Phys. J. B **86**(11), 454 (2013)
2. D. Aldous, J. Fill, *Reversible Markov Chains and Random Walks on Graphs* (Berkeley, 2002)
3. A. Anis, A. Gadde, A. Ortega, Towards a sampling theorem for signals on arbitrary graphs, in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2014), pp. 3864–3868
4. M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)

5. I. Borg, P.J.F. Groenen, *Modern Multidimensional Scaling*, Springer Series in Statistics (Springer, New York, 2005)
6. A.S.L.O. Campanharo, M. Irmak Sirer, R. Dean Malmgren, F.M. Ramos, L.A.N. Amaral, Duality between time series and networks. PloS One **6**(8), e23378 (2011)
7. S. Chen, R. Varma, A. Sandryhaila, J. Kovacevic, Discrete signal processing on graphs: sampling theory. CoRR (2015), arXiv:1503.05432
8. J. Clifford Gower, Properties of Euclidean and non-Euclidean distance matrices. Linear Algebra Appl. **67**, 81–97 (1985)
9. Y. Dekel, J.R. Lee, N. Linial, Eigenvectors of random graphs: nodal domains. Random Struct. Algorithms **39**(1), 39–58 (2011)
10. X. Ding, T. Jiang et al., Spectral distributions of adjacency and Laplacian matrices of random graphs. Ann. Appl. Probab. **20**(6), 2086–2117 (2010)
11. J.F. Donges, J. Heitzig, R.V. Donner, J. Kurths, Analytical framework for recurrence network analysis of time series. Phys. Rev. E **85**(4) (2012)
12. R.V. Donner, Y. Zou, J.F. Donges, N. Marwan, J. Kurths, Recurrence networksa novel paradigm for nonlinear time series analysis. New J. Phys. **12**(3), 033025 (2010)
13. R.V. Donner, M. Small, J.F. Donges, N. Marwan, Y. Zou, R. Xiang, J. Kurths, Recurrence-based time series analysis by means of complex network methods. Int. J. Bifurc. Chaos **21**(04), 1019–1046 (2011)
14. R. Durrett, *Random Graph Dynamics*, vol. 200 (Cambridge University Press, Cambridge, 2007)
15. S. Fortunato, Community detection in graphs. Phys. Rep. **486**(3), 75–174 (2010). Bibtex: Fortunato 2010
16. S. Fortunato, D. Hric, Community detection in networks: a user guide. Phys. Rep. **659**, 1–44 (2016)
17. A. Gadde, A. Ortega, A probabilistic interpretation of sampling theory of graph signals (2015), arXiv:1503.06629
18. B. Girault, P. Gonçalves, E. Fleury, A. Mor, Semi-supervised learning for graph to signal mapping: a graph signal wiener filter interpretation, in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)* (Florence, Italy, 2014), pp. 1115–1119
19. R.M. Gray, Toeplitz and circulant matrices: a review. Commun. Inf. Theory **2**(3), 155–239 (2005)
20. A. Hagberg, P. Swart, D.S. Chult, Exploring network structure, dynamics, and function using networkx. Technical report (Los Alamos National Laboratory (LANL), Los Alamos, 2008)
21. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
22. R. Hamon, P. Borgnat, P. Flandrin, C. Robardet, Networks as signals, with an application to a bike sharing system, in *2013 Global Conference on Signal and Information Processing (GlobalSIP)* (IEEE, 2013), pp. 611–614
23. R. Hamon, P. Borgnat, P. Flandrin, C. Robardet, Extraction of temporal network structures from graph-based signals. IEEE Trans. Signal Inf. Process. Netw. **2**(2), 215–226 (2016)
24. R. Hamon, P. Borgnat, P. Flandrin, C. Robardet, Relabelling vertices according to the network structure by minimizing the cyclic bandwidth sum. J. Complex Netw. **4**(4), 534–560 (2016)
25. Y. Haraguchi, Y. Shimada, T. Ikeguchi, K. Aihara, Transformation from complex networks to time series using classical multidimensional scaling, in *2009 Artificial Neural Networks–ICANN* (Springer, Berlin, 2009), pp. 325–334
26. Z.-K. Huang, K.-W. Chau, A new image thresholding method based on Gaussian mixture model. Appl. Math. Comput. **205**(2), 899–907 (2008)
27. J.D. Hunter, Matplotlib: a 2D graphics environment. Comput. Sci. Eng. **9**(3), 90–95 (2007)
28. H. Jianxiu, Cyclic bandwidth sum of graphs. Appl. Math. A J. Chin. Univ. **16**(2), 115–121 (2001)
29. B. Karrer, M.E.J. Newman, Stochastic blockmodels and community structure in networks. Phys. Rev. E **83**(1), 016107 (2011)
30. T. Kato, *Perturbation Theory for Linear Operators*, Classics in Mathematics (Springer, Berlin, 1995)

31. L. Lacasa, B. Luque, F. Ballesteros, J. Luque, J.C. Nuno, From time series to complex networks: the visibility graph. Proc. Natl. Acad. Sci. **105**(13), 4972–4975 (2008)
32. X.F. Liu, K.T. Chi, M. Small, Complex network structure of musical compositions: algorithmic generation of appealing music. Phys. A Stat. Mech. Appl. **389**(1), 126–132 (2010)
33. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Sampling of graph signals with successive local aggregations (2015), arXiv:1504.04687
34. N. Marwan, J.F. Donges, Y. Zou, R.V. Donner, J. Kurths, Complex network approach for recurrence analysis of time series. Phys. Lett. A **373**(46), 4246–4254 (2009)
35. M. Meila, J. Shi, A random walks view of spectral segmentation, in *8th International Workshop on Artificial Intelligence and Statistics* (2001)
36. S.K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Trans. Signal Process. **60**(6), 2786–2799 (2012)
37. M. Newman, *Networks: An Introduction* (Oxford University Press Inc, New York, 2010)
38. H.Q. Nguyen, M.N. Do, Downsampling of signals on graphs via maximum spanning trees. IEEE Trans. Signal Process. **63**(1), 182–191 (2015)
39. A.M. Nuñez, L. Lacasa, J.P. Gomez, B. Luque, Visibility algorithms: a short review. INTECH Open Access Publisher (2012)
40. S. O'Rourke, V. Vu, K. Wang, Eigenvectors of random matrices: a survey. J. Comb. Theory Ser. A **144**, 361–442 (2016)
41. N. Otsu, A threshold selection method from gray-level histograms. Automatica **11**(285–296), 23–27 (1975)
42. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
43. A. Sakiyama, Y. Tanaka, Oversampled graph Laplacian matrix for graph filter banks. IEEE Trans. Signal Process. **62**(24), 6425–6437 (2014)
44. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs: frequency analysis. IEEE Trans. Signal Process. **62**(12), 3042–3054 (2014)
45. Y. Shimada, T. Kimura, T. Ikeguchi, Analysis of chaotic dynamics using measures of the complex network theory, in *2008 Artificial Neural Networks-ICANN* (Springer, Berlin, 2008), pp. 61–70
46. Y. Shimada, T. Ikeguchi, T. Shigehara, From networks to time series. Phys. Rev. Lett. **109**(15), 158701 (2012)
47. D.I. Shuman, M. Javad Faraji, P. Vandergheynst, A framework for multiscale transforms on graphs (2013), arXiv:1308.4942
48. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
49. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. (2015)
50. N. Tremblay, P. Borgnat, Graph wavelets for multiscale community mining. IEEE Trans. Signal Process. **62**(20), 5227–5239 (2014)
51. N. Tremblay, P. Borgnat, Subgraph-based filterbanks for graph signals. IEEE Trans. Signal Process. **64**(15), 3827–3840 (2016)
52. M. Tsitsvero, S. Barbarossa, On the degrees of freedom of signals on graphs, in *23rd European Signal Processing Conference (EUSIPCO)* (Nice, France, 2015), p. 2015
53. S. van der Walt, S.C. Colbert, G. Varoquaux, The numpy array: a structure for efficient numerical computation. Comput. Sci. Eng. **13**(2), 22–30 (2011)
54. P. Van Mieghem, *Graph Spectra for Complex Networks* (Cambridge University, Cambridge, 2011)
55. U. Von Luxburg, A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)

56. M. Waskom, O. Botvinnik, D. O'Kane, P. Hobson, S. Lukauskas, D.C. Gemperline, T. Augspurger, Y. Halchenko, J.B. Cole, J. Warmenhoven, J. de Ruiter, C. Pye, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, P. Bachant, M. Martin, K. Meyer, A. Miles, Y. Ram, T. Yarkoni, M. Williams, C. Evans, C. Fitzgerald, Brian, C. Fonnesbeck, A. Lee, A. Qalieh, mwaskom/seaborn: v0.8.1 (2017)
57. D.J. Watts, S.H. Strogatz, Collective dynamics of small-world networks. Nature **393**(6684), 440–442 (1998)
58. T. Weng, Y. Zhao, M. Small, D. Huang, Time-series analysis of networks: exploring the structure with random walks. Phys. Rev. E **90**(2), 022804 (2014)
59. Y. Yang, H. Yang, Complex network-based time series analysis. Phys. A Stat. Mech. Appl. **387**(5–6), 1381–1386 (2008)
60. W.W. Zachary, An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**(4), 452–473 (1977)
61. J. Zhang, M. Small, Complex network from pseudoperiodic time series: topology versus dynamics. Phys. Rev. Lett. **96**(23), 238701 (2006)

The Spectral Graph Wavelet Transform: Fundamental Theory and Fast Computation



David K. Hammond, Pierre Vandergheynst and Rémi Gribonval

Abstract The spectral graph wavelet transform (SGWT) defines wavelet transforms appropriate for data defined on the vertices of a weighted graph. Weighted graphs provide an extremely flexible way to model the data domain for a large number of important applications (such as data defined on vertices of social networks, transportation networks, brain connectivity networks, point clouds, or irregularly sampled grids). The SGWT is based on the spectral decomposition of the $N \times N$ graph Laplacian matrix $\mathscr{L}$, where N is the number of vertices of the weighted graph. Its construction is specified by designing a real-valued function g which acts as a bandpass filter on the spectrum of $\mathscr{L}$, and is analogous to the Fourier transform of the "mother wavelet" for the continuous wavelet transform. The wavelet operators at scale s are then specified by $T_g^s = g(s\mathscr{L})$, and provide a mapping from the input data $f \in \mathbb{R}^N$ to the wavelet coefficients at scale s. The individual wavelets $\psi_{s,n}$ centered at vertex n, for scale s, are recovered by localizing these operators by applying them to a delta impulse, i.e. $\psi_{s,n} = T_g^s \delta_n$. The wavelet scales may be discretized to give a graph wavelet transform producing a finite number of coefficients. In this work we also describe a fast algorithm, based on Chebyshev polynomial approximation, which allows computation of the SGWT without needing to compute the full set of eigenvalues and eigenvectors of $\mathscr{L}$.

D. K. Hammond (✉)
Oregon Institute of Technology - Portland Metro, Wilsonville, OR, USA
e-mail: david.hammond@oit.edu

P. Vandergheynst
Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland
e-mail: pierre.vandergheynst@epfl.ch

R. Gribonval
Univ Rennes, Inria, CNRS, IRISA, Rennes, France
e-mail: remi.gribonval@inria.fr

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_3

1 Introduction

Nearly all interesting scientific and engineering problems involve analyzing data. In many cases, data can be described as a real valued function defined on some domain. For example, data sets such as audio recordings, digital photographs, and digital videos may be represented as real valued functions defined on a one, two, or three dimensional Euclidean domains, respectively. Similarly, discrete time-series data may be modeled as a real valued function whose domain is a subset of the integers. As these examples illustrate, many common types of data are defined on domains which are regularly sampled subsets of some Euclidean space. A huge body of signal processing and analysis algorithms have been developed for signals that are defined on such regularly sampled Euclidean domains.

However, a large number of interesting data sets are defined on irregular domains that do not correspond to regularly sampled Euclidean domains. Examples of this included data defined on networks, or on point clouds, or at irregularly sampled points of Euclidean domains. Many topologically complex data domains can be profitably modeled as weighted graphs: i.e. sets of vertices that are connected by edges that each have a non-negative weight, or connection "strength" defined. For some applications, the underlying graph structure for the data domain may be clear. This would be the case for example for analyzing data (such as income, preference for something, or any other scalar value) defined for individuals on a social network, where the underlying graph structure models the relationship strength between individuals. For data defined on point clouds or for irregularly sampled data, generating the underlying graph structure may be calculated in a number of ways based on the proximity of the data points, such as for example using the k-nearest neighbors.

Many signal processing and analysis methods employ some type of transform of the original data, where processing or analysis is performed on the coefficients of the transformed data rather than on the signal in its original domain. A large class of such transforms are linear, where the coefficients of the transform are given by taking inner products of the original signal with some set of transform vectors. For signals defined on regular Euclidean domains, commonly used examples include the Fourier transform, the discrete cosine transform, windowed Fourier transforms, and a large number of different wavelet transforms. Signal transforms are useful as certain analysis or processing problems may be easier to express, or more powerful, when done in the coefficient domain than in the original signal domain. As a trivial example, the problem of estimating the power spectrum of a discrete time signal averaged over a specified frequency range takes some time to write down in the time domain, but is very easy to express (as an average of the magnitudes of a subset of Fourier coefficients) given the Fourier transform of the signal.

Wavelet transforms in particular have proven to be very effective for a wide variety of signal processing and analysis problems. Wavelet transforms have the property that the transform vectors (in this case called wavelets) are well localized in both space and frequency domains, and are self similar - i.e. related to each other by translation and dilation. Wavelets may be designed so that they provide a sparse

representation of signals that consist of relatively smooth regions separated by localized singularities, such as edges for 2d images, or localized jump discontinuities for 1d signals. Much of the power of many wavelet-based signal processing algorithms arises from exploiting this signal sparsity. Since their initial introduction in the 1980's [1–5], wavelet transforms have been very successfully employed for a wide range of signal and image processing applications, including denoising, [6–11], compression [12–16], and inverse problems such as deconvolution [17–22]. We have included only a sampling of the enormous body of literature in the references given above.

The demonstrated effectiveness of wavelet transforms, combined with the growing desire to process data defined on non-Euclidean domains, motivates the adaptation of the wavelet transform to data defined on weighted graphs. The work described in this chapter details one such approach for constructing multiscale Wavelet transforms for data defined on the vertices of weighted graphs. Our approach assumes that all relevant information about the weighted graph is encoded in the symmetric adjacency matrix A, i.e. no other information about the meaning of the vertices or relationships between them apart from what is stored in the Adjacency matrix is used. Accordingly, for any specific application problem the design of the underlying edge weights for the graph forms a crucial part of determining the overall wavelet transformation. The design choices for constructing the graph weights exactly correspond to modeling the underlying data domain, and thus optimal choices may be highly application dependent. In this chapter we illustrate relatively simple examples of computing a appropriate weighted graph for data defined on point clouds, and on irregularly sampled grids.

In general, one may expect modeling the data domain by a weighted graph to be useful whenever the relationships one may describe between vertices with a weighted graph interact with the underlying process which generated the data. For example, if one were analyzing rates of some disease among different cities, it may be reasonable to assume that infection could be propagated by individuals travelling from one city to another. In this case, using a weighted graph representing a transportation network between the different cities may be helpful. Similarly, if one were analyzing data indicating individuals opinions (favorable or unfavorable) of some particular political candidate, it is reasonable to assume that individuals opinions are affected by discussions with their friends. Accordingly, knowledge of a weighted graph representing acquaintance or friendship relationships between individuals may be useful for analyzing such data.

Classical wavelet analysis is based on the idea of taking a single "mother wavelet" function, and generating an entire set of wavelet atoms by translating and dilating the mother wavelet. Wavelet coefficients for a given signal are then produced by taking the inner products of the signal with these wavelet atoms at different scales and spatial locations. The success of this classical construction is based on the ability to perform arbitrary dilation and translation operations. On an arbitrary weighted graph, however, it is not possible to define dilation and translation as may be done on Euclidean spaces such as the real line.

The construction described in this chapter resolves this by using the spectral decomposition (i.e. eigenvectors and eigenvalues) of the graph Laplacian matrix $\mathscr{L}$.

Spectral graph theory [23] enables the definition of a Fourier transform for data defined on the vertices of weighted graphs, where the graph Fourier coefficients are given by the inner product of the signal with the eigenvectors of the graph Laplacian. The Spectral Graph Wavelet Transform (SGWT) described here is obtained by considering the mapping from data to coefficients for the classical continuous wavelet transform in the Fourier domain, and constructing the analogous operations using the graph Fourier transform. The SGWT design requires specifying a real-valued kernel function g. This kernel function is used to define the wavelet operators at scale s (i.e. the mappings from the signal to the wavelet coefficients at scale s) as $T_g^s = g(s\mathscr{L})$. The wavelet coefficients at scale s for an input signal $f(t)$ are then given by $T_g^s f$. The individual graph wavelets themselves are obtained by localizing these wavelet operators by applying them to a delta impulse at a single vertex. We employ the notation $\psi_{s,m}$ to denote the wavelet at scale s centered at vertex m, the previous notion implies that $\psi_{s,m} = T_g^s \delta_m$ where δ_m is a signal with zero values at every vertex except m, and unit value at vertex m. The SGWT coefficients are also the inner products of the original data with these wavelets $\psi_{s,m}$.

This chapter describes the basic theory of the SGWT, and gives implementation details and example images illustrating the wavelets and properties of the overall transform. We show that the SGWT (without discretizing the scale parameter) is analogous to the classical continuous wavelet transform and, subject to an admissibility condition on the kernel function g, may be formally inverted using a similar integral formula. A discrete transform may be obtained by sampling the scale parameter at a discrete set of values, giving a finite number of coefficients organized in distinct wavelet subbands. In this case the SGWT is an overcomplete transform, and we describe how to calculate the corresponding frame bounds.

As the SGWT is defined using a the graph Fourier transform, straightforward computation of the transform requires computing the full set of eigenvectors and eigenvalues of the graph Laplacian $\mathscr{L}$. This limitation would render computing the SGWT infeasible for graphs larger than several thousand vertices, which would severely limit its applicability. In this chapter we describe a method for computing the SGWT based on Chebyshev polynomial approximation of the rescaled kernels $g(s\lambda)$, which does not require explicitly computing the full set of eigenvectors of the graph Laplacian matrix. In particular, this polynomial approximation approach uses $\mathscr{L}$ only through matrix-vector multiplication, and is thus especially computationally efficient for sparse graphs, where only a small number of elements of $\mathscr{L}$ are nonzero. The discrete SGWT may be inverted using the pseudoinverse. We show that this may be done by conjugate-gradients, in a way that is compatible with the Chebyshev polynomial approximation scheme for applying the forward transform.

1.1 Related Work

Other attempts at defining wavelet transforms on graphs have been developed that do not employ spectral graph theory. One approach used by Crovella and Kolaczyk [24] for analyzing computer network traffic was based on the n-hop dis-

tance, where wavelets were defined such that the value of the wavelet at vertex m that was centered at vertex n depended on the n-hop distance from vertex m to n. These functions were chosen so that the wavelets were zero mean. These wavelets were constructed for binary graphs (i.e. making no use of edge weights), additionally no study of the invertibility of the resulting transform was made. These wavelets were used by Smalter et al. [25] as features helping to distinguish chemical structures, as part of a machine learning approach for virtual drug screening.

In [26], Jansen et al. developed a lifting-based approach for multiscale representation of data defined on graphs. This lifting procedure is based on using the weighted average of each vertices neighbors for the lifting prediction step, where the weightings are based on a set of distances assigned to each edge (playing the part of reciprocals of edge weights), in their paper these edge distances were derived from original Euclidean distances for graphs that arise from irregular sampling of Euclidean space. In contrast with the methods described in this chapter, this lifting scheme is defined directly in the vertex domain, and does not employ spectral graph theory.

Several works have considered wavelet transforms for data defined on trees (i.e. graphs with no loops). These include [27], which developed an adaptation of the Haar wavelet transform appropriate for data defined on rooted binary trees. The treelet transform [28] extended this, including automatic construction of trees for multivariate data.

The "Diffusion Wavelets" of Maggioni and Coifman [29] constructs a wavelet transform based on compressed representations of dyadic powers of a diffusion operator T, which may be flexibly specified. The diffusion wavelets construction involves repeated application of the diffusion operator T, somewhat analogously to how our construction is parametrized by the choice of the graph Laplacian operator $\mathscr{L}$. A key difference between the Diffusion Wavelets and the Spectral Graph Wavelets described here is that the Diffusion Wavelets are designed to be orthonormal. The Diffusion Wavelets approach is based on first identifying the approximation spaces produced by differences of dyadic powers of the operator T; wavelets are produced by locally orthogonalizing these approximation spaces. Our approach is conceptually simpler, and yields an overcomplete representation rather than an orthogonal transform.

The "Diffusion polynomial frames" developed by Maggioni and Mhaskar [30] builds multiscale transforms in a more general quasi-metric measure space setting using polynomials of a differential operator, in a manner that is closely related to our Chebyshev polynomial approximation for computing the SGWT. Geller and Mayeli [31] construct wavelets on differentiable manifolds employing scaling defined by an operator of the form tLe^{-tL}, where L is the manifold Laplace–Beltrami operator. Wavelets are obtained by localizing this operator by applying it to a delta impulse, similar to our theory. However, their work is not directly comparable to ours as it is constructed for functions defined on smooth manifolds.

The work described in this chapter was originally published in [32]. For completeness, we note that since its original publication the SGWT and the related polynomial approximation scheme has been used by a number of authors. We provide here

references to a sampling of these applications, include learning dictionaries for signal representation on graphs [33], analysis of cortical thickness measurements [34], shape analysis and surface alignment [35], multiscale community detection [36], 3D mesh compression [37].

2 Classical Continuous Wavelet Transform

As the design of the SGWT is based on examining the the classical Continuous wavelet transform (CWT) in the Fourier domain, we first give an overview of the CWT appropriate for representing $L^2(\mathbb{R})$, i.e. the space of square-integrable signals on the real line.

The CWT is generated by choosing a single "mother" wavelet $\psi(t)$, and then forming a continuous family of wavelets by translating and dilating the mother wavelet. Specifically, for translation by a and dilation by factor $s > 0$, we have the wavelet

$$\psi_{s,a}(t) = \frac{1}{s}\psi\left(\frac{t-a}{s}\right) \tag{1}$$

The factor $\frac{1}{s}$ in front was chosen so that the wavelets all have the same L_1 norm, i.e. $\int_{-\infty}^{\infty} |\psi_{s,a}(t)|dt = \int_{-\infty}^{\infty} |\psi(t)|dt$. The wavelet coefficients are given by the inner products of these wavelets with the signal $f(t)$, as

$$W_f(s,a) = \int_{-\infty}^{\infty} \frac{1}{s}\psi^*\left(\frac{t-a}{s}\right) f(t)dt \tag{2}$$

This expression defines the mapping from the original signal $f(t)$ to the set of wavelet coefficients $W_f(s,a)$. An interesting feature of the CWT is that a single variable function $f(t)$ is represented by the coefficients $W_f(s,a)$ which depend on two parameters s and a. We say the transform may be inverted if it is possible to recover the function $f(t)$ from knowledge of the coefficients $W_f(s,a)$. It has been shown [1] that this is possible if the mother wavelet $\psi(t)$ satisfies the admissibility condition

$$\int_0^{\infty} \frac{|\hat{\psi}(\omega)|^2}{\omega} d\omega = C_\psi < \infty \tag{3}$$

One consequence of the admissibility condition is that for continuously differentiable $\psi(t)$, it must hold that $\hat{\psi}(0) = \int \psi(t)dt = 0$, so $\psi(t)$ must have zero mean. If $\psi(t)$ is admissible, the CWT may be formally inverted as

$$f(t) = \frac{1}{C_\psi}\int_0^{\infty}\int_{-\infty}^{\infty} W_f(s,a)\psi_{s,a}(t)\frac{dads}{s}. \tag{4}$$

Directly forming the analogue of Eq. 1 on weighted graphs is problematic, as it is unclear how to implement arbitrary dilation or translation on an arbitrary weighted graph. We will show next that for the CWT, scaling can be defined in the Fourier domain, which does provide an expression we can extend to weighted graphs. We first consider the CWT for discretized set of scale values, where the translation parameter is left continuous. For each value s, we let T^s represent the mapping from signal $f(t)$ to wavelet coefficients $W_f(s,a)$, so that $(T^s f)(a) = W_f(s,a)$ where a is considered as the independent variable.

For convenience in the following, define the scaled, time-reversed wavelet $\overline{\psi}_s$ via

$$\overline{\psi}_s(t) = \frac{1}{s}\psi^*\left(\frac{-t}{s}\right). \tag{5}$$

One may see then that the operator T^s acts on any signal by convolution with $\overline{\psi}_s$. Specifically, we have

$$\begin{aligned}(T^s f)(a) &= \int_{-\infty}^{\infty} \frac{1}{s}\psi^*\left(\frac{t-a}{s}\right) f(t)dt \\ &= \int_{-\infty}^{\infty} \overline{\psi}_s(a-t) f(t)dt \\ &= (\overline{\psi}_s \star f)(a)\end{aligned} \tag{6}$$

By taking the Fourier transform and applying the convolution theorem we see that

$$\widehat{T^s f}(\omega) = \widehat{\overline{\psi}}_s(\omega)\widehat{f}(\omega) \tag{7}$$

Using the properties of the Fourier transform and Eq. (5) shows that

$$\widehat{\overline{\psi}}_s(\omega) = \widehat{\psi^*}(s\omega) \tag{8}$$

Inserting Eq. 8 into Eq. 7 and inverting the Fourier transform gives

$$(T^s f)(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\omega t}\widehat{\psi}^*(s\omega)\widehat{f}(\omega)d\omega \tag{9}$$

Critically, s appears above only in the argument of $\widehat{\psi}^*$, which is defined in the Fourier domain rather than the original signal domain. We see that the operator T^s mapping $f(t)$ to the set of wavelet coefficients at scale s acts on $f(t)$ by multiplying its Fourier transform by a bandpass filter function $\widehat{\psi}^*(s\omega)$ which is scaled (in the Fourier domain) by s. Equation (9) will form the basis for us to later define the SGWT, where we will replace the Fourier transform by the graph Fourier transform.

We may express the individual wavelets by applying the operator T^s to a translated delta impulse function $\delta_a(t) = \delta(t-a)$. From Eq. (6) it follows that

$$(T^s \delta_a)(t) = \frac{1}{s} \psi^* \left(\frac{t-a}{s} \right), \tag{10}$$

which for even and real-valued $\psi(t)$ simplifies to $(T^s \delta_a)(t) = \psi_{a,s}(t)$.

3 Spectral Graph Theory

In this section we introduce the tools from Spectral Graph theory needed to define the graph Fourier transform. This will provide the ability to define scaling of an operator, in the spectral domain, which is at the core of the SGWT construction. We first fix our notation for weighted graphs.

3.1 Notation for Weighted Graphs

A weighted graph G consists of a finite vertex set V, a set of edges E (which is a subset of the set of all unordered pairs of vertices), and a non-negative valued weight function $w : E \to \mathbb{R}$ which gives a weight associated with each edge. We let $N = |V|$ denote the number of vertices. A finite weighted graph may also be unambiguously described by an $N \times N$ weighted adjacency matrix A, where $A_{i,j}$ equals zero if the the edge $(i, j) \notin E$, and $A_{i,j} = w((i, j))$ if the edge $(i, j) \in E$. We consider only symmetric (i.e. undirected) graphs.

The degree of each vertex is the sum of all the edge weights of edges incident to that vertex. In terms of the adjacency matrix, may write $d(m) = \sum_n A_{m,n}$ for the degree of vertex m. The diagonal degree matrix D is defined by

$$D_{i,j} = \begin{cases} d(i) & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{11}$$

Once a specific numbering of the vertices has been fixed, any function $f : V \to \mathbb{R}$ defined on the vertices can be naturally associated with a vector in $f \in \mathbb{R}^N$, where f_i is simply the value of the function on vertex i. We will also denote $f(i)$ for the value on vertex i.

In this work we use the non-normalized graph Laplacian operator $\mathscr{L}$, defined by $\mathscr{L} = D - A$. For any $f \in \mathbb{R}^N$, it is straightforward to show that

$$(\mathscr{L} f)(m) = \sum_{m \sim n} A_{m,n} \cdot (f(m) - f(n)) \tag{12}$$

where by $m \sim n$ we mean that the sum is taken over all vertices n which are connected to vertex m. This expression shows that the graph Laplacian operator applied to any

function gives a weighted difference of the function values, summed over all edges incident to a given vertex.

Another form of the graph Laplacian that is commonly used elsewhere in the literature is the normalized form, given by

$$\mathscr{L}^{norm} = D^{-1/2}\mathscr{L}D^{-1/2} = I - D^{-1/2}AD^{-1/2} \tag{13}$$

The eigenvalues of this $\mathscr{L}^{norm}$ all lie in the interval $[0, 2]$. We note that the normalized and non-normalized Laplacian matrices are not similar matrices, and their eigenvectors are different. The entire SGWT machinery could be defined using either form of the graph Laplacian, which would produce two different transforms. We will use the non-normalized form exclusively in the remainder of this work.

For the non-normalized Laplacian $\mathscr{L}$ constructed for a graph that corresponds to a regularly sampled grid, we note that $\mathscr{L}$ is proportional (with a difference in sign) to a standard finite difference approximation of the continuous Laplacian operator. For example, consider a regular two-dimensional grid with unit weights on the edges, where $v_{m,n}$ represents the vertex at index position (m, n) on the grid. Using these two-dimensional indices, one sees that for a function $f = f_{m,n}$ defined on the vertices, applying $\mathscr{L}$ yields (for (m, n) away from the boundary of the grid)

$$(\mathscr{L}f)_{m,n} = 4f_{m,n} - f_{m+1,n} - f_{m-1,n} - f_{m,n+1} - f_{m,n-1}. \tag{14}$$

Aside from a missing factor of h^2, where h is the mesh spacing, this is the standard 5-point stencil for computing $-\nabla^2 f$.

3.2 Graph Fourier Transform

The standard Fourier transform on the real line is

$$\widehat{f}(\omega) = \int e^{-it\omega} f(t)dt, \tag{15}$$

with inverse transform

$$f(t) = \frac{1}{2\pi}\int \widehat{f}(\omega)e^{i\omega t}d\omega. \tag{16}$$

The complex exponentials $e^{i\omega t}$ are eigenfunctions of the one-dimensional Laplacian $\frac{d}{dx^2}$. We may view the forward transform as computing the Fourier coefficient $\widehat{f}(\omega)$ as the inner product of the signal $f(t)$ with a Laplacian eigenfunction. Similarly, the inverse transform may be viewed as expanding the signal $f(t)$ as a weighted sum of Laplacian eigenfunctions.

The graph Fourier transform is obtained by analogy from the previous statements, by replacing the continuous Laplacian $\frac{d}{dx^2}$ by the graph Laplacian $\mathscr{L}$. The matrix

$\mathscr{L}$ is symmetric, and so has a set of orthonormal eigenvectors which span $\mathbb{R}^N$. We write these as χ_ℓ for $0 \leq \ell \leq N-1$. The corresponding eigenvalues λ_ℓ satisfy $\mathscr{L}\chi_\ell = \lambda_\ell \chi_\ell$. The symmetry of $\mathscr{L}$ implies that these eigenvalues are real, so we may organize them in ascending order. In addition, for the graph Laplacian $\mathscr{L}$, it holds that the eigenvalues are all non-negative, the smallest eigenvalue is 0, and the multiplicity of the 0 eigenvalue is equal to the number of connected components of the weighted graph G [23]. Assuming that G is connected, the eigenvalues λ_ℓ satisfy

$$0 = \lambda_0 < \lambda_1 \leq \lambda_2 \cdots \leq \lambda_{N-1} \tag{17}$$

We now define the graph Fourier transform. For any signal $f \in \mathbb{R}^N$, the graph Fourier transform $\widehat{f}$ is given by

$$\widehat{f}(\ell) = \langle \chi_\ell, f \rangle = \sum_{n=1}^{N} \chi_\ell^*(n) f(n). \tag{18}$$

The inverse graph Fourier transform expresses the original signal f as an expansion using the graph Fourier coefficients $\widehat{f}$ as

$$f(n) = \sum_{\ell=0}^{N-1} \widehat{f}(\ell) \chi_\ell(n) \tag{19}$$

The validity of the inverse Fourier transform is a straightforward consequence of the orthonormality of the eigenvectors χ_ℓ. It can similarly be shown that the graph Fourier coefficients satisfy the Parseval relation, i.e. for any two signals $f, g \in \mathbb{R}^N$ one has

$$\langle f, g \rangle = \left\langle \widehat{f}, \widehat{g} \right\rangle. \tag{20}$$

4 Spectral Graph Wavelets

Equipped with the graph Fourier transform, we are now prepared to describe the Spectral Graph Wavelet Transform. As alluded to earlier, specification of the SGWT requires fixing a non-negative real-valued kernel function g, which behaves as a band-pass filter and is analogous to the Fourier transform of the mother wavelet $\widehat{\psi}^*$ from Eq. (9). We will require that $g(0) = 0$ and that $\lim_{\lambda \to \infty} g(\lambda) = 0$. Specific choices for the kernel g will be discussed later.

4.1 Wavelets

The wavelet operators producing the SGWT coefficients at each scale are obtained as rescaled kernel functions of the graph Laplacian operator. One may define a function

of a self-adjoint operator by using the continuous functional calculus [38] based on the spectral representation of the operator. For the finite dimensional graph Laplacian, this is afforded by the eigenvectors and eigenvalues of the Laplacian matrix $\mathscr{L}$. Specifically, we set the wavelet operator by $T_g = g(\mathscr{L})$. T_g is a mapping from $\mathbb{R}^N$ to $\mathbb{R}^N$, and $T_g f$ gives the wavelet coefficients for the signal f at unit scale ($s = 1$). This operator is defined by its action on the eigenvectors χ_ℓ, specifically as

$$T_g \chi_\ell = g(\lambda_\ell)\chi_\ell \tag{21}$$

This implies that for any graph signal f, the operator T_g acts on f by modulating each of its graph Fourier coefficients, according to

$$\widehat{T_g f}(\ell) = g(\lambda_\ell)\hat{f}(\ell) \tag{22}$$

Applying the inverse Fourier transform then shows

$$(T_g f)(m) = \sum_{\ell=0}^{N-1} g(\lambda_\ell)\hat{f}(\ell)\chi_\ell(m) \tag{23}$$

This relation should be compared with Eq. (9) describing the mapping from signal to wavelet coefficients for the Continuous Wavelet Transform.

We next define T_g^s, the wavelet operator at scale s, as $T_g^s = g(s\mathscr{L})$. The crucial point enabling this definition of scaling is that while the original spatial domain (set of vertices) is discrete, the domain of the kernel function $g(\lambda)$ is continuous, which enables proper definition of T_g^s, for any $s > 0$.

The individual wavelets are obtained by localizing these operators by applying them to δ_n, where $\delta_n \in \mathbb{R}^N$ is the signal with a 1 on vertex n and zeros elsewhere. This reads as

$$\psi_{s,n} = T_g^s \delta_n, \tag{24}$$

so that $\psi_{s,n}$ is the Spectral Graph Wavelet at scale s, centered on vertex n. We now observe that

$$\widehat{\delta_n}(\ell) = \sum_{m=1}^{N} \chi_\ell^*(m)\delta_n(m) = \chi_\ell^*(n). \tag{25}$$

Using this with Eq. (23) then implies that

$$\psi_{s,n}(m) = \sum_{\ell=0}^{N-1} g(s\lambda_\ell)\chi_\ell^*(n)\chi_\ell(m) \tag{26}$$

The wavelet coefficients $W_f(s, n)$ may then be considered as the inner products of f with the wavelet $\psi_{s,n}$, i.e. via

$$W_f(s,n) = \left\langle \psi_{s,n}, f \right\rangle. \tag{27}$$

Equivalently, we have $W_f(s,n)$ as the value of $(T_g^s f)_n$. Using Eq. (23) this may be expanded as

$$W_f(s,n) = \left(T_g^s f\right)(n) = \sum_{\ell=0}^{N-1} g(s\lambda_\ell)\hat{f}(\ell)\chi_\ell(n) \tag{28}$$

4.2 Scaling Functions

As the wavelet kernel g satisfies $g(0) = 0$, the wavelets $\psi_{s,n}$ are all orthogonal to the eigenvector χ_0, and are close to orthogonal to χ_ℓ for eigenvectors where λ_ℓ is close to zero. In order to stably represent the lower frequency content of signals, it is helpful to introduce a set of spectral graph scaling functions. These are defined in analogy with scaling functions for the classical wavelet transform, which are needed to represent low frequency content of signals when the scale parameter is not allowed to become arbitrarily large. We define the spectral graph scaling functions similarly to the wavelets, using a non-negative valued scaling function kernel $h(\lambda)$ which may be viewed as a low-pass filter. The scaling function kernel h satisfies $h(0) > 0$ and $\lim_{\lambda\to\infty} h(\lambda) = 0$, from which the scaling function operator $T_h = h(\mathscr{L})$ is defined. The scaling functions centered at vertex n are given by $\phi_n = T_h\delta_n$, and the scaling function coefficients are given by $S_f(n) = \langle \phi_n, f \rangle$.

In Fig. 1 we show the graphs of representative scaled wavelet kernels and the scaling function kernel, for a chosen set of discrete scale values s_j. Details of the choices for h and g are deferred until later. We will show later in Sect. 5.2 that stable recovery of f from its wavelet and scaling function coefficients is possible if the function $G(\lambda) = h(\lambda)^2 + \sum_j g(s_j\lambda)^2$ is nonzero for all λ in the spectrum of $\mathscr{L}$. Clearly, as each of the scaled wavelet kernels $g(s_j\lambda)$ approach zero as $\lambda \to 0$, this condition can only hold if the scaling function $h(\lambda)$ satisfies $h(0) > 0$.

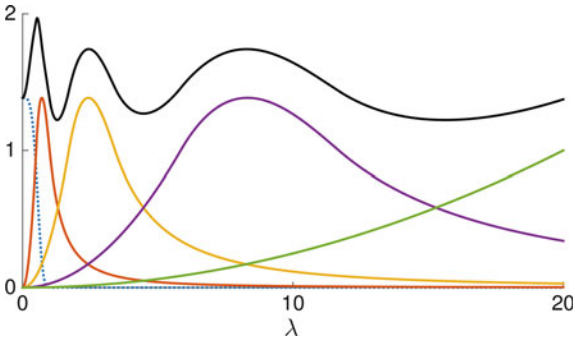


Fig. 1 Scaling function $h(\lambda)$ (dotted blue curve), wavelet generating kernels $g(t_j\lambda)$ (green, magenta, yellow, orange curves), and sum of squares G (black curve), for $J = 4$ scales, $\lambda_{max} = 20$. Scale values are $t_1 = 2.0$, $t_2 = 0.5848$, $t_3 = 0.171$, $t_4 = 0.05$. Details for the functional form of h and g are in Sect. 7

The scaling functions defined here are used to represent low frequency content of the signal f. We note that the kernels h and g used here do not satisfy the two-scale relation as in classical orthogonal wavelet design [3]. We thus have much freedom in choosing the form of h, provided that the resulting G does not become close to zero over the spectrum of $\mathscr{L}$.

5 Properties of the SGWT

We next describe several properties of the spectral graph wavelet transform, including the inverse of the continuous transform, the localization properties in the small-scale limit, and the frame bounds for the scale-discretized transform.

5.1 Inverse for Continuous SGWT

For any type of signal transform to be useful for signal processing (rather than only signal analysis), one must be able to invert the transform, i.e. to reconstruct a signal corresponding to a given set of transform coefficients. The continuous SGWT (i.e. where the scale parameter is not discretized) admits an inverse formula that has a very similar form to the inverse expression for the continuous wavelet transform in Eq. (4).

Each wavelet coefficient $W_f(s, n)$ may be viewed as measuring the "amount" of the wavelet $\psi_{s,n}$ present in the original signal f. The continuous SGWT inverse uses these measurements to reconstruct the original signal, with a weighting ds/s. As mentioned previously, however, as all of the wavelets are orthogonal to the eigenvector χ_0, the subspace spanned by χ_0 must be handled separately.

Lemma 1 *Let $f \in \mathbb{R}^N$ be a signal, and let $f^\#$ be the projection of f onto the orthogonal complement of the span of χ_0, i.e. $f^\# = f - \langle \chi_0, f \rangle \chi_0$. Let g be a kernel function satisfying $g(0) = 0$, and the admissibility condition*

$$\int_0^\infty \frac{g^2(x)}{x} dx = C_g < \infty. \tag{29}$$

Then the continuous reconstruction formula holds:

$$\frac{1}{C_g} \sum_{n=1}^{N} \int_0^\infty W_f(s, n) \psi_{s,n}(m) \frac{ds}{s} = f^\#(m) \tag{30}$$

The complete reconstruction of f is then given by $f = f^\# + \hat{f}(0)\chi_0$.

Proof We first expand the left side of the above, using Eqs. (26) and (28) to write $\psi_{t,n}$ and $W_f(t,n)$ in terms of the Laplacian eigenvectors χ_ℓ. This gives

$$\frac{1}{C_g}\int_0^\infty \frac{1}{s}\sum_n \left(\sum_\ell g(s\lambda_\ell)\chi_\ell(n)\hat{f}(\ell)\sum_{\ell'} g(s\lambda_{\ell'})\chi^*_{\ell'}(n)\chi_{\ell'}(m)\right) ds \tag{31}$$

$$= \frac{1}{C_g}\int_0^\infty \frac{1}{s}\left(\sum_{\ell,\ell'} g(s\lambda_{\ell'})g(s\lambda_\ell)\hat{f}(\ell)\chi_{\ell'}(m)\sum_n \chi^*_{\ell'}(n)\chi_\ell(n)\right) ds \tag{32}$$

We have that $\sum_n \chi^*_{\ell'}(n)\chi_\ell(n) = \delta_{\ell,\ell'}$, applying this and summing over ℓ' gives

$$= \frac{1}{C_g}\sum_\ell \left(\int_0^\infty \frac{g^2(s\lambda_\ell)}{s} ds\right)\hat{f}(\ell)\chi_\ell(m) \tag{33}$$

Using the substitution $u = s\lambda_\ell$, provided that $\lambda_\ell \neq 0$, reduces the integral appearing above to $\int \frac{g^2(u)}{u} du$, which is finite and equals C_g by the admissibility condition for g. If $\lambda_\ell = 0$, which holds only for $\ell = 0$, then the integral is 0 as $g(0) = 0$. This implies that Eq. (33) is precisely the inverse Fourier transform evaluated at vertex m, with the $\ell = 0$ omitted from the sum. As the omitted $\ell = 0$ term is precisely $\hat{f}(0)\chi_0 = \langle \chi_0, f\rangle\, \chi_0$, the lemma is proved.

This expression for the inverse of the continuous transform is of theoretical interest, however any practical implementation of the SGWT must use a finite number of wavelet scales. We shall discuss reconstruction from the scale discretized SGWT later in this chapter.

5.2 Frame Bounds for SGWT

As alluded to previously, practical computation of the SGWT must involve discretizing the scale parameter s to a finite set of values. Fixing our notation, we let J be the number of scales chosen and let $\{s_1, s_2, \ldots, s_J\}$ denote the specific scale values. The SGWT at each scale produces a set (often termed a "subband") of N coefficients $W_{s_j,n}$ for $1 \leq n \leq N$. Together with the N scaling function coefficients, the full transform with J scales may be considered as a mapping from $\mathbb{R}^N$ to $\mathbb{R}^{N(J+1)}$, producing $N(J+1)$ coefficients.

Some insight into how stable the coefficients for the entire set of NJ wavelets and N scaling functions are for representing signals may be gained by considering the frame formed by the entire set of wavelets and scaling functions. Briefly, for a Hilbert space $\mathscr{H}$, a set of vectors $\Gamma_n \in \mathscr{H}$ is said to be a frame with frame bounds A and B if for all $f \in \mathscr{H}$ it is true that

$$A\,||f||^2 \leq \sum_n |c_n|^2 \leq B\,||f||^2\,. \tag{34}$$

where the coefficients c_n are given by $c_n = \langle \Gamma_n, f \rangle$. These constants A and B describe the numerical stability of recovering the original signal f from the coefficients c_n. In particular, if $A = B$, then the set $\{\Gamma_n\}$ is called a tight frame, and the signal may be recovered simply by

$$f = \frac{1}{A} \sum_n c_n \Gamma_n \tag{35}$$

In general A and B will not be equal, however the guiding principal that the frame is easier to invert if B/A is close to 1 still holds. In fact, as discussed in Sect. 6.3, these frame bounds provide a precise estimate for the speed of convergence of the conjugate-gradients algorithm for inverting the discrete SGWT. For further details of the fundamentals of the theory of frames, see [39] or [40].

For the scale-discretized SGWT, the frame bounds are given by the following.

Theorem 1 *Fix a choice of a set of scales* $\{s_1, \ldots, s_J\}$. *Set* $G(\lambda) = h^2(\lambda) + \sum_j g(s_j\lambda)^2$, *where* h *and* g *are the scaling function and wavelet kernels. Then the set* $\Gamma = \{\phi_n\}_{n=1}^N \cup \{\psi_{s_j,n}\}_{j=1}^J{}_{n=1}^N$ *is frame with frame bounds* A, B *given by*

$$\begin{aligned} A &= \min_{\lambda \in [0, \lambda_{N-1}]} G(\lambda) \\ B &= \max_{\lambda \in [0, \lambda_{N-1}]} G(\lambda). \end{aligned} \tag{36}$$

Proof Expression (28), shows that, for a fixed signal f, we may write

$$\begin{aligned} \sum_n |W_f(s,n)|^2 &= \sum_n \sum_\ell g(s\lambda_\ell)\chi_\ell(n)\hat{f}(\ell) \sum_{\ell'} \left(g(s\lambda_{\ell'})\chi_{\ell'}(n)\hat{f}(\ell') \right)^* \\ &= \sum_\ell |g(s\lambda_\ell)|^2 |\hat{f}(\ell)|^2, \end{aligned} \tag{37}$$

where we have used the orthonormality of the χ_n. For the scaling function coefficients we have, similarly,

$$\sum_n |S_f(n)|^2 = \sum_\ell |h(\lambda_\ell)|^2 |\hat{f}(\ell)|^2 \tag{38}$$

Fix an ordering of the elements of Γ, so that $\Gamma = \cup_{k=1}^{N(J+1)} \gamma_k$. Note that $\langle \gamma_k, f \rangle$ may be either a scaling function coefficient or wavelet coefficient, depending on k. Equations (37) and (38) show that

$$\sum_{k=1}^{N(J+1)} |\langle \gamma_k, f \rangle|^2 = \sum_{\ell} \left(|h(\lambda_\ell)|^2 + \sum_{j=1}^{J} |g(s_j \lambda_\ell)|^2 \right) |\hat{f}(\ell)|^2 = \sum_{\ell} G(\lambda_\ell) |\hat{f}(\ell)|^2 \tag{39}$$

Then by the definition of A and B, we have

$$A \sum_{\ell=0}^{N-1} |\hat{f}(\ell)|^2 \leq \sum_{k=1}^{N(J+1)} |\langle \gamma_k, f \rangle|^2 \leq B \sum_{\ell=0}^{N-1} |\hat{f}(\ell)|^2 \tag{40}$$

The Parseval relation $||f||^2 = \sum_\ell |\hat{f}(\ell)|^2$ then implies that A and B are frame bounds for the frame Γ.

5.3 *Limit of Small Scales*

Much of the effectiveness of classical wavelets for signal processing follows as the wavelets may be designed to be localized in both the spatial domain and the frequency domain. The spectral graph wavelets may be designed to be localized in the frequency domain, provided that g is chosen as a band-pass filter. However, we have not yet demonstrated localization of the spectral graph wavelets in the spatial (i.e. vertex) domain.

The spatial localization properties of classical wavelets derived from a single wavelet via dilation and translation are straightforward to infer from the mother wavelet $\psi(t)$ itself. If $\psi(t)$ is well localized on the interval $[-d, d]$, then the derived wavelet $\psi_{s,a}(t)$ will be well localized on $[a - ds, a + ds]$. In the limit of small scales as $s \to 0$, this implies that $\psi_{s,a}(t) \to 0$ for all $t \neq a$, as long as the original mother $\psi(t)$ wavelet decays to zero as $t \to \infty$.

As scaling for the spectral graph wavelets is defined in the graph Fourier domain, localization in the limit as $s \to 0$ is not as straightforward to infer. We will demonstrate that normalized spectral graph wavelets $\frac{\psi_{s,n}}{||\psi_{s,n}||}$ will approach zero for vertices far enough away from the central vertex n, as $s \to 0$. Our result is based on the fact that powers of $\mathscr{L}$ are localized, and that T_g^s may be approximated as proportional to a power of $\mathscr{L}$ in the limit of small scales.

As noted previously, the operator T_g^s depends only on the values of $g(s\lambda)$ for λ in the spectrum of $\mathscr{L}$, in particular the values of $g(s\lambda)$ for $\lambda > \lambda_{N-1}$ have no effect on T_g^s. As the graph of $g(s\lambda)$ is obtained from the graph of $g(\lambda)$ by zooming in by a factor $1/s$, the operator T_g^s is determined by the values of $g(\lambda)$ over the small interval $[0, \lambda_{N-1} s]$. Our approach will be to approximate $g(\lambda)$ in a neighborhood of 0 by its Taylor polynomial, which will let us transfer the study of localization of T_g^s to studying localization of the first nonzero power of $\mathscr{L}$ appearing in the Taylor series.

The Taylor polynomial for g at the origin will provide an approximation of $g(s\lambda)$ that we will use for small s. In order to study the resulting approximate wavelets, we first establish a bound on how perturbations of the kernel function g affect the resulting wavelets. If two kernel functions g and $\tilde{g}$ are close to each other, then their resulting wavelets should also be close to each other.

Lemma 2 *Let g and $\tilde{g}$ be two kernel functions, and $\psi_{s,n} = T_g^s \delta_n$ and $\tilde{\psi}_{s,n} = T_{\tilde{g}}^s \delta_n$ be their corresponding wavelets at scale s. Suppose that there is a bound $M(s)$ so that $|g(s\lambda) - \tilde{g}(s\lambda)| \leq M(s)$ for all $\lambda \in [0, \lambda_{N-1}]$. It then follows that for each value of s and for each vertex m, $|\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)| \leq M(s)$, and that $\left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right|_2 \leq \sqrt{N} M(s)$.*

Proof As by definition $\psi_{s,n}(m) = \langle \delta_m, g(s\mathscr{L})\delta_n \rangle$, we may write

$$|\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)| = |\, \langle \delta_m, (g(s\mathscr{L}) - \tilde{g}(s\mathscr{L}))\, \delta_n \rangle\, | \tag{41}$$

Using the Parseval relation for the graph Fourier transform (20) shows this may be written as

$$\begin{aligned} |\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)| &= \left| \sum_\ell \chi_\ell(m)(g(s\lambda_\ell) - \tilde{g}(s\lambda_\ell))\chi_\ell^*(n) \right| \\ &\leq M(s) \sum_\ell |\chi_\ell(m)\chi_\ell(n)^*| \end{aligned} \tag{42}$$

Using the Cauchy–Schwartz inequality shows the above sum over ℓ is bounded by 1, as

$$\sum_\ell |\chi_\ell(m)\chi_\ell^*(n)| \leq \left(\sum_\ell |\chi_\ell(m)|^2 \right)^{1/2} \left(\sum_\ell |\chi_\ell^*(n)|^2 \right)^{1/2}, \tag{43}$$

and $\sum_\ell |\chi_\ell(m)|^2 = 1$ for all m, as the χ_ℓ are a complete orthonormal basis. Applying this to (42) proves $|\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)| \leq M(s)$. We may then write

$$\left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right|_2^2 = \sum_m \left(\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)\right)^2 \leq \sum_m M(s)^2 = N M(s)^2 \tag{44}$$

which proves the statement $\left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right|_2 \leq \sqrt{N} M(s)$.

Our localization results will be stated using a notion of distance between vertices. We employ the shortest-path distance, which defines the distance between two vertices as the number of edges in the shortest path connecting them, i.e.

$$d_G(m, n) = \operatorname*{argmin}_s \{k_1, k_2, \ldots, k_s\} \tag{45}$$

$$\text{s.t. } m = k_1, n = k_s, \text{ and } A_{k_r, k_{r+1}} > 0 \text{ for } 1 \leq r < s. \tag{46}$$

This distance measure treats the graph G as a binary graph, i.e. the particular values of the nonzero edge weights are not used.

For integer powers of $\mathscr{L}$, we have the following localization result. Note that this holds for both the normalized and non-normalized forms of the Laplacian.

Lemma 3 *Let G be a weighted graph, and $\mathscr{L}$ the graph Laplacian of G. Fix an integer $s > 0$, and pick vertices m and n. Then $(\mathscr{L}^s)_{m,n} = 0$ whenever $d_G(m, n) > s$.*

Proof By the construction of $\mathscr{L}$, we have that $\mathscr{L}_{r,s} = 0$ for any vertices $r \neq s$ that are not connected (i.e. where $A_{r,s} = 0$). Writing out repeated matrix multiplication, we see

$$(\mathscr{L}^s)_{m,n} = \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} \cdots \sum_{k_{s-1}=1}^{N} \mathscr{L}_{m,k_1} \mathscr{L}_{k_1,k_2} \ldots \mathscr{L}_{k_{s-1},n} \tag{47}$$

Suppose for sake of contradiction that $(L^s)_{m,n} \neq 0$. This implies that at least one of the terms in the above sum is nonzero, which demonstrates the existence of a sequence of vertices $k_1, k_2, \ldots, k_{s-1}$ with $\mathscr{L}_{m,k_1} \neq 0$, $L_{k_1,k_2} \neq 0, \ldots, \mathscr{L}_{k_{s-1}} \neq 0$. However, this is precisely a path of length s from vertex m to vertex n, with possible repeats of vertices. Removing these possible repeated vertices gives a path of length $k \leq s$ from vertex m to n, which implies $d_G(m, n) \leq s$, which is a contradiction.

This result implies that any kernel function that can be approximated by an integer power of $\mathscr{L}$ will produce localized wavelets. Every valid kernel g satisfies $g(0) = 0$, if g is smooth in a neighborhood of 0 then we may approximate $g(s\lambda)$ using the first nonzero term of the Taylor series for g, which will allow us to use Lemma 3. We first clarify this truncated Taylor approximation for kernels $g(x)$ that have a zero with integer multiplicity at $x = 0$.

Lemma 4 *Suppose g satisfies $g(0) = 0$, $g^{(r)}(0) = 0$ for all $r < K$, and $g^{(K)}(0) = C \neq 0$. Let there be some $s' > 0$ so that g is $K + 1$ times continuously differentiable on $[0, s'\lambda_{N-1}]$ and that $|g^{(K+1)}(\lambda)| \leq B$ for all $\lambda \in [0, s'\lambda_{N-1}]$. Define the monomial approximation kernel $\tilde{g}$ by $\tilde{g}(x) = (C/K!)x^K$, and set*

$$M(s) = \sup_{\lambda \in [0, \lambda_{N-1}]} |g(s\lambda) - \tilde{g}(s\lambda)|. \tag{48}$$

Then for all $s < s'$, the error $M(s)$ is bounded by

$$M(s) \leq s^{K+1} \frac{\lambda_{N-1}^{K+1}}{(K+1)!} B \tag{49}$$

Proof Using the assumed information about the derivatives of g at $x = 0$, Taylor's formula with remainder shows that for any $x < s'\lambda_{N-1}$,

$$g(x) = C\frac{x^K}{K!} + g^{(K+1)}(x^*)\frac{x^{K+1}}{(K+1)!}$$
$$= \tilde{g}(x) + g^{(K+1)}(x^*)\frac{x^{K+1}}{(K+1)!} \tag{50}$$

for some $x^* \in [0, x]$. By assumption we have $g^{(K+1)}(x^*) < B$. Now fix $s < s'$, and set $x = s\lambda$. We then have for all $0 \leq \lambda \leq \lambda_{N-1}$ that

$$|g(s\lambda) - \tilde{g}(s\lambda)| \leq B\frac{s^{K+1}\lambda^{K+1}}{(K+1)!} \leq B\frac{s^{K+1}\lambda_{N-1}^{K+1}}{(K+1)!}, \tag{51}$$

which implies $M(s) \leq s^{K+1}\frac{\lambda_{N-1}^{K+1}}{(K+1)!}B$.

We are now equipped to state our localization result for the spectral graph wavelets in the limit of small scales. We note that simply due to the definition of the SGWT, if $g(0) = 0$ and g is continuous it follows that $\lim_{s\to 0}\psi_{s,n}(m) = 0$ for all m, n. This explains why the statement of our result includes normalization by $||\psi_{s,n}||$.

Theorem 2 *Let g be a kernel function satisfying $g^{(r)}(0) = 0$ for $0 \leq r < K$, and $g^{(K)}(0) \neq 0$, and let s' and B be such that $|g^{(K+1)}(\lambda)| \leq B$ for all $0 \leq \lambda \leq s'\lambda_{N-1}$. Let m and n be vertices separated by distance greater than K, i.e. with $d_G(m, n) > K$. Then there are constants D and s'' so that*

$$\frac{\psi_{s,n}(m)}{||\psi_{s,n}||} \leq Ds \tag{52}$$

holds for all sufficiently small scales $s < \min(s', s'')$.

Proof Define $\tilde{g}(\lambda) = \frac{g^{(K)}(0)}{K!}\lambda^K$ and set $\tilde{\psi}_{s,n} = T_{\tilde{g}}^s\delta_n$. We expand $\tilde{\psi}_{s,n} = \frac{g^{(K)}(0)}{K!}s^K\mathscr{L}^K\delta_n$, so that

$$\tilde{\psi}_{s,n}(m) = \frac{g^{(K)}(0)}{K!}s^K\left\langle\delta_m, \mathscr{L}^K\delta_n\right\rangle$$
$$= \frac{g^{(K)}(0)}{K!}s^K\left(\mathscr{L}^K\right)_{m,n}$$
$$= 0 \tag{53}$$

as $(\mathscr{L}^K)_{m,n} = 0$ by Lemma 3. Combining Lemmas 2 and 4 shows

$$|\psi_{s,n}(m) - \tilde{\psi}_{s,n}(m)| = |\psi_{s,n}(m)| \leq s^{K+1}E \tag{54}$$

with $E = \frac{\lambda_{N-1}^{K+1}}{(K+1)!}B$.

We next need to bound $||\psi_{s,n}||$ away from 0. The triangle inequality applied to $\tilde{\psi}_{s,n} = \psi_{s,n} + (\tilde{\psi}_{s,n} - \psi_{s,n})$ directly gives $\left|\left|\tilde{\psi}_{s,n}\right|\right| \leq ||\psi_{s,n}|| + \left|\left|\tilde{\psi}_{s,n} - \psi_{s,n}\right|\right|$, so

$$\left|\left|\tilde{\psi}_{s,n}\right|\right| - \left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right| \leq \left|\left|\psi_{s,n}\right|\right| \tag{55}$$

Lemma 4 shows that $\left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right| \leq \sqrt{N} s^{K+1} \frac{\lambda_{N-1}^{K+1}}{(K+1)!} B$, while we may simply calculate $\left|\left|\tilde{\psi}_{s,n}\right|\right| = s^K \frac{g^{(K)}(0)}{K!} \left|\left|\mathscr{L}^K \delta_n\right|\right|$. These show that

$$s^K \left(\frac{g^{(K)}(0)}{K!} \left|\left|\mathscr{L}^K \delta_n\right|\right| - s\sqrt{N} \frac{\lambda_{N-1}^{K+1}}{(K+1)!} B \right) \leq \left|\left|\tilde{\psi}_{s,n}\right|\right| - \left|\left|\psi_{s,n} - \tilde{\psi}_{s,n}\right|\right| \tag{56}$$

Equations (56) and (54) together yield

$$\frac{\psi_{s,n}(m)}{\left|\left|\psi_{s,n}\right|\right|} \leq \frac{sE}{q - sp} \tag{57}$$

where we define $q = \frac{g^{(K)}(0)}{K!} \left|\left|\mathscr{L}^K \delta_n\right|\right|$ and $p = \sqrt{N} \frac{\lambda_{N-1}^{K+1}}{(K+1)!} B$. Straightforward computation demonstrates that $\frac{sE}{q-sp} \leq \frac{2E}{q} s$ whenever $s \leq \frac{q}{2p}$. This implies the stated theorem once we define $D = \frac{2EK!}{g^{(K)}(0)\left|\left|\mathscr{L}^K \delta_n\right|\right|}$ and $s'' = \frac{g^{(K)}(0)\left|\left|\mathscr{L}^K \delta_n\right|\right|(K+1)}{2\sqrt{N}\lambda_{N-1}^{K+1} B}$.

The localization result stated in Theorem 2 uses the shortest-path distance, and thus as stated is really only meaningful for graphs where the shortest-path distance (which treats all non-zero edges the same, even if the edge weights are close to zero) is a useful measure of distance. This will be the case if a significant number of edge weights are exactly zero. We note that many large graphs arising in practice are sparse (i.e. the number of nonzero edges is small relative to the total number of possible edges), for such sparse weighted graphs the shortest-path distance does provide a meaningful notion of distance.

6 Polynomial Approximation

The SGWT is defined using the eigenvectors χ_ℓ and eigenvalues λ_ℓ of the $N \times N$ matrix $\mathscr{L}$. Directly computing the transform according to Eq. 28 requires diagonalizing $\mathscr{L}$, i.e. computing the full set of eigenvectors and eigenvalues. This is computationally intensive, requiring $O(N^3)$ operations for the commonly used QR algorithm [41]. This computational complexity renders the direct computation of the entire set of eigenvectors impractical for graphs with more than a several thousand vertices. However, signal processing problems routinely involve data with hundreds of thousands or millions of dimensions. The SGWT cannot be a practical tool for such larger problems if its computation relies on fully diagonalizing $\mathscr{L}$.

In this section we describe a fast algorithm for the SGWT that avoids the need to diagonalize the graph Laplacian. This is a achieved by directly approximating the scaled wavelet kernels $g(s_j \lambda)$ by polynomials. A polynomial function of $\mathscr{L}$ may

be applied to a signal f in a manner which uses only matrix-vector multiplication. In general, multiplying a vector by $\mathscr{L}$ requires a number of operations equal to the number of nonzero edges in the graph. For sparse graphs, where the number of nonzero edges is small, this yields an efficient procedure.

6.1 Chebyshev Polynomial Approximation

As mentioned previously, the wavelet operator T_g^s depends on the values of $g(s\lambda)$ only for λ within the spectrum of $\mathscr{L}$. This implies that the polynomial approximations we seek need only be valid on an interval containing the spectrum of $\mathscr{L}$.

Lemma 5 *Let λ_{max} be an upper bound on the spectrum of $\mathscr{L}$, so that $\lambda_{max} \geq \lambda_{N-1}$. Let $p(\lambda)$ be a polynomial such that, for fixed scale s, $\max_{\lambda\in[0,\lambda_{max}]} |g(s\lambda) - p(\lambda)| = B$, and define the approximate wavelet coefficients by $\tilde{W}_f(s,n) = (p(\mathscr{L})f)_n$. Then the error in the approximate wavelet coefficients satisfies*

$$|W_f(s,n) - \tilde{W}_f(s,n)| \leq B\,||f|| \tag{58}$$

Proof Equation (28) shows

$$\begin{aligned} |W_f(s,n) - \tilde{W}_f(s,n)| &= \left| \sum_\ell g(s\lambda_\ell)\hat{f}(\ell)\chi_\ell(n) - \sum_\ell p(\lambda_\ell)\hat{f}(\ell)\chi_\ell(n) \right| \\ &\leq \sum_\ell |g(s\lambda_\ell) - p(\lambda_\ell)||\hat{f}(\ell)\chi_\ell(n)| \\ &\leq B \sum_\ell |\hat{f}(\ell)\chi_\ell(n)| \end{aligned} \tag{59}$$

The Cauchy–Schwartz inequality applied on the last sum above shows

$$\sum_\ell |\hat{f}(\ell)\chi_\ell(n)| \leq \left(\sum_\ell (\hat{f}(\ell))^2 \right)^{1/2} \left(\sum_\ell (\chi_\ell(n))^2 \right)^{1/2} = ||f||\,, \tag{60}$$

using the Parseval equality and the orthonormality of the χ_ℓ's. Together (59) and (60) imply Eq. (58).

An upper bound λ_{max} such as used in the above lemma can be found by calculating the largest eigenvalue of $\mathscr{L}$. It is important to note that good algorithms exist for finding the largest eigenvalue of a symmetric matrix that access the matrix only via matrix-vector multiplication. Examples include Arnoldi iteration and the Jacobi–Davidson method [41, 42]. These algorithms are able to compute accurate estimates of λ_{N-1} with much lower computational cost than needed to find the entire spectrum of $\mathscr{L}$. Additionally, as only a rough estimate of λ_{N-1} is needed to form an upper

bound (the rough estimate may simply be increased to ensure a valid upper bound), the Arnoldi iteration need not be run until close convergence is achieved for computing λ_{N-1}.

In this work we will use polynomial approximations computed from the truncated Chebyshev polynomial expansion of the scaled wavelet kernels $g(s\lambda)$, over the interval $[0, \lambda_{max}]$. Lemma 5 suggests that polynomial approximations $p(\lambda)$ should be chosen to minimize the supremum norm $B = \max_{\lambda \in [0, \lambda_{max}]} |p(\lambda) - g(s\lambda)|$. Truncated Chebyshev polynomial expansions give polynomials that in many cases are a close approximation of the so-called minimax polynomials that exactly minimize the supremum norm [43]. The minimax polynomial $p(x)$ of degree M for approximating $g(sx)$ has the property that the error $|p(x) - g(sx)|$ reaches the same maximum value at $M + 2$ points across the domain. This is illustrative of the fact that the minimax polynomials distribute the approximation error evenly over the entire interval. In contrast, for the wavelet kernels used in this work we have observed that the truncated Chebyshev polynomials have a maximum error only slightly greater than that of the minimax polynomials, and typically have significantly lower approximation error in regions where $g(s\lambda)$ is smooth. We have also observed that for graphs that are small enough where the SGWT may be computed exactly using Eq. (28), the polynomial approximation using truncated Chebyshev expansions produces a slightly lower approximation error than that based on the minimax polynomials. We illustrate a scaled wavelet kernel and both the truncated Chebyshev and minimax polynomial approximations (computed using the Remez exchange algorithm [44]) in Fig. 2.

Another reason we adopt the truncated Chebyshev approximation is that we can use the recurrence properties of the Chebyshev polynomials to conveniently evaluate Chebyshev polynomials of $\mathscr{L}$ applied to an input signal f via repeated matrix-vector multiplication. Chebyshev polynomial approximation is a classical topic, a good

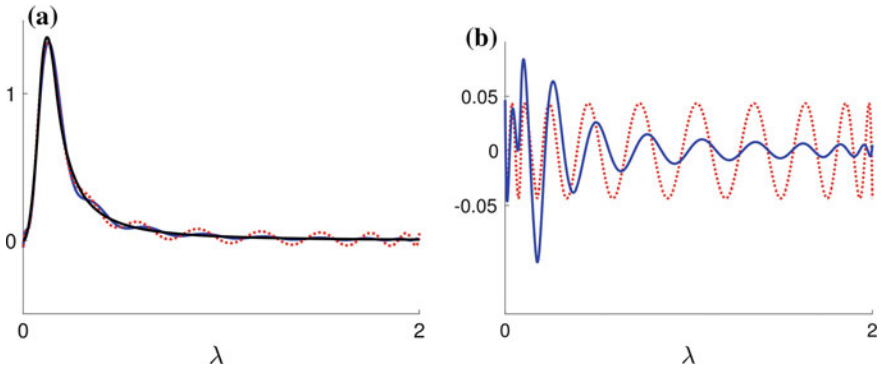


Fig. 2 **a** Wavelet kernel $g(\lambda)$ (black), truncated Chebyshev expansion (blue) and minimax polynomial approximation (red, dashed) for degree $m = 20$, shown for $[0, \lambda_{max}] = [0, 2]$. Approximation errors shown in **b**, the truncated Chebyshev expansion has maximum error 0.1023, the minimax polynomial has maximum error 0.0434

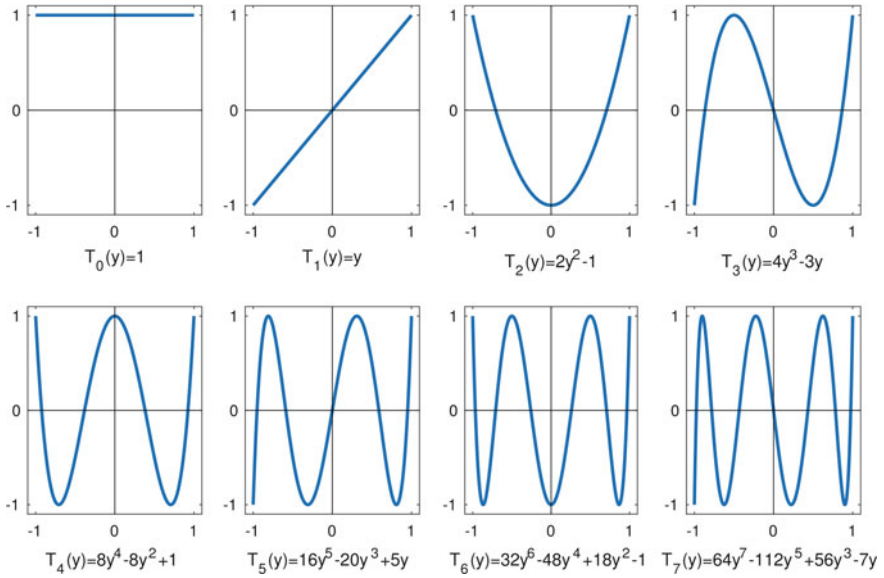


Fig. 3 Graphs of the Chebyshev polynomials $T_k(y)$ for $0 \leq k \leq 7$, plotted on the interval $[-1, 1]$

overview is [45]. For completeness we briefly describe a few key properties of the Chebyshev polynomials here.

The (unscaled) Chebyshev polynomials are a set of polynomials convenient for representing functions on the interval $[-1, 1]$. On this interval they satisfy $T_k(y) = \cos(k \arccos(y))$, showing that they oscillate between $[-1, 1]$, and that the kth order polynomial $T_k(y)$ has zeros at the points $y = \cos(\frac{\pi}{k}(n + \frac{1}{2}))$ for $n = 0, 1, 2, \ldots k - 1$. The Chebyshev polynomials satisfy the two-term recurrence relation

$$T_k(y) = 2yT_{k-1}(y) - T_{k-2}(y), \tag{61}$$

which together with the starting expressions $T_0(y) = 1$ and $T_1(y) = y$ can be used to generate the entire sequence. The graphs of the first 8 Chebyshev polynomials are shown in Fig. 3.

Many of the approximation properties of the Chebyshev polynomials follow from them being an orthogonal set, with respect to the inner product defined with the measure $\frac{dy}{\sqrt{1-y^2}}$. Specifically, one has

$$\int_{-1}^{1} \frac{T_l(y)T_m(y)}{\sqrt{1-y^2}} dy = \begin{cases} \delta_{l,m}\pi/2 & \text{if } m, l > 0 \\ \pi & \text{if } m = l = 0 \end{cases} \tag{62}$$

Any function h which is square-integrable on $[-1, 1]$ with respect to the measure $dy/\sqrt{1-y^2}$ has a convergent Chebyshev series, given by

$$h(y) = \frac{1}{2}c_0 + \sum_{k=1}^{\infty} c_k T_k(y) \tag{63}$$

where the Chebyshev coefficients c_k are given by

$$c_k = \frac{2}{\pi}\int_{-1}^{1} \frac{T_k(y)h(y)}{\sqrt{1-y^2}}dy = \frac{2}{\pi}\int_0^{\pi} \cos(k\theta)h(\cos(\theta))d\theta. \tag{64}$$

We now detail the polynomial approximation scheme used for the fast computation of the SGWT. We first rescale the argument of the Chebyshev polynomials by the change of variables $x = \lambda_{max}(y+1)/2$, which transforms the interval $[-1, 1]$ onto $[0, \lambda_{max}]$. We write $\overline{T}_k(x)$ for these shifted Chebyshev polynomials, which satisfy

$$\overline{T}_k(x) = T_k\left(\frac{2x - \lambda_{max}}{\lambda_{max}}\right) \tag{65}$$

We let M denote the degree of the polynomial approximations for each of the scaled wavelet kernels, and assume we have fixed some set of scales s_j. Larger values of M will yield more accurate approximations, at the expense of higher computational cost. For each scale s_j, the truncated Chebyshev polynomial $p_j(x)$ which approximates $g(s_j x)$ has the expression

$$p_j(x) = \frac{1}{2}c_{j,0} + \sum_{k=1}^{M} c_{j,k}\overline{T}_k(x), \tag{66}$$

where the Chebyshev coefficients are given by

$$c_{j,k} = \frac{2}{\pi}\int_0^{\pi} \cos(k\theta) g\left(s_j \tfrac{\lambda_{max}}{2}(\cos(\theta)+1)\right) d\theta. \tag{67}$$

Exactly the same scheme is used to construct the M degree polynomial $p_0(x)$ for approximating the scaling function kernel h.

These Chebyshev coefficients may be computed independent of any particular knowledge of the graph signal f, beyond of knowing an appropriate spectral bound λ_{max}. Once they are obtained, the wavelet and scaling function coefficients for the fast SGWT are:

$$\begin{aligned} \tilde{W}_f(s_j, n) &= \left(\frac{1}{2}c_{j,0}f + \sum_{k=1}^{M} c_{j,k}\overline{T}_k(\mathscr{L})f\right)_n \\ \tilde{S}_f(n) &= \left(\frac{1}{2}c_{0,0}f + \sum_{k=1}^{M} c_{0,k}\overline{T}_k(\mathscr{L})f\right)_n \end{aligned} \tag{68}$$

The fast SGWT relies on computation of the terms $\overline{T}_k(\mathscr{L})f$ using the recurrence relation satisfied by the shifted Chebyshev polynomials. Equation (61) and the inverse change of variables $y = \frac{2}{\lambda_{max}} - 1$ shows $\overline{T}_k(x) = \frac{4}{\lambda_{max}}(x-1)\overline{T}_{k-1}(x) - \overline{T}_{k-2}(x)$, which immediately implies

$$\overline{T}_k(\mathscr{L})f = \frac{4}{\lambda_{max}}(\mathscr{L} - I)\left(\overline{T}_{k-1}(\mathscr{L})f\right) - \overline{T}_{k-2}(\mathscr{L})f \tag{69}$$

Critically, this recurrence relation can be calculated using only matrix-vector multiplication, storing only the vector result $\overline{T}_k(\mathscr{L})f$ for each $k \leq M$ and never explicitly computing the matrix $\overline{T}_k(\mathscr{L})$. The above recurrence shows that the vector $\overline{T}_k(\mathscr{L})f$ can be computed from the vectors $\overline{T}_{k-1}(\mathscr{L})f$ and $\overline{T}_{k-2}(\mathscr{L})f$, with computational cost dominated by matrix-vector multiplication by $\mathscr{L} - I$.

We estimate the computational complexity of computing the approximate SGWT this way, for a graph with a total number of nonzero edges $|E|$. If $\mathscr{L}$ is stored using a sparse matrix representation, then the cost of the matrix-vector produce $\mathscr{L}v$ for any $v \in \mathbb{R}^N$ is $O(|E|)$ (as opposed to $O(N^2)$ for full matrix-vector multiplication). For sparse graphs, where $|E|$ is small compared to N^2, this difference may be very significant. Computing all of the terms $\overline{T}_k(\mathscr{L})f$ for $k \leq M$ requires $O(M|E|)$ operations. We compute the wavelet and scaling function coefficients according to Eq. (68), this may be done by adding the term $c_{j,k}\overline{T}_k(\mathscr{L})f$ for $j = 0, \ldots, J$ to a vector containing the jth set of coefficients, as the terms $\overline{T}_k(\mathscr{L})f$ are computed. Computing the scalar-vector product $c_{j,k}\overline{T}_k(\mathscr{L})f$ and adding it to the running total vector requires $O(N)$ operations, this cost is incurred $M(J+1)$ times for computing each of the $J+1$ wavelet or scaling function bands, up to polynomial order M. All together, this implies a total computational cost of $O(M|E| + MN(J+1))$ to compute the SGWT via polynomial approximation.

As the recurrence relation (69) involves only three terms, computing all of the $\overline{T}_k(\mathscr{L})f$ may be done with memory of size $3N$ if the lower degree terms are overwritten once they are no longer needed for the recurrence. A straightforward implementation of the fast SGWT would also need enough memory to hold each of the $J+1$ wavelet or scaling function bands, implying a total memory size requirement of $N(J+1) + 3N = N(J+4)$.

6.2 Polynomial Approximation for the SGWT Adjoint Operator

The SGWT wavelet and scaling function operators define linear mappings from $\mathbb{R}^N$ to the corresponding wavelet or scaling function coefficients. Once a set of J scales s_j is fixed, one may form the overall SGWT operator $W : \mathbb{R}^N \to \mathbb{R}^{N(J+1)}$ by concatenating all of the scaling function and wavelet coefficients into a single

$N(J+1)$ length vector, as $Wf = \left((T_h f)^T, (T_g^{s_1} f)^T, \ldots, (T_g^{s_J} f)^T\right)^T$. Letting $\tilde{W} : \mathbb{R}^N \to \mathbb{R}^{N(J+1)}$ be the SGWT operator computed by polynomial approximation, we have

$$\tilde{W} f = \left((p_0(\mathscr{L}) f)^T, (p_1(\mathscr{L}) f)^T, \ldots, (p_J(\mathscr{L}) f)^T\right)^T. \tag{70}$$

where the approximating polynomials p_j are defined in Eq. (66).

Both the adjoint operator $\tilde{W}^T \tilde{W} : \mathbb{R}^{N(J+1)} \to \mathbb{R}^N$ and the operator $W^T W : \mathbb{R}^N \to \mathbb{R}^N$ can be computed using Chebyshev polynomial approximation. These operators are used in the method we will detail in Sect. 6.3 for computing the inverse transformation. In addition, many wavelet based signal processing algorithms (in particular iterative algorithms for solving minimization problems arising from regularization using sparsity-promoting penalty functions of wavelet coefficients, see for example [46]) are described using the adjoint operator, so knowing that the adjoint may be efficiently computed is important for adapting such algorithms to graph signal processing using the SGWT.

The adjoint W^T is the linear operator such that $\langle W^T u, v \rangle = \langle u, Wv \rangle$ for every $u \in \mathbb{R}^{N(J+1)}$ and $v \in \mathbb{R}^N$. Below, we write $u \in \mathbb{R}^{N(J+1)}$ as the partitioned vector $u = (u_0^T, u_1^T, \ldots, u_J^T)$. We may then write

$$\begin{aligned} \langle u, Wv \rangle_{N(J+1)} &= \langle u, T_h v \rangle + \sum_{j=1}^{J} \left\langle u_j, T_g^{s_j} v \right\rangle_N \\ &= \left\langle T_h^T u_0, v \right\rangle + \left\langle \sum_{j=1}^{J} (T_g^{s_j})^T u_j, v \right\rangle_N = \left\langle T_h u_0 + \sum_{j=1}^{J} T_g^{s_j} u_j, f \right\rangle_N \end{aligned} \tag{71}$$

as the operators T_h and $T_g^{s_j}$ are all symmetric. Equation 71 shows that for any $u \in \mathbb{R}^{N(J+1)}$ viewed as the concatenation of $J+1$ coefficient subbands, application of the adjoint $W^T u$ is given by $W^T u = T_h u_0 + \sum_{j=1}^{J} T_g^{s_j} u_j$. Similarly, the adjoint of the approximate SGWT operator $\tilde{W}$ from Eq. (66) is given as

$$\tilde{W}^T u = \sum_{j=0}^{J} p_j(\mathscr{L}) u_j. \tag{72}$$

This can be computed efficiently, using only matrix-vector multiplication, using exactly the same approach as described in Sect. 6.1.

From Eqs. (70) and (72) we see that

$$\tilde{W}^T \tilde{W} f = \sum_{j=0}^{J} p_j(\mathscr{L}) p_j(\mathscr{L}) f = \left(\sum_{j=0}^{J} (p_j(\mathscr{L}))^2 \right) f \tag{73}$$

This expression may be computed efficiently by determining the Chebyshev coefficients d_k for $0 \leq k \leq 2M$ for the 2M degree sum-of-squares polynomial $P(x) = \sum_{j=0}(p_j(x))^2$. Once these are calculated, we compute

$$\tilde{W}^T \tilde{W} f = \sum_{k=0}^{2M} d_k \overline{T}_k(\mathscr{L}) f. \tag{74}$$

We detail the determination of the coefficients d_k below.

The expression $T_k(x) = \cos(k \arccos(x))$, together with the trigonometric identity $\cos(\alpha)\cos(\beta) = \frac{1}{2}(\cos(\alpha+\beta) + \cos(\alpha-\beta))$ implies the the product relation

$$T_k(x)T_l(x) = \frac{1}{2}\left(T_{k+l}(x) + T_{|k-l|}(x)\right). \tag{75}$$

We will use this to express the $d'_k s$ in terms of the $c'_{j,k} s$. For convenience below we denote $c'_{j,k} = c_{j,k}$ for $k \geq 1$ and $c'_{j,0} = \frac{1}{2} c_{j,0}$, so that we may write $p_j(x) = \sum_{k=0}^{M} c'_{j,k} \overline{T}_k(x)$, without the factor of $\frac{1}{2}$ as in (66).

Similarly we define the coefficients $d'_{j,k}$ so that $(p_j(x))^2 = \sum_{k=0}^{2M} d'_{j,k} \overline{T}_k(x)$. Expanding $(p_j(x))^2$ and using (66) yields, after a lengthy but straightforward calculation, that

$$d'_{j,k} = \begin{cases} \frac{1}{2}\left({c'_{j,0}}^2 + \sum_{i=0}^{M} {c'_{j,i}}^2\right) & \text{if } k = 0 \\ \frac{1}{2}\left(\sum_{i=0}^{k} c'_{j,i} c'_{j,k-i} + \sum_{i=0}^{M-k} c'_{j,i} c'_{j,k+i} + \sum_{i=k}^{M} c'_{j,i} c'_{j,i-k}\right) & \text{if } 0 < k \leq M \\ \frac{1}{2}\left(\sum_{i=k-M}^{M} c'_{j,i} c'_{j,k-i}\right) & \text{if } M < k \leq 2M \end{cases} \tag{76}$$

Defining $d_{n,0} = 2d'_{j,0}$ and $d_{j,k} = d'_{j,k}$ for $k \geq 1$, we have $d_k = \sum_{j=0}^{J} d_{j,k}$. These are used with Eq. 74 to compute $\tilde{W}^T \tilde{W} f$.

6.3 SGWT Inverse Transform

Many wavelet based signal processing algorithms function by computing wavelet coefficients of the original signal, manipulating the signal in the coefficient domain, and then inverting the wavelet transform. To be useful for signal processing, it is important to be able to invert the SGWT, i.e. to reconstruct a signal from a set of spectral graph wavelet coefficients. The scale discretized SGWT operator W computes $N(J+1)$ wavelet and scaling function coefficients for each N dimensional signal f. As the number of coefficients is greater than the dimension of the original signal, the SGWT is an overcomplete transform, and thus cannot have a unique linear inverse. Provided that the entire set of wavelet and scaling functions (equivalently, the

columns of W^T) span $\mathbb{R}^N$, there will be infinitely many different left inverse matrices M satisfying $MW = I$. We note that this condition holds if the frame bound A from Theorem 1 is positive.

We use the pseudoinverse, formally given by $M = (W^T W)^{-1} W^T$, as the inverse of the SGWT. For a given set of SGWT coefficients $c \in \mathbb{R}^{N(J+1)}$, the inverse SGWT will be the signal $f \in \mathbb{R}^N$ obtained by solving the linear system $(W^T W) f = W^T c$. For most applications this system is too large to be solved directly (for instance by the LU factorization and back substitution). Instead, we employ the well-known conjugate gradients algorithm [47]. This is an iterative algorithm, the computational cost at each step is dominated by computing the product of $W^T W$ with a single vector. We use the Chebyshev polynomial approximation scheme for computing $\tilde{W}^T \tilde{W}$ in each step of the conjugate gradients algorithm.

We note that the frame bounds from Theorem 1 may be used to estimate the convergence speed of the conjugate gradients iteration. The remaining error in the conjugate gradients algorithm after i iterations (as measured by the norm of the residual) is bounded by the norm of the first residual times

$$2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i, \tag{77}$$

where κ is the ratio of the largest and smallest eigenvalues of $W^T W$. The frame bounds A and B are bounds on the spectrum of $W^T W$ (see [39]), and thus $\kappa < A/B$. This explicitly shows that the convergence properties for the conjugate gradients reconstruction depend on the frame bounds, with faster convergence for smaller A/B.

7 SGWT Kernel Design Details

The described theory of the SGWT places few constraints on the wavelet kernel g, scaling function kernel h, or the selection of scales. We give details of the design choices described in the original paper [32], which are also those used in the example illustrative images included later in this chapter. The wavelet kernel g is chosen to give exact localization in the limit of small scales. By Theorem 2, this will occur if $g(x)$ behaves like a power of x near the origin. We ensure this by choosing g to be an exact monic polynomial for x in a neighborhood of the origin. For large x, g should decay to zero. This is enforced by setting g to decay as a negative power of x, for x larger than some fixed value. The final design connects these two regions by a cubic spline ensuring continuity of g and its derivative. Specifically, we set

$$g(x;\alpha,\beta,x_1,x_2) = \begin{cases} x_1^{-\alpha} x^{\alpha} & \text{for } x < x_1 \\ s(x) & \text{for } x_1 \le x \le x_2 \\ x_2^{\beta} x^{-\beta} & \text{for } x > x_2 \end{cases} \tag{78}$$

where α and β are integers, and x_1 and x_2 specify the transition region between the monic polynomial and decaying regions. The examples included in this chapter used α and β both set to 2, $x_1 = 1$ and $x_2 = 2$. In this case the cubic polynomial $s(x) = -5 + 11x - 6x^2 + x^3$.

The discrete scale values s_j are chosen by first specifying the maximum scale s_1 and the minimum scale s_J, then setting the intermediate scales decreasing and logarithmically equally spaced as $s_j = s_1(\frac{s_J}{s_1})^{\frac{j-1}{J-1}}$ for $1 \leq j \leq J$.

The minimum and maximum scales are adapted to the spectrum of $\mathscr{L}$ as follows. Given an upper bound λ_{max} on the spectrum of $\mathscr{L}$, and a value K that is considered a design parameter of the transform, we set $\lambda_{min} = \lambda_{max}/K$. The scales s_1 and s_J are chosen so that the smallest scale kernel $g(s_J x)$ is a monic polynomial over the interval $[0, \lambda_{max}]$, and so that the largest scale kernel $g(s_1 x)$ decays as $x^{-\beta}$ over the interval $[\lambda_{min}, \infty)$. This is ensured by setting $s_1 = x_2/\lambda_{min}$ and $s_J = x_1/\lambda_{max}$.

The scaling function kernel $h(x)$ is set as $h(x) = \gamma \exp(-(\frac{x}{0.6\lambda_{min}})^4)$. Here γ is determined by the condition that that $h(0)$ has the same value as the maximum value of g. An illustration of these choices for the scaling function and scaled wavelet kernels, for $\lambda_{max} = 20$, $K = 20$, $J = 4$, $\alpha = 2$, $\beta = 2$, $x_1 = 1$ and $x_2 = 2$ is given in Fig. 1.

We note that many other design choices for the wavelet and scaling function kernels are possible. In particular, Leonardi and Van De Ville have developed a design leading to a SGWT that is a tight frame, i.e. where the bounds A and B from (34) are equal [48].

8 Illustrative Examples

In order to illustrate the SGWT, we provide several examples of weighted graphs arising from different application areas, and present images of some wavelets and scaling functions for these graphs. As a first example, we consider a point cloud sampled randomly from the "swiss roll", a 2 dimensional manifold embedded that is widely used as a benchmark example for dimensionality reduction and manifold learning algorithms [49]. Our example is based on sampling 500 points from the embedding in $\mathbb{R}^3$ given parametrically by $x(u, v) = (v\cos(v)/4\pi, u, v\sin(v)/4\pi)$ for $-1 \leq u \leq 1, \pi \leq v \leq 4\pi$.

The adjacency for the weighted graph A is computed from the points x_i by assigning a greater edge weight to edges connecting points that are close in $\mathbb{R}^3$. Specifically, we set $A_{i,j} = \exp(-\left|\left|x_j - x_j\right|\right|^2/2\sigma^2)$, with $\sigma = 0.1$. We show the point cloud, a scaling function and four wavelets all centered on the same vertex, in Fig. 4.

The swiss roll point cloud is a toy example of data constrained to a lower dimensional manifold that are embedded in a higher dimensional space, a situation which commonly arises in many examples relevant to machine learning. We note in particular that the support of the wavelets and the scaling function automatically adapts to the structure of the underlying 2d manifold, and does not jump across to the upper

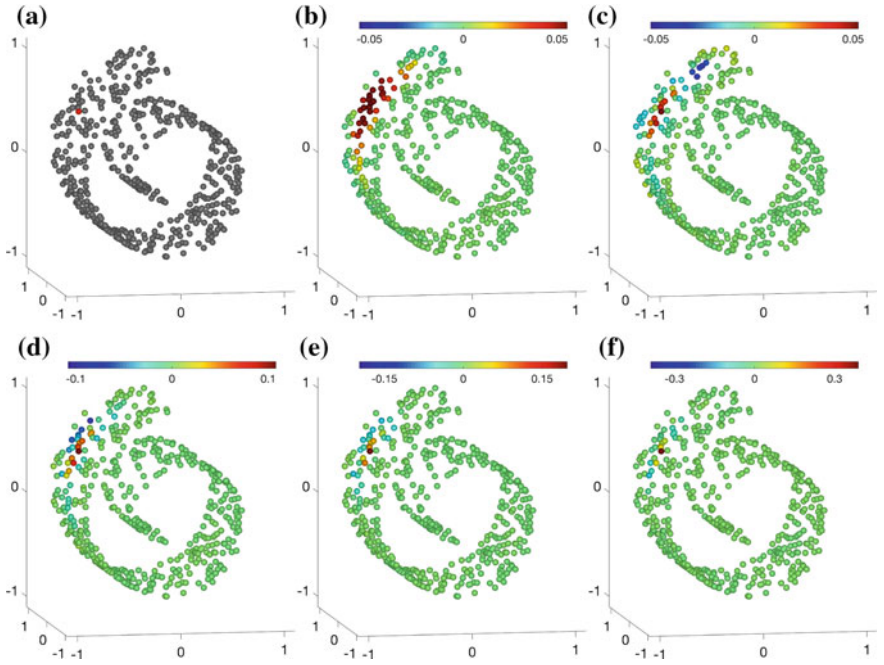


Fig. 4 Spectral graph wavelets on Swiss Roll data cloud, for transform with $J = 8$ wavelet scales (showing only wavelets for the 4 coarsest scales). **a** Vertex at which wavelets are centered, **b** scaling function, **c**–**f** wavelets, scales 1–4. Figure adapted from [32]

portion of the roll, even though the geometric separation in the 3D embedding space is smaller in some cases than the diameter of the support of the wavelet or scaling function.

We next consider an example transportation network, arising from a graph describing the road network in Minnesota. Here edges correspond to major roads, each vertex is the intersection of two roads. The vertices thus do not always exactly correspond to population centers (incorporated towns or cities), although many do. For this particular dataset the edges are unweighted, and do not for instance reflect the capacity of the road. Figure 5 shows a set of wavelets and scaling function centered at a single vertex, for the SGWT computed with parameter $K = 100$ and $J = 4$ scales. We note that for display purposes each vertex has associated 2d coordinates, however these were used only for rendering the figure and were not used for the actual computation of the SGWT.

With an eye towards applications, we note that the SGWT be useful for analysis of data measured at vertices of a transportation network where the phenomena generating the measured data was influenced in some way by the transportation network. Possible examples could include analysis of data describing disease rates during an epidemic (if it were expected that transportation network could influence patterns of

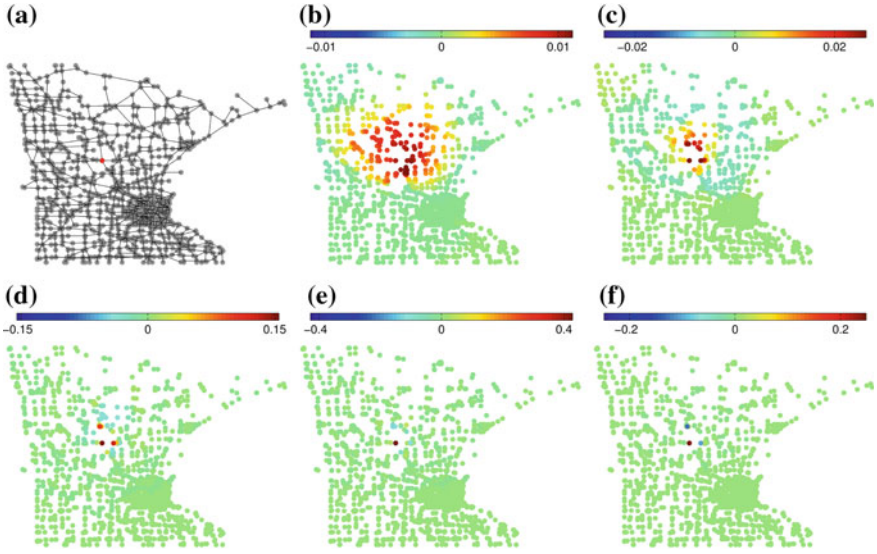


Fig. 5 Spectral graph wavelets on Minnesota road graph, with $K = 100$, $J = 4$ scales. **a** Vertex at which wavelets are centered, **b** scaling function, **c**–**f** wavelets, scales 1–4. Reproduced with permission from [32]

disease transmission), or analysis of inventory data for goods that are moved along the transportation network.

A third example shows the SGWT appropriate for data measured on irregularly shaped domains. We take as an example irregular domain the geometry of the surface of Lake Geneva. The SGWT for this case could be used for analysis or processing of some physical measurement (such as water temperature, or concentration of some solute) that was taken at regularly spaced points on the surface of the body of water. Using classical wavelet analysis for such data would require some special handling of the geometrically complex boundary between land and water. In contrast, the SGWT implicitly handles the boundary, and needs no special adaptation beyond encoding the domain with the adjacency matrix.

In this example the geometry of the lake surface is described as a binary function on a regular rectangular grid. The graph is constructed by retaining only vertices corresponding to grid points within the lake interior, and with edges connecting the (at most 4) neighboring vertices. We use a binary graph (all edge weights set to 1). At interior points of the domain, the graph Laplacian thus corresponds to the standard 5-point stencil for approximating $-\nabla^2$ (see Eq. (14)), while at points on the boundary the Laplacian is modified by the deletion of grid points outside of the lake domain. We constructed our lake geometry mask using shoreline data from the GSHHS database [50]. The lake mask was calculated on a 256×153 pixel grid (corresponding to a physical scale of 232 meters per pixel). We show a single wavelet for the largest scale value in Fig. 6, for the design with parameter $K = 100$ and $J = 5$ scales.

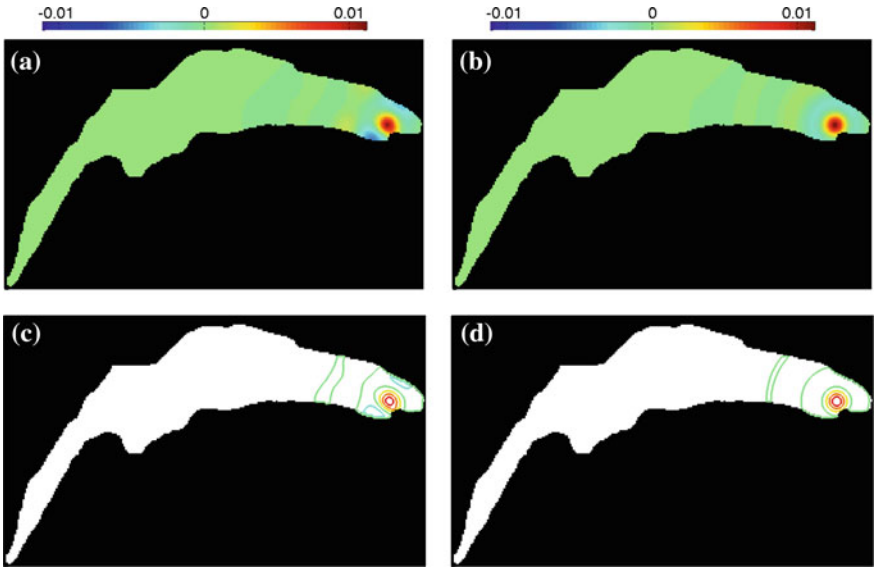


Fig. 6 Spectral graph wavelets on lake Geneva domain, (spatial map (**a**), contour plot (**c**)); compared with truncated wavelets from graph corresponding to complete mesh (spatial map (**b**), contour plot (**d**)). Note that the graph wavelets adapt to the geometry of the domain. Reproduced with permission from [32]

To illustrate the implicit adaption of the wavelet to the geometry of the domain, we compare it with a SGWT wavelet computed with the same wavelet kernel and scale parameter for a large regular grid, that is simply truncated. These true and truncated wavelets will coincide for central vertices that are far from the boundary, however they may be very different for wavelets centered on vertices near the boundary. This can be seen clearly in Fig. 6, which illustrates how the SGWT adapts to an irregular boundary.

9 Conclusion

We have described a wavelet transform for data defined on the vertices of arbitrary weighted graphs. Our approach uses spectral graph theory, based on the eigenvectors and eigenvalues of the graph Laplacian matrix, to define a notion of scaling that is analogous to classical wavelet operators. Our graph wavelet operators are defined by taking kernel functions of the graph Laplacian, where the kernel functions are formed by rescaling a single bandpass function. We have shown that defining the graph wavelets by applying these wavelet operators to a delta impulse centered on a single vertex gives wavelets that are localized in the limit of small scales. A fast algorithm based on Chebyshev polynomial approximation was described, demonstrating that

the SGWT can be applied to large graphs without the need for explicit computation of the eigenvectors and eigenvalues of the graph Laplacian matrix. We studied the frame bounds of the SGWT, and described a computation of the inverse transform. Finally, we showed a series of example images of the SGWT computed for several different graphs, highlighting potential applications of the transform.

Software Implementation

A MATLAB toolbox with complete functionality for computing the SGWT may be found online at http://wiki.epfl.ch/sgwt. Much of this functionality has also been incorporated into the larger Graph Signal Processing toolbox [51] (GSPBox), a MATLAB implementation is available at http://epfl-lts2.github.io/gspbox-html, and a Python implementation may be found at http://pygsp.readthedocs.io/en/stable.

References

1. A. Grossmann, J. Morlet, Decomposition of Hardy functions into square integrable wavelets of constant shape. SIAM J. Math. Anal. **15**(4), 723–736 (1984)
2. I. Daubechies, Orthonormal bases of compactly supported wavelets. Commun. Pure Appl. Math. **41**(7), 909–996 (1988)
3. S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans. Pattern Anal. Mach. Intell. **11**, 674–693 (1989)
4. Y. Meyer, Orthonormal wavelets, *Wavelets*. Inverse Problems and Theoretical Imaging (Springer, Berlin, 1989)
5. G. Beylkin, R. Coifman, V. Rokhlin, Fast wavelet transforms and numerical algorithms I. Commun. Pure Appl. Math. **44**(2), 141–183 (1991)
6. D.L. Donoho, I.M. Johnstone, Ideal spatial adaptation by wavelet shrinkage. Biometrika **81**, 425–455 (1994)
7. S. Chang, B. Yu, M. Vetterli, Adaptive wavelet thresholding for image denoising and compression. IEEE Trans. Image Process. **9**, 1532–1546 (2000)
8. L. Sendur, I. Selesnick, Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency. IEEE Trans. Signal Process. **50**, 2744–2756 (2002)
9. J. Portilla, V. Strela, M.J. Wainwright, E.P. Simoncelli, Image denoising using scale mixtures of Gaussians in the wavelet domain. IEEE Trans. Image Process. **12**, 1338–1351 (2003)
10. I. Daubechies, G. Teschke, Variational image restoration by means of wavelets: simultaneous decomposition, deblurring, and denoising. Appl. Comput. Harmon. Anal. **19**(1), 1–16 (2005)
11. F. Luisier, C. Vonesch, T. Blu, M. Unser, Fast interscale wavelet denoising of poisson-corrupted images. Signal Process. **90**(2), 415–427 (2010)
12. J. Shapiro, Embedded image coding using zerotrees of wavelet coefficients. IEEE Trans. Signal Process. **41**, 3445–3462 (1993)
13. A. Said, W. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical trees. IEEE Trans. Circuits Syst. Video Technol. **6**, 243–250 (1996)
14. M. Hilton, Wavelet and wavelet packet compression of electrocardiograms. IEEE Trans. Biomed. Eng. **44**, 394–402 (1997)
15. R. Buccigrossi, E. Simoncelli, Image compression via joint statistical characterization in the wavelet domain. IEEE Trans. Image Process. **8**, 1688–1701 (1999)
16. D. Taubman, M. Marcellin, *JPEG2000: Image Compression Fundamentals, Standards and Practice* (Kluwer Academic Publishers, Dordrecht, 2002)
17. J.-L. Starck, A. Bijaoui, Filtering and deconvolution by the wavelet transform. Signal Process. **35**(3), 195–211 (1994)

18. D.L. Donoho, Nonlinear solution of linear inverse problems by wavelet-vaguelette decomposition. Appl. Comput. Harmon. Anal. **2**(2), 101–126 (1995)
19. E. Miller, A.S. Willsky, A multiscale approach to sensor fusion and the solution of linear inverse problems. Appl. Comput. Harmon. Anal. **2**(2), 127–147 (1995)
20. R. Nowak, E. Kolaczyk, A statistical multiscale framework for Poisson inverse problems. IEEE Trans. Inf. Theory **46**, 1811–1825 (2000)
21. J. Bioucas-Dias, Bayesian wavelet-based image deconvolution: a GEM algorithm exploiting a class of heavy-tailed priors. IEEE Trans. Image Process. **15**, 937–951 (2006)
22. J.R. Wishart, Wavelet deconvolution in a periodic setting with long-range dependent errors. J. Stat. Plan. Inference **143**(5), 867–881 (2013)
23. F.K. Chung, *Spectral Graph Theory*. CBMS Regional Conference Series in Mathematics, vol. 92 (AMS Bookstore, Providence, 1997)
24. M. Crovella, E. Kolaczyk, Graph wavelets for spatial traffic analysis, in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, Jan 2003*, vol. 3 (IEEE, 2003), pp. 1848–1857
25. A. Smalter, J. Huan, G. Lushington, Graph wavelet alignment kernels for drug virtual screening. J. Bioinform. Comput. Biol. **7**, 473–497 (2009)
26. M. Jansen, G.P. Nason, B.W. Silverman, Multiscale methods for data on graphs and irregular multidimensional situations. J. R. Stat. Soc. Ser. (Stat. Methodol.) **71**(1), 97–125 (2009)
27. F. Murtagh, The Haar wavelet transform of a dendrogram. J. Classif. **24**, 3–32 (2007)
28. A.B. Lee, B. Nadler, L. Wasserman, Treelets - an adaptive multi-scale basis for sparse unordered data. Ann. Appl. Stat. **2**, 435–471 (2008)
29. R.R. Coifman, M. Maggioni, Diffusion wavelets. Appl. Comput. Harmon. Anal. **21**, 53–94 (2006)
30. M. Maggioni, H. Mhaskar, Diffusion polynomial frames on metric measure spaces. Appl. Comput. Harmon. Anal. **24**(3), 329–353 (2008)
31. D. Geller, A. Mayeli, Continuous wavelets on compact manifolds. Mathematische Zeitschrift **262**, 895–927 (2009)
32. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
33. D. Thanou, D.I. Shuman, P. Frossard, Learning parametric dictionaries for signals on graphs. IEEE Trans. Signal Process. **62**, 3849–3862 (2014)
34. W.H. Kim, D. Pachauri, C. Hatt, M.K. Chung, S. Johnson, V. Singh, Wavelet based multiscale shape features on arbitrary surfaces for cortical thickness discrimination, in *Advances in Neural Information Processing Systems 25* ed. by F. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Curran Associates Inc., 2012), pp. 1241–1249
35. W.H. Kim, M.K. Chung, V. Singh, Multi-resolution shape analysis via non-Euclidean wavelets: applications to mesh segmentation and surface alignment problems, in *2013 IEEE Conference on Computer Vision and Pattern Recognition, June 2013* (2013), pp. 2139–2146
36. N. Tremblay, P. Borgnat, Multiscale community mining in networks using spectral graph wavelets, in *21st European Signal Processing Conference (EUSIPCO 2013), Sept 2013* (2013), pp. 1–5
37. M. Zhong, H. Qin, Sparse approximation of 3d shapes via spectral graph wavelets. Visual Comput. **30**, 751–761 (2014)
38. M. Reed, B. Simon, *Methods of Modern Mathematical Physics Volume 1: Functional Analysis* (Academic Press, London, 1980)
39. I. Daubechies, *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics (1992)
40. C.E. Heil, D.F. Walnut, Continuous and discrete wavelet transforms. SIAM Rev. **31**(4), 628–666 (1989)
41. D. Watkins, *The Matrix Eigenvalue Problem - GR and Krylov Subspace Methods*. Society for Industrial and Applied Mathematics (2007)
42. G.L.G. Sleijpen, H.A.V. der Vorst, A Jacobi–Davidson iteration method for linear eigenvalue problems. SIAM J. Matrix Anal. Appl. **17**(2), 401–425 (1996)

43. K.O. Geddes, Near-minimax polynomial approximation in an elliptical region. SIAM J. Numer. Anal. **15**(6), 1225–1233 (1978)
44. W. Fraser, A survey of methods of computing minimax and near-minimax polynomial approximations for functions of a single independent variable. J. Assoc. Comput. Mach. **12**, 295–314 (1965)
45. G.M. Phillips, *Interpolation and Approximation by Polynomials*. CMS Books in Mathematics (Springer, Berlin, 2003)
46. I. Daubechies, M. Defrise, C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. Commun. Pure Appl. Math. **57**(11), 1413–1457 (2004)
47. J.R. Shewchuk, An introduction to the conjugate gradient method without the agonizing pain. Technical report, Pittsburgh, PA, USA, 1994
48. N. Leonardi, D.V.D. Ville, Tight wavelet frames on multislice graphs. IEEE Trans. Signal Process. **61**, 3357–3367 (2013)
49. J.B. Tenenbaum, V.d. Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
50. P. Wessel, W.H.F. Smith, A global, self-consistent, hierarchical, high-resolution shoreline database. J Geophys. Res. **101**(B4), 8741–8743 (1996)
51. N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, D.K. Hammond, GSPBOX: a toolbox for signal processing on graphs, ArXiv e-prints (2014)

Spectral Design of Signal-Adapted Tight Frames on Graphs



Hamid Behjat and Dimitri Van De Ville

Abstract Analysis of signals defined on complex topologies modeled by graphs is a topic of increasing interest. Signal decomposition plays a crucial role in the representation and processing of such information, in particular, to process graph signals based on notions of scale (e.g., coarse to fine). The graph spectrum is more irregular than for conventional domains; i.e., it is influenced by graph topology, and, therefore, assumptions about spectral representations of graph signals are not easy to make. Here, we propose a tight frame design that is adapted to the graph Laplacian spectral content of a given class of graph signals. The design exploits the ensemble energy spectral density, a notion of spectral content of the given signal set that we determine either directly using the graph Fourier transform or indirectly through approximation using a decomposition scheme. The approximation scheme has the benefit that (i) it does not require diagonalization of the Laplacian matrix, and (ii) it leads to a smooth estimate of the spectral content. A prototype system of spectral kernels each capturing an equal amount of energy is defined. The prototype design is then warped using the signal set's ensemble energy spectral density such that the resulting subbands each capture an equal amount of ensemble energy. This approach accounts at the same time for graph topology and signal features, and it provides a meaningful interpretation of subbands in terms of coarse-to-fine representations.

H. Behjat (✉)
Lund University, Lund, Sweden
e-mail: hamid.behjat@bme.lth.se

D. Van De Ville
Institute of Bioengineering, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
e-mail: dimitri.vandeville@epfl.ch

D. Van De Ville
Department of Radiology and Medical Informatics, University of Geneva, Geneva, Switzerland

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*, Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_4

1 Introduction

Many fields of science rely on network analysis to study complex systems. Networks are modeled mathematically as weighted graphs that have a set of nodes (vertices) with interactions between them represented by connections (links) and associated strengths. A rich repertoire of methods have been developed to pursue original queries and integrate the complexity of network structure into the analysis, subsequently providing new interpretations of datasets in divers scientific disciplines ranging from social sciences to physics and biology. One of the successes in network analysis is the ability to identify sets of nodes based on their connectivity. Traditional graph partitioning goes back to optimizing graph cuts [14], while more recent community detection identifies sets of nodes that are more densely connected inside the set than outside [17]. Community detection has been widely applied and many variants of the corresponding optimization criterion have been proposed [31].

Another significant trend in the field is the emergence of methods to process signals on graphs [7, 32, 38, 41]. Measurements on the nodes of a given network can be considered as graph signals for which classical signal processing operations can be generalized; e.g., how to properly denoise, filter, or transform graph signals by taking into account the underlying connectivity. Many generalization schemes have been proposed to extend classical multi-resolution transforms, filter bank designs and dictionary constructions to the graph setting. These studies fall essentially within two families: spatial (vertex) and spectral (frequency) designs. Schemes that fall within the former family include methods in designing wavelets for hierarchical trees [16, 34, 35] and methods based on lifting schemes [22, 29, 36]. The latter family is based on spectral graph theory [8], which is a powerful approach based on the eigendecomposition of matrices associated with graphs such as the adjacency matrix or graph Laplacian. Its strength originates from the global nature of the eigenvectors that summarize key graph properties and can be used to solve convex relaxed versions of graph cut minimization [50], or to define signal-processing operations by a graph equivalent of the Fourier transform [38, 41]. In its application to graph signal processing, operations are performed in the spectral domain using graph spectral filters. One of the first proposals is the spectral graph wavelet transform (SGWT) frame [20] that is constructed based on a system of scaled cubic spline spectral kernels together with a lowpass spectral polynomial kernel. Moreover, various constructions of systems of spectral graph kernels leading to tight frames were proposed in [13, 18, 26]. Tight frames are particularly interesting because of their property of energy conservation between the original and transformed domain [5]. Other approaches to spectral domain design include diffusion wavelets [10], vertex-frequency frames [42, 44] and approaches to graph filter-bank design using bipartite graph decompositions [30, 37, 45, 46], connected sub-graph decomposition [48], graph coloring [40] and Slepian functions that provide a tradeoff between temporal and spectral energy concentration [49].

One of the difficulties of the graph spectrum is that its construction depends on the graph itself. Consequently, the graph spectral representation of a graph signal is

determined by both the domain and the signal. However, the aforementioned spectral designs typically define spectral windows in a way that is independent of the graph and graph signal. One example of adaptation to the spectral properties of the graph domain was recently proposed in [43] for the construction of spectrum-adapted tight graph wavelet and vertex-frequency frames. The spectrum-adapted kernels account for the non-uniform distribution of Laplacian eigenvalues, and are designed such that that a similar number of eigenvalues falls within the support of each spectral kernel. Moreover, in [47, 51], numerical dictionary learning approaches have been proposed in which dictionaries are learnt based on a set of training signals. In these design, the learned kernels are indirectly adapted to the graph Laplacian spectrum as well as to the training data since the graph structure is incorporated into the learning process. In an application specific approach, in [1–3], the Meyer-like frame design [26] has been tailored to the spectral content of functional MRI signals to obtain a set of narrow-support kernels covering the lower end of the spectrum.

In this chapter, we propose an approach for constructing tight graph frames that account not only for the intrinsic topological structure of the underlying graph as proposed in [43], but also for the characteristics of a given set of signals. This is accomplished by considering a graph-based energy spectral density notion that includes signal and topology properties and encodes the energy-wise significance of the graph eigenvalues. A system of spectral kernels tailored to the energy spectral density is constructed by starting from the design of a prototype Meyer-type tight frame with uniform spectral coverage, followed by a warping step which incorporates the energy spectral density information to the prototype design, resulting in a tight frame with equi-energy subbands.

2 Preliminaries

2.1 Graphs and Spectral Graph Theory

A graph can be denoted as $G = (V, E)$ with N_g vertices in set V, a set of edges as tuples (i, j) in E where $i, j \in V$. The size of the graph is the number of vertices. In this chapter we only consider undirected graphs without self-loops. Algebraically, G can be represented with the node-to-node adjacency matrix A, with elements $a_{i,j}$ denoting the weight of the edge (i, j) if $(i, j) \in E; a_{i,j} = 0$ if $(i, j) \notin E$. The degree matrix D of G is diagonal with elements $d_{i,i} = \sum_j a_{i,j}$. The Laplacian matrices of G in combinatorial form L and normalized form $\mathcal{L}$ are defined as

$$L = D - A, \tag{1}$$

$$\mathcal{L} = D^{-1/2} L D^{-1/2}, \tag{2}$$

respectively. Both L and $\mathcal{L}$ are symmetric and positive semi-definite, and thus, their diagonalizations lead to a set of N_g real, non-negative eigenvalues that define the graph Laplacian spectrum

$$\Lambda(G) = \{0 = \lambda_1 \leq \lambda_2 \cdots \leq \lambda_{N_g} = \lambda_{\max}\}. \tag{3}$$

The corresponding set of eigenvectors $\{\boldsymbol{\chi}_l\}_{l=1}^{N_g}$ forms a complete set of orthonormal vectors that span the graph spectral domain [8]. When necessary, we use the notations $\Lambda_L(G)$ and $\Lambda_{\mathcal{L}}(G)$ to distinguish between the two definitions of the graph Laplacian. As the eigenvalues may be repetitive, for each λ_l, we denote its algebraic multiplicity by m_{λ_l} and the index of its first occurrence by i_{λ_l}. That is, if λ_l is singular, i.e. $m_{\lambda_l} = 1$, then $i_{\lambda_l} = l$, and if λ_l is repetitive, then $i_{\lambda_l} \leq l$. The multiplicity of eigenvalues equal to zero reflects the number of connected components in the graph. In this paper, only connected graphs are considered, and thus, $m_{\lambda_1} = 1$.

2.2 Graph Signals: Vertex Versus Spectral Representations

Let $\ell_2(G)$ denote the Hilbert space of all square-summable real-valued vectors $\mathbf{f} \in \mathbb{R}^{N_g}$, with the inner product defined as

$$\langle \mathbf{f}_1, \mathbf{f}_2 \rangle = \sum_{n=1}^{N_g} f_1[n] f_2[n], \quad \forall \mathbf{f}_1, \mathbf{f}_2 \in \ell_2(G) \tag{4}$$

and the norm as

$$||\mathbf{f}||_2^2 = \langle \mathbf{f}, \mathbf{f} \rangle = \sum_{n=1}^{N_g} |f[n]|^2, \quad \forall \mathbf{f} \in \ell_2(G). \tag{5}$$

A real signal defined on the vertices of a graph, $\mathbf{f} : V \to \mathbb{R}$, can be seen as a vector in $\ell_2(G)$, where the n-th element represents the value of the signal on the n-th vertex.

For any $\mathbf{f} \in \ell_2(G)$, its spectral representation $\widehat{\mathbf{f}} \in \ell_2(G)$, known as the graph Fourier transform of $\mathbf{f}$, can be used to express $\mathbf{f}$ in terms of the graph Laplacian eigenvectors

$$f[n] = \sum_{l=1}^{N_g} \underbrace{\langle \mathbf{f}, \boldsymbol{\chi}_l \rangle}_{=\widehat{f}[l]} \chi_l[n]. \tag{6}$$

With this definition of the Fourier transform, it can be shown that the Parseval relation holds [42]

$$\forall \mathbf{f}_1, \mathbf{f}_2 \in \ell_2(G), \quad \langle \mathbf{f}_1, \mathbf{f}_2 \rangle = \langle \widehat{\mathbf{f}}_1, \widehat{\mathbf{f}}_2 \rangle. \tag{7}$$

2.3 Filtering of Graph Signals

In the graph setting, the generalized convolution product is defined as

$$(\mathbf{f}_1 * \mathbf{f}_2)[n] = \sum_{l=1}^{N_g} \widehat{f}_1[l]\widehat{f}_2[l]\chi_l[n], \quad \forall \mathbf{f}_1, \mathbf{f}_2 \in \ell_2(G)). \tag{8}$$

In analogy with conventional signal processing, filtering of graph signals can be viewed as an operation in the spectral domain. For a given graph signal $\mathbf{f} \in \ell_2(G)$ and graph filter $\mathbf{g} \in \ell_2(G)$, defined through its Fourier transform $\widehat{\mathbf{g}}$, the filtered signal, denoted by $(F_\mathbf{g}\mathbf{f})$, can be obtained as

$$(F_\mathbf{g}\mathbf{f})[n] = (\mathbf{g} * \mathbf{f})[n] \tag{9}$$

$$\overset{(8)}{=} \sum_{l=1}^{N_g} \widehat{g}[l]\widehat{f}[l]\chi_l[n]. \tag{10}$$

For the graph filter $\mathbf{g}$, the filter response of an impulse at vertex m

$$\mathbf{f} = \boldsymbol{\delta}_m \leftrightarrow \widehat{\delta}_m[l] = \langle \boldsymbol{\delta}_m, \boldsymbol{\chi}_l \rangle = \chi_l[m], \tag{11}$$

can be obtained as

$$(F_\mathbf{g}\boldsymbol{\delta}_m)[n] = \sum_{l=1}^{N_g} \widehat{g}[l]\chi_l[m]\chi_l[n]. \tag{12}$$

The impulse response of a graph filter is, in general, shift-variant; i.e, the impulse response at one vertex is not simply a shifted version of the impulse response at any other node. This is due to the absence of a well-defined shift operator in the graph setting as that defined in the Euclidean setting. Therefore, a graph filter is conventionally defined by its spectral kernel $\widehat{\mathbf{g}}$ rather than by its impulse response.

Although the graph spectrum is discrete, to design spectral kernels, it is often more elegant to define an underlying smooth continuous kernel. Let $L_2(G)$ denote the Hilbert space of all square-integrable spectral functions $K(\lambda) : [0, \lambda_{\max}] \to \mathbb{R}^+$, with the inner product defined as

$$\langle K_1, K_2 \rangle_{\mathrm{L}_2} = \int_{-\infty}^{+\infty} K_1(\lambda)K_2(\lambda)d\lambda, \quad \forall K_1, K_2 \in \mathrm{L}_2(G), \tag{13}$$

and the L_2-norm defined as

$$\|K\|_{\mathrm{L}_2}^2 = \langle K, K \rangle_{\mathrm{L}_2}, \quad \forall K \in \mathrm{L}_2(G). \tag{14}$$

A discrete version of $K(\lambda) \in L_2(G)$ can then be determined as

$$k[l] = K(\lambda_l), \quad l = 1, \ldots, N_g. \tag{15}$$

Note that although $\mathbf{k}$ is defined in the spectral domain, it is not linked to any explicit vertex representation, and thus, the Fourier symbol $\widehat{\ }$ is not used for their denotation. This notation convention will be used throughout the chapter.

2.4 Dictionary of Graph Atoms

For a given spectral kernel $\mathbf{k}$ associated with $K(\lambda)$, the vertex-domain impulse responses are obtained as

$$\boldsymbol{\psi}_{K,m} = (F_{\mathbf{k}}\boldsymbol{\delta}_m) \leftrightarrow \widehat{\boldsymbol{\psi}}_{K,m}[l] = k[l]\chi_l[m]. \tag{16}$$

The collection of impulse responses $\{\boldsymbol{\psi}_{K,m}\}_{m=1}^{N_g}$ are considered as graph *atoms* associated with spectral kernel $K(\lambda)$. Given a set of J spectral kernels $\{\mathbf{k}_j \in \ell_2(G)\}_{j=1}^{J}$, a dictionary of graph atoms D_G with JN_g elements can be obtained

$$D_G = \left\{ \{\boldsymbol{\psi}_{K_j,m}\}_{j=1}^{J} \right\}_{m=1}^{N_g}. \tag{17}$$

The atoms of D_G form a frame in $\ell_2(G)$ if there exist bounds $B_2 \geq B_1 > 0$ such that [5]

$$\forall \mathbf{f} \in \ell_2(G), \quad B_1||\mathbf{f}||_2^2 \leq \sum_{j,m} |\langle \mathbf{f}, \boldsymbol{\psi}_{K_j,m} \rangle|^2 \leq B_2||\mathbf{f}||_2^2, \tag{18}$$

where the frame bounds are given by

$$B_1 = \min_{\lambda \in [0,\lambda_{\max}]} G(\lambda), \quad B_2 = \max_{\lambda \in [0,\lambda_{\max}]} G(\lambda), \tag{19}$$

and $G(\lambda) \in L_2(G)$ is defined as

$$G(\lambda) = \sum_{j=1}^{J} |K_j(\lambda)|^2. \tag{20}$$

In particular, D_G forms a tight frame if

$$\forall \lambda \in [0, \lambda_{\max}], \quad G(\lambda) = C, \tag{21}$$

and a Parseval frame if $C = 1$.

2.5 Decomposition of Graph Signals

Direct Decomposition

To decompose a graph signal $\mathbf{f}$ onto a set of the atoms in D_G, the coefficients can be obtained as

$$c_{K_j,m} = \langle \mathbf{f}, \boldsymbol{\psi}_{K_j,m} \rangle \tag{22}$$

$$\overset{(7)}{=} \sum_{l=1}^{N_g} \widehat{\psi}_{K_j,m}[l]\widehat{f}[l], \tag{23}$$

$$\overset{(16)}{=} \sum_{l=1}^{N_g} k_j[l]\widehat{f}[l]\chi_l[m]. \tag{24}$$

Relation (24) shows that the direct decomposition requires a full eigendecomposition of the L since it requires (i) the Laplacian eigenvectors $\{\boldsymbol{\chi}_l\}_{l=1}^{N_g}$ and (ii) the graph Fourier transform of the signal $\widehat{\mathbf{f}}$.

If D_G forms a Parseval frame, the coefficients can be used to recover the original signal as

$$\begin{aligned} f[n] &= \sum_j \sum_m c_{K_j,m} \boldsymbol{\psi}_{K_j,m} \\ &= \sum_j \sum_m \sum_l k_j[l]\widehat{f}[l]\chi_l[m] \sum_{l'} k_j[l']\boldsymbol{\chi}_{l'}[m]\chi_{l'}[n] \\ &= \sum_l \sum_{l'} \sum_j k_j[l]k_j[l']\widehat{f}[l]\chi_{l'}[n] \underbrace{\sum_m \chi_l[m]\chi_{l'}[m]}_{\delta_{l-l'}} \\ &= \sum_l \underbrace{\sum_j k_j^2[l]}_{=1} \widehat{f}[l]\chi_l[n]. \end{aligned} \tag{25}$$

Decomposition Through Polynomial Approximation

The decomposition of $\mathbf{f}$ on D_G leads to a coefficient vector associated to each $\mathbf{k}_j$ given as

$$\mathbf{c}_{K_j} = [c_{K_j,1}, c_{K_j,2}, \ldots, c_{K_j,N_g}]^T \tag{26}$$

$$\overset{(24)}{=} \sum_{l=1}^{N_g} k_j[l]\widehat{f}[l]\boldsymbol{\chi}_l, \tag{27}$$

that can be interpreted as filtered versions of $\mathbf{f}$ with different spectral kernels $\{\mathbf{k}_j\}_{j=1}^{J}$. Due to the redundancy of such a transform, it is beneficial to implement the transform using a fast algorithm, rather than using the explicit computation of the coefficients through (24). Moreover, for large graphs, it can be cumbersome to compute the full eigendecomposition of L, and in extensively large graphs this can in fact be impossible. One solution to overcome this computational burden is to use a polynomial approximation scheme.

One such algorithm is the truncated Chebyshev polynomial approximation method [20], which is based on considering the expansion of the continuous spectral window functions $\{K_j(\lambda)\}_{j=1}^{J}$ with the Chebyshev polynomials $C_p(x) = \cos(p \arccos(x))$ as

$$K_j(\lambda) = \frac{1}{2} d_{K_j,0} + \sum_{p=1}^{\infty} d_{K_j,p}\, \bar{C}_p(\lambda)\,, \tag{28}$$

where $\bar{C}_p(x) = C_p(\frac{x-b}{b})$, $b = \lambda_{\max}/2$ and $d_{K_j,p}$ denote the Chebyshev coefficients obtained as

$$d_{K_j,p} = \frac{2}{\pi} \int_0^{\pi} \cos(p\theta) K_j(b(\cos(\theta)+1)) d\theta. \tag{29}$$

By truncating (28) to M terms, $K_j(\lambda)$ can be approximated as an M-th order polynomial $P_j(\lambda) \in L_2(G)$. Consequently, $\mathbf{c}_{K_j}$ can be approximated as

$$\mathbf{c}_{K_j} \overset{(27)}{=} \sum_{l=1}^{N_g} \underbrace{k_j[l]}_{K_j(\lambda_l)} \widehat{f}[l] \boldsymbol{\chi}_l \tag{30}$$

$$\approx \sum_{l=1}^{N_g} P_j(\lambda_l) \widehat{f}[l] \boldsymbol{\chi}_l \tag{31}$$

$$= P_j(L) \sum_{l=1}^{N_g} \widehat{f}[l] \boldsymbol{\chi}_l \tag{32}$$

$$\overset{(6)}{=} P_j(L)\mathbf{f} \tag{33}$$

where in (32) we exploit the property $L\chi_l = \lambda_l \chi_l \Rightarrow P_j(L)\chi_l = P_j(\lambda_l)\chi_l$.

3 Ensemble Energy Spectral Density

The ensemble energy spectral density can be either computed using the graph Fourier transform or approximated through decomposition of the signals using polynomial approximation. In the former approach the ensemble energy is determined at the resolution of eigenvalues whereas in the latter approach it is determined at the

resolution of a given number of subbands. The direct computation approach has two shortcomings. Firstly, it requires explicit computation of the graph spectrum and the associated eigenvectors; i.e., a full eigendecomposition of the graph Laplacian matrix, which is computationally cumbersome for large graphs and infeasible for extensively large graphs. Secondly, it typically results in a non-smooth description of the ensemble energy. These shortcomings are resolved by using the polynomial approximation scheme.

3.1 Direct Computation: Using the Graph Fourier Transform

Definition (Ensemble Energy Spectral Density)

For a given graph G, with spectrum $\Lambda(G)$, and graph signal set $F = \{\mathbf{f}_s\}_{s=1}^{N_s}$, the ensemble energy spectral density of F is obtained as

$$e_F[l] = \frac{1}{N_s} \sum_{s=1}^{N_s} \left| \widehat{\widetilde{f}}_s[l] \right|^2, \quad l = 1, \ldots, N_g, \tag{34}$$

where $\widetilde{\mathbf{f}}_s$ denotes the de-meaned and normalized version of $\mathbf{f}_s$ obtained as

$$\widetilde{\mathbf{f}}_s = \frac{\mathbf{f}_s - \sum_{r=1}^{1+m_{\lambda_1}} \langle \mathbf{f}_s, \boldsymbol{\chi}_r \rangle \boldsymbol{\chi}_r}{||\mathbf{f}_s - \sum_{r=1}^{1+m_{\lambda_1}} \langle \mathbf{f}_s, \boldsymbol{\chi}_r \rangle \boldsymbol{\chi}_r||_2}, \quad s = 1, \ldots, N_s. \tag{35}$$

The ensemble energy spectral density has the following properties: (i) $\{e_F[r] = 0\}_{r=1}^{1+m_{\lambda_1}}$, and (ii) $\sum_l e_F[l] = 1$.

3.2 Approximation: Using Decomposition Through Polynomial Approximation

The ensemble energy spectral density can be approximated through a multi subband decomposition scheme. In the sequel, we first design a B-spline based system of spectral kernels. The benefit in using a B-spline basis is in the smoothness characteristic of such kernels. Smooth overlapping kernels are advantageous it that (i) they enable obtaining a smooth estimation of the ensemble energy spectral density and (ii) they can be approximated as low order polynomials. We then decompose the graph signals using the designed system of kernels with a large number of subbands by exploiting the polynomial approximation scheme in decomposition. With such a decomposition, we approximate the ensemble spectral content of the signal set at the resolution of subbands.

3.2.1 B-spline Based Parseval Frames on Graphs

The central B-spline of degree n, denoted $\beta^{(n)}(x)$, is a compactly-supported function in the interval $[-\Delta^{(n)}, \Delta^{(n)}]$, i.e., $\beta^{(n)}(x) = 0$ for all $|x| \geq \Delta^{(n)}$ where $\Delta^{(n)} = (n+1)/2$, and is obtained through the $(n+1)$-fold convolution as

$$\beta^{(n)}(x) = \underbrace{\beta^{(0)}(x) * \beta^{(0)}(x) * \cdots \beta^{(0)}(x)}_{(n+1)\text{times}}, \tag{36}$$

where

$$\beta^{(0)}(x) = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ \frac{1}{2}, & |x| = \frac{1}{2} \\ 0, & \text{otherwise.} \end{cases} \tag{37}$$

Proposition 1 (B-spline based Parseval Frame on Graphs) *For a given graph G and B-spline generating function $\beta^{(n)}(x)$, $n \geq 2$, a set of B-spline based spectral kernels $\{B_j(\lambda) \in L_2(G)\}_{j=1}^{J}$ can be defined as*

$$B_j(\lambda) = \begin{cases} \widetilde{B}_j(\lambda) + \sum_{i=-\Delta}^{0} \widetilde{B}_i(\lambda), & j = 1 \\ \widetilde{B}_j(\lambda), & j = 2, \ldots, J-1 \\ \widetilde{B}_j(\lambda) + \sum_{i=J+1}^{J+\Delta+1} \widetilde{B}_i(\lambda), & j = J \end{cases} \tag{38}$$

where $\Delta = \lfloor n/2 \rfloor - 1$ and $\widetilde{B}_\cdot(\lambda) \in L_2(G)$ is defined as

$$\widetilde{B}_l(\lambda) = \sqrt{\beta^{(n)}\left(\frac{\lambda_{max}}{J-1}(\lambda - l + 1)\right)}, \quad l = -\Delta, \ldots, J + \Delta + 1. \tag{39}$$

The system of kernels $\{B_j(\lambda)\}_{j=1}^{J}$ satisfy

$$\sum_{j=1}^{J} |B_j(\lambda)|^2 = 1, \quad \forall \lambda \in [0, \lambda_{max}], \tag{40}$$

and, thus, their associated dictionary of atoms forms a Parseval frame.

Proof See Appendix 1.

Figure 1 shows two realizations of spline-type systems of spectral kernels. The spline-type system of spectral kernels have wide, overlapping passbands. Moreover, the kernels are smooth and can thus be approximated as low order polynomials.

Using a system of N_a B-spline based spectral kernels, $\{B_i(\lambda)\}_{i=1}^{N_a}$, the ensemble spectral energy of F can be approximated at N_a overlapping bands across the spectrum as

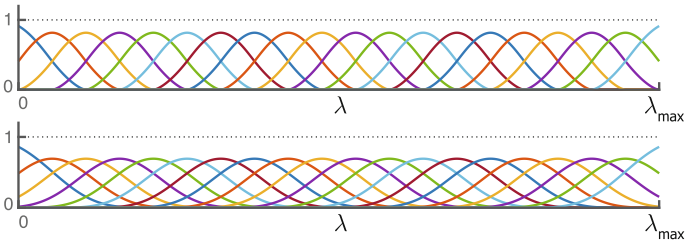


Fig. 1 Spline-type system of spectral kernels with 20 spectral bands constructed based on B-splines of order 3 (top) and 7 (bottom)

$$a_F[i] = \frac{1}{N_s} \sum_{s=1}^{N_s} \sum_{n=1}^{N_g} \left| \langle \widetilde{\mathbf{f}}_s, \boldsymbol{\psi}_{B_i,n} \rangle \right|^2, \quad i = 1, \ldots, N_a, \tag{41}$$

where $\widetilde{\mathbf{f}}_s$ is as given in (35). Let $\mathbf{b}_j \in \ell_2(G)$ denote the discrete version of $B_j(\lambda)$, i.e.,

$$b_j[l] = B_j(\lambda_l), \quad l = 1, \ldots, N_g. \tag{42}$$

We have $\sum_i a_F[i] = 1$ since

$$\sum_i a_F[i] \overset{(24)}{=} \frac{1}{N_s} \sum_{i=1}^{N_a} \sum_{s=1}^{N_s} \sum_{n=1}^{N_g} \left| \sum_{l=1}^{N_g} b_i[l] \widehat{\widetilde{f}}_s[l] \chi_l[n] \right|^2 \tag{43}$$

$$= \frac{1}{N_s} \sum_{s=1}^{N_s} \sum_{n=1}^{N_g} \left| \sum_{l=1}^{N_g} \underbrace{\sum_{i=1}^{N_a} b_i{}^2[l]}_{\overset{(40)}{=} 1} \widehat{\widetilde{\mathbf{f}}}_s[l] \chi_l[n] \right|^2 \tag{44}$$

$$= \frac{1}{N_s} \sum_{s=1}^{N_s} \sum_{n=1}^{N_g} \left| \sum_{l=1}^{N_g} \widehat{\widetilde{f}}_s[l] \chi_l[n] \right|^2 \tag{45}$$

$$\overset{(6)}{=} \frac{1}{N_s} \sum_{s=1}^{N_s} \sum_{n=1}^{N_g} |\widetilde{f}_s[n]|^2 \tag{46}$$

$$= \frac{1}{N_s} \sum_{s=1}^{N_s} ||\widetilde{\mathbf{f}}_s||_2^2 \tag{47}$$

$$\overset{(35)}{=} 1. \tag{48}$$

If desired, an explicit approximation of the ensemble energy spectral density of F, denoted $e_F^{(a)}[l]$, can also be determined. First, a continuous ensemble spectral energy representation, denoted $E_F^{(a)}(\lambda)$, is obtained through interpolating the set of points

$$\left\{(0,0) \cup \left\{\left(\frac{\lambda_{\max}}{C}\sum_{k=1}^{i}||B_k(\lambda)||_2^2\ ,\ a_F[k]\right)\right\}_{i=1}^{N_a}\right\}, \tag{49}$$

where $C = \sum_{k=1}^{N_a} ||B_k(\lambda)||_2^2$. Then, $e_F^{(a)}[l]$ is obtained through sampling $E_F^{(a)}(\lambda)$ at $\Lambda(G)$ as

$$e_F^{(a)}[l] = E_F^{(a)}(\lambda_l), \quad l = 1, \ldots, N_g. \tag{50}$$

4 Signal-Adapted System of Spectral Kernels

The construction of a signal-adapted system of spectral kernels is motivated by two observations: (i) the eigenvalues of the graph Laplacian that define the graph's spectrum are irregularly spaced, and depend in a complex way on the graph topology; (ii) the distribution of graph signals' energy is generally non-uniform across the spectrum. Based on these observations, the idea is to construct an 'adapted' frame, such that the energy-wise significance of the eigenvalues is taken into account, rather than only adapting based on the distribution of the eigenvalues as proposed in [43]. In this way, also the topological information of the graph is implicitly incorporated in the design, since the energy content is given in the graph spectral domain that is in turn defined by the eigenvalues.

For the design of a signal-adapted system of spectral kernels with J subbands, denoted $\{S_j(\lambda)\}_{j=1}^J$, we start off from a prototype system of spectral kernels $\{U_j(\lambda)\}_{j=1}^J$ that satisfies the following two properties:

- (Uniformity constraint)

$$\exists\, C \in \mathbb{R}^+, \quad \int_0^{\lambda_{\max}} U_j(\lambda)d\lambda = C, \quad j = 1, \ldots, J. \tag{51}$$

- (Tight Parseval frame constraint)

$$\sum_{j=1}^{J} |U_j(\lambda)|^2 = 1, \quad \forall\lambda \in [0, \lambda_{\max}]. \tag{52}$$

We then exploit the ensemble energy spectral density $\mathbf{e}_F$ or the approximated ensemble spectral energy $\mathbf{a}_F$ to introduce the desired signal adaptivity. The adaptivity is introduced by first transforming the ensemble spectral energy measures to an energy-equalizing transformation $T_F(\lambda) : [0, \lambda_{\max}] \to [0, \lambda_{\max}]$, which is then in turn incorporated into the prototype design.

4.1 Prototype Uniform System of Spectral Kernels

There is no unique system of kernels that satisfies (51) and (52). We present a design in which the kernels have a finite support of the bandpass type.

Proposition 2 (uniform Meyer-type (UMT) system of spectral kernels) *Using the auxiliary function of the Meyer wavelet, given by [28]*

$$\nu(x) = x^4(35 - 84x + 70x^2 - 20x^3), \tag{53}$$

a set of $J \geq 2$ spectral kernels defined as

$$U_1(\lambda) = \begin{cases} 1 & \forall \lambda \in [0, a] \\ \cos\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda}{a}-1\right)\right)\right) & \forall \lambda \in]a, \gamma a] \\ 0 & \textit{elsewhere} \end{cases} \tag{54a}$$

$$U_j(\lambda) = \begin{cases} \sin\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda-(j-2)\Delta}{a}-1\right)\right)\right) & \forall \lambda \in]\lambda_I, \lambda_{II}] \\ \cos\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda-(j-1)\Delta}{a}-1\right)\right)\right) & \forall \lambda \in]\lambda_{II}, \lambda_{II}+\Delta] \\ 0 & \textit{elsewhere} \end{cases} \tag{54b}$$

$$U_J(\lambda) = \begin{cases} \sin\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda-(J-2)\Delta}{a}-1\right)\right)\right) & \forall \lambda \in]\lambda_I, \lambda_{II}] \\ 1 & \forall \lambda \in]\lambda_{II}, \lambda_{II}+a] \\ 0 & \textit{elsewhere} \end{cases} \tag{54c}$$

can be constructed, where

$$\Delta = \gamma a - a, \tag{55a}$$

$$\lambda_I = a + (j-2)\Delta, \tag{55b}$$

$$\lambda_{II} = \gamma a + (j-2)\Delta, \tag{55c}$$

$$a = \frac{\lambda_{max}}{J\gamma - J - \gamma + 3}. \tag{55d}$$

Figure 2 illustrates the notations used. By setting $\gamma = 2.73$, the set of kernels defined in (54) satisfies the uniformity constraint given in (51). The atoms of a dictionary constructed using this set of spectral kernels form a Parseval frame on $\ell_2(G)$.

Proof See Appendix 2.

Figure 3a and b show realizations of the resulting UMT system of spectral kernels for a fixed $\lambda_{\max}$ and two different J. The UMT system of spectral kernels have a narrow passband characteristic with the support of each kernel being a rather strict subset of the spectrum, with minimal overlap of adjacent kernels.

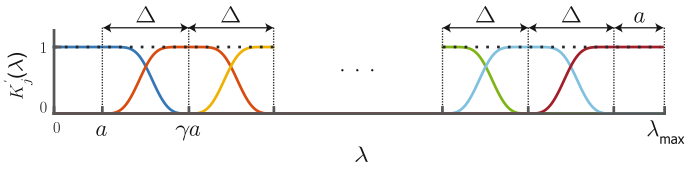


Fig. 2 Construction of UMT system of spectral kernels

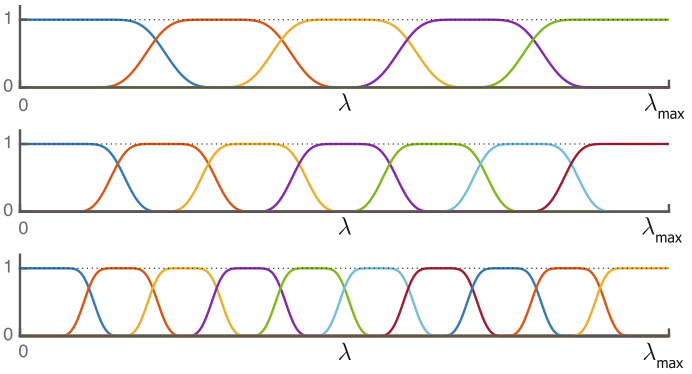


Fig. 3 UMT system of spectral kernels with $J = 5$ (top), $J = 7$ (middle) and $J = 10$ (bottom) spectral scales

4.2 Energy-Equalizing Transformation

If the ensemble spectral density function is available, $T_F(\lambda)$ is obtained through monotonic cubic interpolation [15] of the pair of points

$$\left\{\left(\lambda_l\,,\ \frac{\lambda_{\max}}{m_{\lambda_l}}\sum_{r=i_{\lambda_l}}^{i_{\lambda_l}+m_{\lambda_l}}\sum_{k=1}^{r} e_F[k]\right)\right\}_{l=1}^{N_g}. \tag{56}$$

If the ensemble energy spectral density is approximated using a system of N_a B-spline based spectral kernels (cf. Sect. 3.2), $T_F(\lambda)$ can instead be obtained through monotonic cubic interpolation of the set of points

$$\left\{(0,0)\cup\left\{\left(\frac{\lambda_{\max}}{C}\sum_{k=1}^{i}||B_k(\lambda)||_2^2\,,\ \lambda_{\max}\sum_{k=1}^{i}a_F[k]\right)\right\}_{i=1}^{N_a}\right\}, \tag{57}$$

where $C = \sum_{j=1}^{J}||B_j(\lambda)||_2^2$.

4.3 Warping the Prototype Design

By incorporating $T_F(\lambda)$ in $\{U_j(\lambda)\}_{j=1}^J$, a warped version of the prototype design is obtained as

$$S_j(\lambda) = U_j(T_F(\lambda)), \quad j = 1, \ldots, J. \tag{58}$$

We refer to $\{S_j(\lambda)\}_{j=1}^J$ as a signal-adapted system of spectral kernels. The atoms of a dictionary constructed using $\{S_j(\lambda)\}_{j=1}^J$ form a Parseval frame on $\ell_2(G)$ since

$$\begin{aligned}\sum_{j=1}^{J} |S_j(\lambda)|^2 &\overset{(58)}{=} \sum_{j=1}^{J} |U_j(\underbrace{T_F(\lambda)}_{:=\lambda'})|^2, \quad \forall \lambda \in [0, \lambda_{\max}] \\ &= \sum_{j=1}^{J} |U_j(\lambda')|^2, \quad \forall \lambda' \in [0, \lambda_{\max}] \\ &= 1\end{aligned}$$

where the last equality follows from Proposition 2.

If a discrete representation is needed for direct decomposition as in (24), $\{\boldsymbol{s}_j\}_{j=1}^J$ can be obtained through sampling $S_j(\lambda)$ at $\Lambda(G)$.

With this design, each of the J spectral kernel $\{\boldsymbol{s}_j\}_{j=1}^J$ capture an equal amount of *ensemble* energy. That is, if the ensemble energy spectral density is used we have

$$\sum_{l=1}^{N_g} \boldsymbol{s}_j[l]\boldsymbol{e}_F[l] = \frac{1}{J}, \quad j = 1, \ldots, J, \tag{59}$$

and if the approximation scheme is used we have

$$\sum_{l=1}^{N_g} \boldsymbol{s}_j[l]\boldsymbol{e}_F^{(a)}[l] = \frac{1}{J}, \quad j = 1, \ldots, J. \tag{60}$$

Moreover, the resulting system of spectral kernels form a partition of unity, i.e.,

$$\sum_{j=1}^{J} |\boldsymbol{s}_j[l]|^2 = 1, \quad l = 1, \ldots, N_g, \tag{61}$$

and thus, their associated dictionary of atoms, i.e., $\left\{\{\boldsymbol{\psi}_{S_j,m}\}_{j=1}^J\right\}_{m=1}^{N_g}$, forms a Parseval frame.

5 Example Spectral Designs of Signal-Adapted Tight Frame

We present constructions of signal-adapted systems of spectral kernels for signal sets realized on the Minnesota road graph, the Alameda graph [47] and the cerebellum gray matter graph [1, 2]. Before proceeding to the constructions, let us consider a model for simulating random graph signals of varying smoothness. The model will be used to realize signals on the Minnesota and Alameda graphs, although there exists also real data for the latter graph. For a given graph with adjacency matrix A, we consider a general model for realizing graph signals of density $\eta \in]0, 1]$ and smoothness $n \in \mathbb{Z}^+$ as

$$\mathbf{x}_{\eta,n} = A^n \mathbf{p}_\eta, \tag{62}$$

where $\mathbf{p}_\eta \in \ell_2(G)$ denotes a random realization of a spike signal as $\{\mathbf{p}_\eta[i] \in \{0, 1\}\}_{i=1,\ldots,N_g}$ such that $\sum_i \mathbf{p}_\eta[i] = \eta N_g$. Application of the n-th power of A to $\mathbf{p}_\eta$ leads to a signal that (i) respects the intrinsic structure of the graph and (ii) has a desired smoothness determined by n, a higher n leading to a smoother graph signal.

5.1 The Minnesota Road Graph

The edges of the Minnesota Road Graph represent major roads and its vertices their intersection points, which often correspond to towns or cities, see Fig. 4a. Figure 4b shows the graph's normalized Laplacian spectrum presented as the distribution of the eigenvalues.

Two sets of graph signals were constructed as

$$F_1 = \left\{ \left\{ \mathbf{x}_{\eta,2}^{[i]} \right\}_{\eta=0.2,0.5} \right\}_{i=1,\ldots,10},$$
$$F_2 = \left\{ \left\{ \mathbf{x}_{\eta,4}^{[i]} \right\}_{\eta=0.2,0.5} \right\}_{i=1,\ldots,10},$$

where index i denotes random realizations of $\boldsymbol{p}_\eta$ in (62), resulting in 20 signals in each set. Figure 5a and b show a realizations of a signal from F_1 and F_2, respectively.

Figure 6a shows the energy-equalizing transformation functions associated to F_1 and F_2. The transformations constructed based on $\boldsymbol{a}_{F_\cdot}$, cf. (57) closely matches that constructed based on $\boldsymbol{e}_{F_\cdot}$, cf. (56). The former transformation has the benefit that it is smooth, and indeed, that it was computed without the explicit need to diagonalize L. By incorporating the transformations in the UMT system of spectral kernels, signal-adapted systems of spectral kernels are obtained, see Fig. 6b and c.

A comparison of Figs. 6b, c and 5c, d highlights the energy-wise optimality of the proposed signal-adapted frame construction; i.e., more filters are allocated to spectral ranges that have higher ensemble energy. The support of the filters in the two sets vary

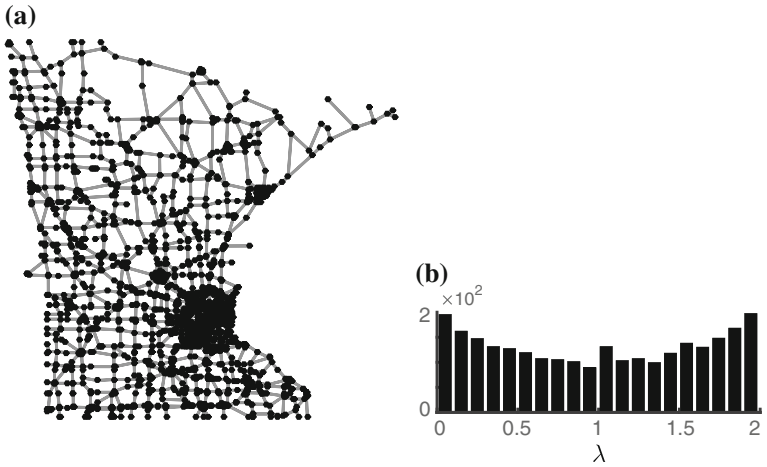


Fig. 4 **a** Minnesota road graph. **b** Histograms of the eigenvalues $\Lambda_{\mathcal{L}}(G)$ of the Minnesota road graph. Each bar indicates the number of eigenvalues that lie in the corresponding spectral range

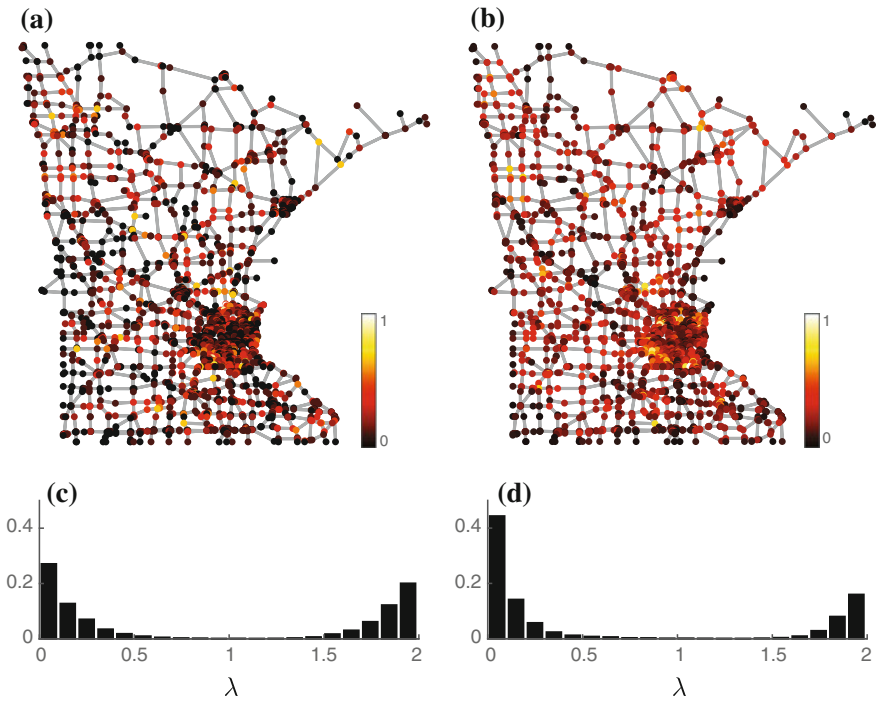


Fig. 5 Sample signal realizations on the Minnesota road graph, **a** $\mathbf{x}_{0.2,2}$ and **b** $\mathbf{x}_{0.5,4}$. The plots are normalized as $\mathbf{x}_{\eta,n}/||\mathbf{x}_{\eta,n}||_\infty$. **c–d** Distribution of the ensemble energy spectral density $\boldsymbol{e}_{F_1}$ and $\boldsymbol{e}_{F_2}$, respectively. Each bar indicates the sum of ensemble energies of the eigenvalues lying in the corresponding spectral range

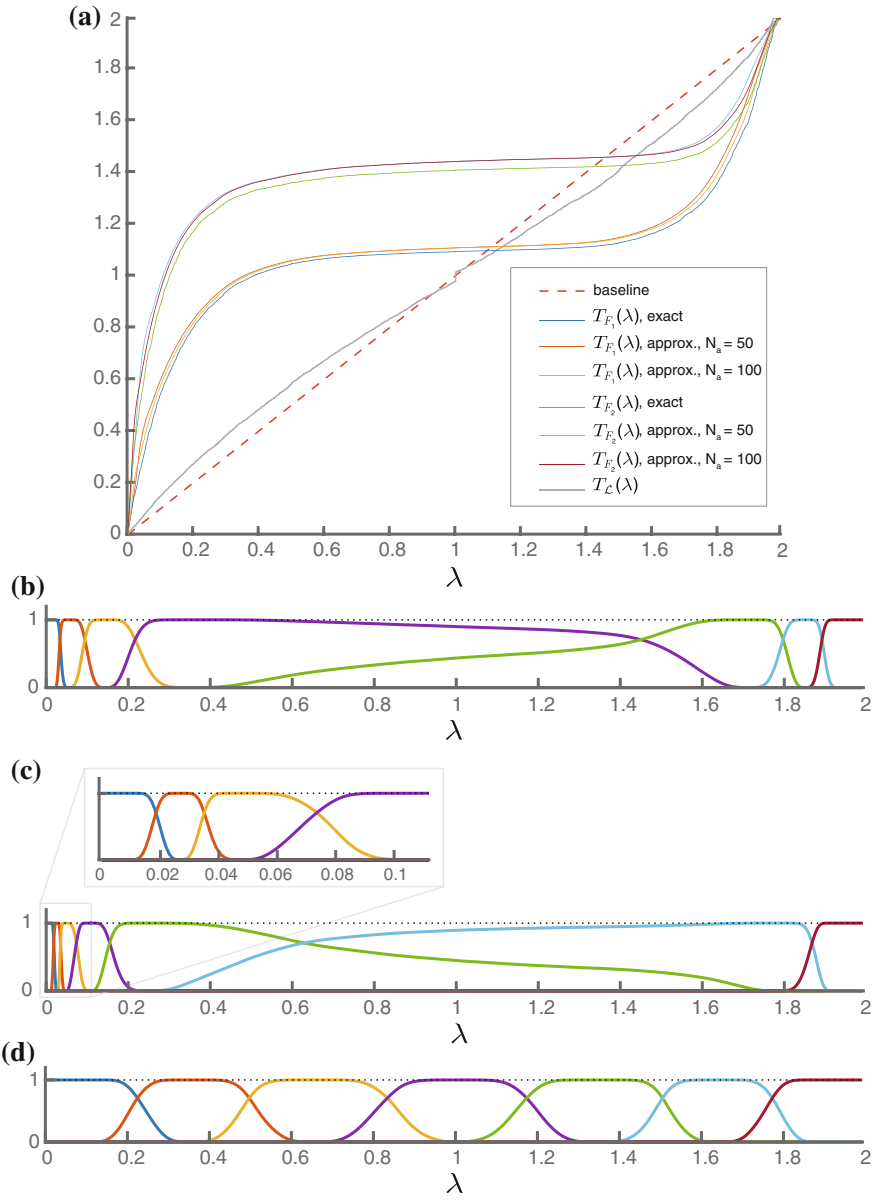


Fig. 6 **a** Constructed energy-equalizing transformation functions, $T_{F_1}(\lambda)$ and $T_{F_2}(\lambda)$ using the exact and approximation schemes. N_a denotes the number of spectral kernels used for the approximation, cf. (41). **b–c** Signal-adapted system of spectral kernels constructed by warping the UMT system of spectral kernels ($J = 7$) using $T_{F_1}(\lambda)$ (approx, $N_a = 100$) and $T_{F_2}(\lambda)$ (approx., $N_a = 100$), respectively. **d** Spectrum-adapted system of spectral kernels constructed by warping the UMT system of spectral kernels ($J = 6$) using $T_{\mathcal{L}}(\lambda)$. In **b–d**, the dashed lines corresponds to the function $G(\lambda)$ in (19)

relative to the difference in the distribution of the ensemble energy of the two signal sets, with more filters allocated to the lower end of the spectrum for the F_2 frame than for the F_1 frame, and vice versa at the upper end of the spectrum. For comparison, a spectrum-adapted system of kernels is shown in Fig. 6d. The spectrum-adapted system of kernels is obtained by warping the UMT prototype system of kernels with a spectrum-equalizing transformation function $T_{\mathcal{L}}(\lambda)$ which equalizes the distribution of the eigenvalues [43]. As the distribution of the eigenvalues of the Minnesota Road graph minimally deviate from a uniform distribution, so does the spectrum-adapted system of kernels relative to the UMT prototype, compare Figs. 3 and 6d. On the contrary, the signal-adapted design optimizes the construction of the kernels such that the energy-wise significance of the eigenvalues is taken into account, rather than only considering the distribution of the eigenvalues as in the spectrum-adapted frame. Such adaptation results in a system of spectral kernels that largely deviate from the UMT prototype.

5.1.1 Robustness to Noise

It is interesting to study the robustness of the design to possible additive noise. Let F_{1,σ_e} denote the noise added version of signal set F_1 computed as

$$F_{1,\sigma_e} = \{\mathbf{y}_i = \mathbf{x}_i + \mathbf{e}_i \mid \mathbf{x}_i \in F_1\}_{i=1,\ldots,20}\,, \tag{63}$$

where $\{\mathbf{e}_i\}_{i=1}^{20}$ denote random realizations of additive white Gaussian noise of standard deviation σ_e. We construct signal sets F_{1,σ_e} of varying SNR $= \sigma_x^2/\sigma_e^2$, where σ_x denotes the standard deviation of each signal $\mathbf{x}_i \in F_1$. Let $T_{F_{1,\sigma_e}}(\lambda)$ denote the energy-equalizing transformation function associated to F_{1,σ_e}. Figure 7a shows mean-square error metrics $||T_{F_{1,\sigma_e}}(\lambda) - T_{F_1}(\lambda)||_2$ and $||T_{F_{1,\sigma_e}}(\lambda) - T_{\mathcal{L}}(\lambda)||_2$ across signal sets F_{1,σ_e} of varying SNR, where $T_{\mathcal{L}}(\lambda)$ and $T_{F_1}(\lambda)$ are the transformation functions shown in Fig. 6a, $T_{F_1}(\lambda)$ being the approximated version using $N_a = 100$. The estimated energy-equalizing transformation functions $T_{F_{1,\sigma_e}}(\lambda)$ become more similar to $T_{F_1}(\lambda)$ as the SNR increases. At low SNRs, $T_{F_{1,\sigma_e}}(\lambda)$ become more similar to $T_{\mathcal{L}}(\lambda)$. The signal-adapted system of spectral kernels using noise-added signal sets of five different SNRs are shown in Fig. 7b. At the two extremes, i.e., $+20$ and -20 dB, the system of kernels become almost identical to the system of kernels shown in Fig. 6b and d, respectively. At 0 dB, the signal-adapted system of kernels at each subband can be seen as the average of the corresponding kernels in the associated subbands in Fig. 6b and d. Equivalently, this can be seen as constructing a system of kernels through warping the the UMT prototype system of kernels with a warping function defined as the average of the spectrum-equalizing and energy-equalizing transformation functions, i.e., $(T_{F_1}(\lambda) + T_{\mathcal{L}}(\lambda))/2$, see Fig. 7b at 0 dB.

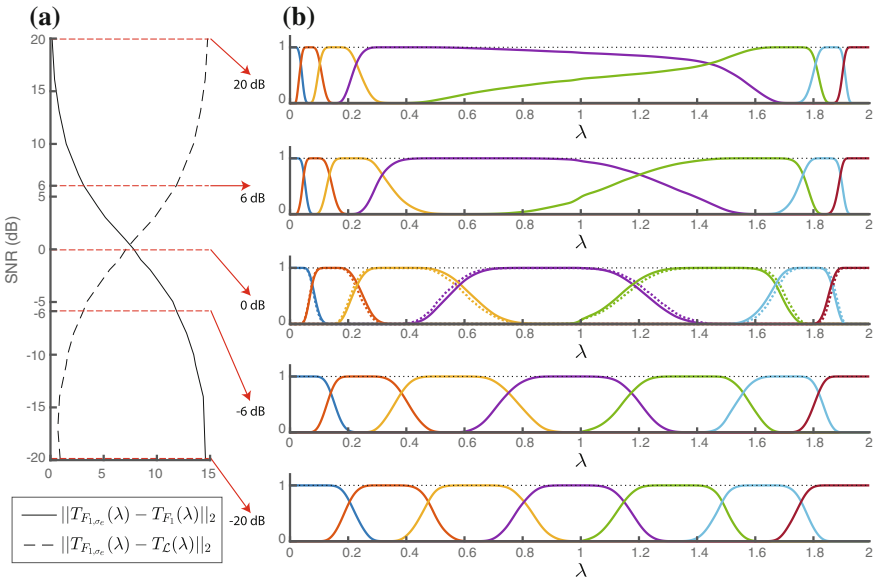


Fig. 7 **a** Deviation of energy-equalizing transformation functions of noise added signal sets $T_{F_{1,\sigma_e}}(\lambda)$ relative to $T_{\mathcal{L}}(\lambda)$ and $T_{F_1}(\lambda)$ (cf. Fig. 6a) as a function of the signal sets' SNRs. **b** Signal-adapted system of spectral kernels constructed by warping the UMT system of spectral kernels ($J = 7$) using $T_{F_{1,\sigma_e}}(\lambda)$ of noise-added signal sets at five different SNRs. At 0 dB, the resulting system of kernels are overlaid on the system of kernels obtained by warping the UMT system of spectral kernels using the transformation function $(T_{F_1}(\lambda) + T_{\mathcal{L}}(\lambda))/2$, shown in dashed lines

5.2 *The Alameda Graph*

The Alameda Graph is constructed based on Caltrans Performance Measurement System database,[1] see Fig. 8a. The vertices of the graph represent detector stations where bottlenecks were identified over the period January 2011 and December 2015. A bottleneck is a location where there is a persistent drop in speed, such as merges, large on-ramps and incidents. Two stations are considered as connected through an edge if either (1) they are adjacent along a freeway, or (2) there is a connection near the two stations at crossings between freeways. The latter type of edges were defined based on Google Maps' satellite images of Alameda county.

We use (62) to simulate a synthetic graph signal set as

$$F_s = \left\{ \mathbf{x}_{0.8,3}^{[i]} \right\}_{i=1,\ldots,20},$$

where index i denotes random realizations of $\boldsymbol{p}_\eta$ in (62), resulting in a set of 20 signals. As real data, we treat the average duration of bottlenecks for each specific

[1] The data are publicly available at http://pems.dot.ca.gov.

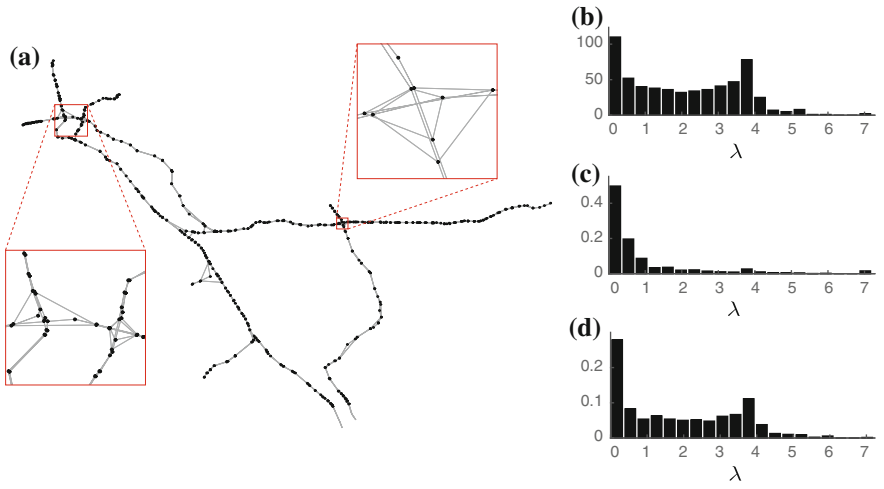


Fig. 8 **a** The Alameda graph. **b** Histogram of the eigenvalues $\Lambda_L(G)$ of the Alameda graph. **c–d** Distribution of the ensemble energy spectral density $\boldsymbol{e}_F$ of the simulated dataset F_s and the real traffic dataset F_r, respectively

month and shift (AM shift: 5–10 am, noon shift: 10 am–3 pm, and PM shift: 3–8 pm) as a graph signal, resulting in 180 signals in total. We denote this dataset as F_r.

The spectral characteristics of both F_s and F_r deviate considerably from that of the Minnesota Road graph. The distribution of the ensemble energy spectral density of F_s emulates an exponential distribution. Comparing the histogram of the eigenvalues $\Lambda_L(G)$ in Fig. 8b and the distribution of the ensemble energy spectral density of F_r in Fig. 8c shows that the ensemble energy is almost uniformly spread across the spectrum.

Figure 9a shows the energy-equalizing transformation functions associated to F_s and F_r. Also, a spectrum-equalizing transformation $T_L(\lambda)$ function is displayed. $T_L(\lambda)$ is constructed such that the distribution of eigenvalues is equalized [43]. Due to the similarity of the distributions of ensemble energy of Fig. 8b (see) and the distribution of eigenvalues (see Fig. 8a), $T_{F_r}(\lambda)$ closely resembles $T_L(\lambda)$. Figure 9b shows the signal-adapted system of spectral kernels associated to F_s. The majority of the spectral kernels are realized in the lower end of the spectrum where the majority of the ensemble energy is present. The zoomed-in inset in Fig. 9b show the benefit of the signal-adapted scheme in allocating a large number of spectral kernels to a narrow band of the spectrum, and yet result in smooth kernels. Figure 9c and d show the signal-adapted system of spectral kernels associated to F_r and the spectrum-adapted system of spectral kernels, respectively. The similarity between $T_{F_r}(\lambda)$ and $T_L(\lambda)$, leads to the resulting signal-adapted and spectrum-adapted systems of kernels having a similar distribution of kernels across the spectrum, with more kernels allocated to the lower half of the spectrum and vice versa. This example demonstrates where the signal-adapted frame design coincides with the spectrum-adapted frame design [43] coincide in terms of their respective approach to adaptivity: if the ensemble spectral

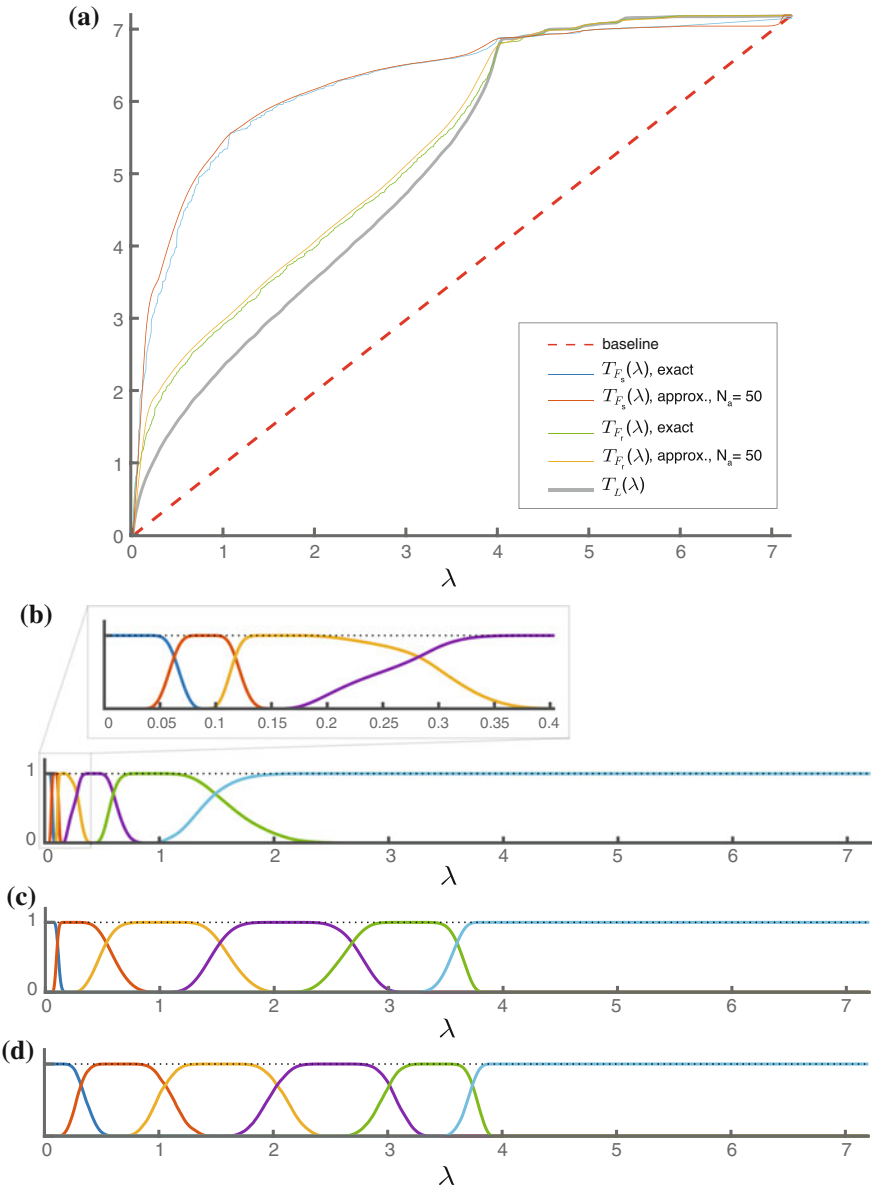


Fig. 9 **a** Energy-equalizing and spectrum-equalizing transformation functions. **b–c** Signal-adapted system of spectral kernels constructed by warping the UMT system of spectral kernels ($J = 6$) using $T_{F_s}(\lambda)$ (approx, $N_a = 50$) and $T_{F_r}(\lambda)$ (approx., $N_a = 50$), respectively. **d** Spectrum-adapted system of spectral kernels constructed by warping the UMT system of spectral kernels ($J = 6$) using $T_{F_L}(\lambda)$. In **b–d**, the dashed lines corresponds to the function $G(\lambda)$ in (19)

energy is equally spread across the eigenvalues, the energy-equalizing and spectrum-equalizing transformation functions become almost identical. Thus, although the signal-adapted design approach is developed based on spectral energy characteristics of a signal set, it is inherently also adapted to the graph's spectrum.

5.3 The Cerebellum Gray Matter Graph

Functional magnetic resonance imaging (fMRI) is a conventional neuroimaging technique used in the study of brain functionality. Its principle is in detecting a contrast that arises as a result of increased blood flow to activated regions of the brain, the so called blood-oxygen-level-dependent (BOLD) signal. Acquired fMRI data are generally corrupted with an extensive amount of noise mainly due to the fast acquisition rate; high temporal resolution is necessary to enable correlate brain activity with the experimental paradigm. The BOLD signal is not detectable across the entire brain tissue. Rather, the signal is only expected within the brain's gray matter [27]. The gray matter is convoluted layer interleaved with the brain's white matter tissue as well as the cerebrospinal fluid. As such, the BOLD signal exhibits spatial patterns that are not well suited to be characterized within a Euclidean setting. In the classical Euclidean setting, filters and wavelets used in image processing are isotropic in structure and quasi shift-invariant. The latter property infers that their structure does not vary when applied to different regions within an image/volume. Such filters are thus not well suited for detecting the BOLD signal, with its aformentioned spatial characteristics. At the spatial resolution of fMRI, isotropically shaped basis functions will cross boundaries of gray matter, even at the finest scale. Thus, it is advantageous to construct filters that adapt to this intricately convoluted domain rather than to assume that the spatial characteristics of the underlying signal is independent of its location. To date, various approaches have been proposed to address this concern (see for example, [9, 19, 25, 33]). In particular, the construction of anatomically-adapted graph wavelets was recently proposed [2]. Yet, the deficiency of a fixed graph frame design and the lack of a systematic approach in determining the spectral coverage of spectral bands for analyzing fMRI data have been pointed out in [1–3]. These findings motivated the need for a frame design that adapts to the spectral characteristics of fMRI graph signals.

We consider a graph that encodes the 3-D topology of the cerebellar gray matter [2], which is constructed based on an atlas template of the cerebellum [12]. The graph vertices represent gray matter voxels within the cerebellum. The graph edges are defined by determining the adjacency of the gray matter voxels within their $3 \times 3 \times 3$ voxel neighbourhood, see Fig. 10. The fMRI data were acquired from 26 healthy subjects performing an event-related visual stimulation task [24].[2] For each subject, a structural MRI scan of the brain anatomy and a series of functional volumes were acquired. The structural and functional volumes were registered together and mapped to the same spatial resolution, leading to a one-to-one

[2]The data are publicly available at https://openfmri.org/dataset/ds000102.

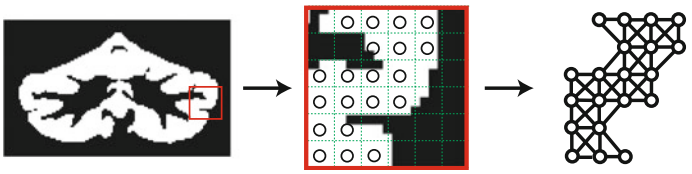


Fig. 10 Illustration of the cerebellum graph

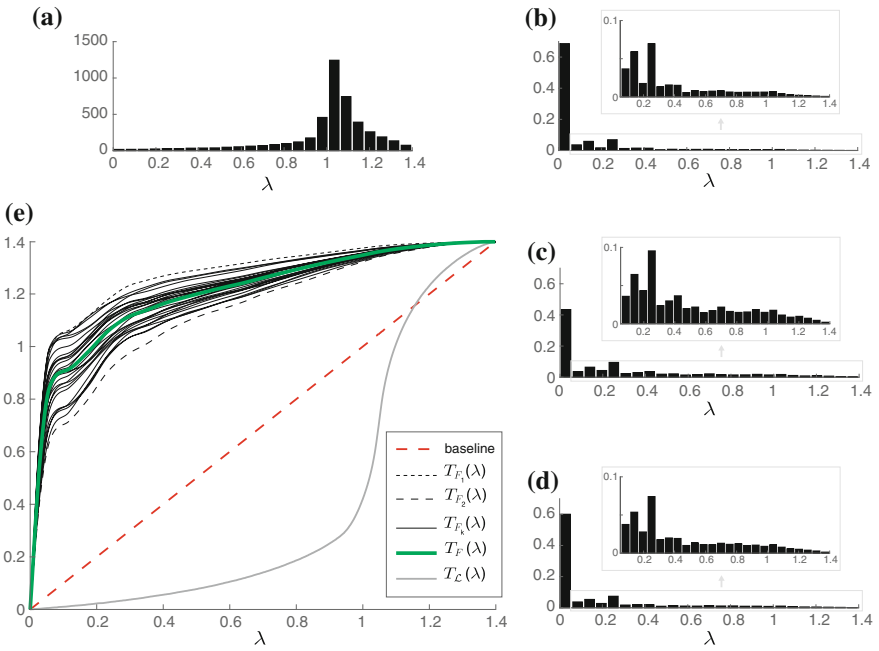


Fig. 11 **a** Histogram of the eigenvalues $\Lambda_{\mathcal{L}}(G)$ of the cerebellum graph. **b–d** Distribution of the ensemble energy spectral density of F_1, F_2 and F. **c** Energy-equalizing and spectrum-equalizing transformation functions. The black curves correspond to the energy-equalizing transformation for each subject's signal set. The upper and lower extreme transformations represented with dashed curves are associated to signal sets F_1 and F_2, respectively

correspondence between functional and structural voxels. Functional voxels associated to cerebellar gray matter were then extracted and treated as cerebellar graph signals. A signal set was constructed for each subject, $\{F_k\}_{k=1}^{26}$, by randomly selecting 20 signals from each subject's functional signal set. A signal set including the signals from all subjects was also constructed as $F = F_1 \cup F_2 \cup \cdots \cup F_{26}$.

Figure 11a shows the distribution of the eigenvalues $\Lambda_{\mathcal{L}}(G)$ of the cerebellum gray matter graph. The distribution of the ensemble energy spectral density of signals sets F_1, F_2 and F are shown in Fig. 11b, c and d, respectively. The distribution of eigenvalues is significantly different from that of the ensemble energy spectral densities; most eigenvalues are located at the upper end of the spectrum, whereas the ensemble energy is significantly concentrated at the lower end of the spectrum. The

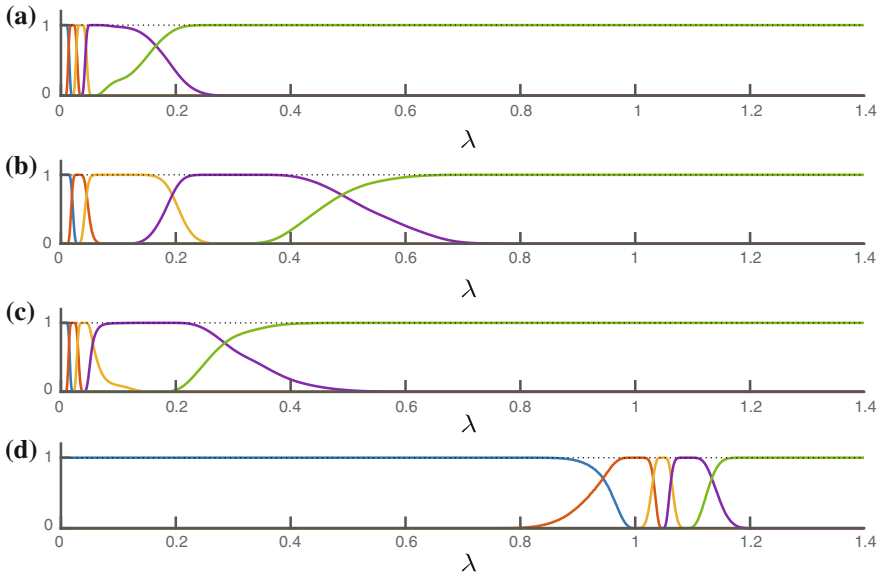


Fig. 12 **a–c** Signal-adapted system of spectral kernels adapted to the ensemble spectral content of F_1, F_2 and F, respectively. **d** Spectrum-adapted system of spectral kernels

ensemble energy spectral densities also vary across the signal sets. Signal set F_1 has more low energy spectral content than F_2 (compare the height of the first bins of the histograms in Fig. 11b and c), whereas F_2 show greater spectral content at higher harmonics. F_1 and F_2 represent the two extremes in spectral content distribution among the 26 subjects. The distribution of the ensemble energy content of F falls in between that of F_1 and F_2, see Fig. 11d. This is better observed by comparing the energy-equalizing transformation functions, see Fig. 11e. The transformations associated to $\{F_k\}_{k=3}^{26}$ span the space in between $T_{F_1}(\lambda)$ and $T_{F_2}(\lambda)$, and $T_F(\lambda)$ falls almost in the mid range. Moreover, the significant difference between the distribution of the eigenvalues and that of the ensemble signal energies is reflected as a major discrepancy between $T_{\mathcal{L}}(\lambda)$ and the energy equalizing transformations. Figure 12 shows the resulting signal-adapted and spectrum-adapted systems of spectral kernels.

The kernels of the spectrum-adapted frame are localized at the higher end of the spectrum where a significant proportion of the eigenvalues fall. In contrast, kernels of the signal-adapted frames are localized at the lower end of the spectrum. This shows that the signal-adapting scheme leads to an optimal configuration of filters relative to the given ensemble energy content. Laplacian eigenmodes corresponding to large eigenvalues tend to become localized and less stable (i.e., influenced by small changes to the structure of the graph). The ensemble energy will capture the consistency of the energy for each mode across signals of the class, and thus these eigenmodes will in practice aggregate in larger subbands. The narrowband configuration of the proposed signal-adapted frame at the lower end of the spectrum closely resembles the design previously adopted for analyzing cerebellar data in [1, 2], which was obtained by empirically tuning the spectral design of the Meyer-like graph wavelet frame [26].

6 Conclusion and Outlook

A scheme for the spectral design of signal-adapted frames on graphs was presented. The scheme exploits the ensemble energy spectral density of a given signal class to introduce adaptivity of the spectral kernels to signal content. The design only uses stationary signal information, with a flexibility to represent non-stationary features based on the width and smoothness of the bandpass characteristics. The design has been formulated on the graph Laplacian spectrum but can be readily extended to the spectrum of the graph adjacency matrix to enable signal-adapted decomposition of signals defined on directed graphs. Various potential applications can be envisioned for the proposed developments. For instance, in functional brain imaging, another major research theme where graph signal processing can be advantageous is the study of intrinsic brain activity that fully takes into account the dynamic aspects [21]. In such case, the moment-to-moment functional data can be analyzed on a graph "backbone" [21, 23]. Time-dependent functional data can then be used to constitute the ensemble energy spectral density. As alternative avenues, signal decompositions provided by the proposed signal-adapted system of kernels can also be found beneficial in applications such as graph signal compression [39] and deep neural network learning schemes over graphs [6, 11].

Acknowledgements This chapter draws in part on material previously published in [4].

Appendix 1 - Proof of Proposition 1

The sum of squared magnitudes of B-spline based spectral kernels $\{B_j(\lambda)\}_{j=1}^{J}$ forms a partition of unity since

$$\begin{aligned}\sum_{j=1}^{J}|B_j(\lambda)|^2 &\overset{(38)}{=} \sum_{i=\Delta}^{J+\Delta+1}|\widetilde{B}_i(\lambda)|^2\\ &\overset{(39)}{=} \sum_{i=\Delta}^{J+\Delta+1}\beta^{(n)}\left(\frac{\lambda_{\max}}{J-1}(\lambda-i+1)\right)\\ &\overset{i-1\to k}{=} \sum_{k=\Delta-1}^{J+\Delta}\beta^{(n)}\left(\frac{\lambda_{\max}}{J-1}(\lambda-k)\right)\\ &= 1.\end{aligned}$$

where in the last equality we use the property that integer shifted splines form a partition of unity.

Appendix 2 - Proof of Proposition 2

In order to ensure that the spectral kernels cover the full spectrum, a must be chosen such that

$$\lambda_{\max} \overset{(54c)}{=} \lambda_{\mathrm{II}} + a \overset{(j=J)}{=} \gamma a + (J-2)\Delta + a,$$

which using (55a) leads to $a = \frac{\lambda_{\max}}{J\gamma - J - \gamma + 3}$.

To prove that the UMT system of spectral kernels form a tight frame, (21) needs to be fulfilled. Since, for all j, the supports of $U_{j-1}(\lambda)$ and $U_{j+1}(\lambda)$ are disjoint, $G(\lambda)$ can be determined as

$$\begin{aligned}
G(\lambda) &= \sum_{j=1}^{J} |U_j(\lambda)|^2 \\
&\overset{(54)}{=} \begin{cases} |U_1(\lambda)|^2 \overset{(54a)}{=} 1 & \forall \lambda \in [0, a] \\ |U_1(\lambda)|^2 + |U_2(\lambda)|^2 & \forall \lambda \in]a, \gamma a] \\ |U_2(\lambda)|^2 + |U_3(\lambda)|^2 & \forall \lambda \in]\gamma a, \gamma a + \Delta] \\ \vdots & \vdots \\ |U_J(\lambda)|^2 \overset{(54c)}{=} 1 & \forall \lambda \in]\lambda_{\max} - a, \lambda_{\max}] \end{cases} \\
&\overset{(54b)}{=} \begin{cases} 1 & \forall \lambda \in [0, a] \\ \cos^2(x_{\mathrm{I}}) + \sin^2(x_{\mathrm{I}}) & \forall \lambda \in]a, \gamma a] \\ \cos^2(x_{\mathrm{II}}) + \sin^2(x_{\mathrm{II}}) & \forall \lambda \in]\gamma a, \gamma a + \Delta] \\ \vdots & \vdots \\ 1 & \forall \lambda \in]\lambda_{\max} - a, \lambda_{\max}] \end{cases} \\
&= 1 \quad \forall \lambda \in [0, \lambda_{\max}]
\end{aligned} \tag{64}$$

where $x_{\mathrm{I}} = \frac{\pi}{2}\nu(\frac{1}{\gamma-1}(\frac{\lambda}{a} - 1))$ and $x_{\mathrm{II}} = \frac{\pi}{2}\nu(\frac{1}{\gamma-1}(\frac{\lambda-\Delta}{a} - 1))$.

For any given γ, the constructed set of spectral kernels form a tight frame. However, in order for the frame to satisfy the uniformity constraint given in (51), the appropriate γ needs to be determined. From (54b), we have $\forall j \in \{2, \ldots, J-2\}$

$$U_j(\lambda) = U_{j+1}(\lambda + \Delta) \quad \forall \lambda \in]\lambda_{\mathrm{I}}, \lambda_{\mathrm{II}} + \Delta]. \tag{65}$$

By considering an inverse linear mapping of the spectral support where $U_1(\lambda) \neq 0$, i.e. $[0, \gamma a]$, to the spectral support where $U_J(\lambda) \neq 0$, i.e. $[\lambda_{\max} - \gamma a, \lambda_{\max}]$, we have

$$U_1(\lambda) = U_J(-\lambda + 2a + J\Delta) \quad \forall \lambda \in [0, \gamma a]. \tag{66}$$

Thus, from (65) and (66) we have

$$\int_0^{\lambda_{\max}} U_j(\lambda)d\lambda = C_2, \quad j = 2, \ldots, J-1 \tag{67a}$$

$$\int_0^{\lambda_{\max}} U_1(\lambda)d\lambda = \int_0^{\lambda_{\max}} U_J(\lambda)d\lambda = C_1, \tag{67b}$$

respectively, where $C_1, C_2 \in \mathbb{R}^+$. Thus, in order to satisfy (51), γ should be chosen such that

$$\begin{aligned} C_1 &= C_2 \\ \int_0^{\lambda_{\max}} U_1(\lambda)d\lambda &= \int_0^{\lambda_{\max}} U_2(\lambda)d\lambda \\ a + \int_a^{\gamma a} U_1(\lambda)d\lambda &= \int_a^{\gamma a} \sin\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda}{a}-1\right)\right)\right)d\lambda \\ &\quad + \int_{\gamma a}^{\gamma a+\Delta} U_2(\lambda)d\lambda \\ a &\overset{(65)}{=} \int_a^{\gamma a} \sin\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda}{a}-1\right)\right)\right)d\lambda. \end{aligned} \tag{68}$$

The optimal γ that satisfies (68) was obtained numerically by defining

$$Q(\gamma) = \int_a^{\gamma a} \sin\left(\frac{\pi}{2}\nu\left(\frac{1}{\gamma-1}\left(\frac{\lambda}{a}-1\right)\right)\right)d\lambda - a, \tag{69}$$

and discretizing $Q(\gamma)$ within the range $(a, \gamma a]$, with a sampling factor of 1×10^{-4}. Testing for $\gamma \geq 1$, with a step size of 1×10^{-2}, the optimal value, which is independent of $\lambda_{\max}$ and J, was found to be $\gamma = 2.73$.

References

1. H. Behjat, N. Leonardi, L. Sörnmo, D. Van De Ville, Canonical cerebellar graph wavelets and their application to fMRI activation mapping, in *Proceedings of the IEEE International Conference of the Engineering in Medicine and Biology Society* (2014), pp. 1039–1042
2. H. Behjat, N. Leonardi, L. Sörnmo, D. Van De Ville, Anatomically-adapted graph wavelets for improved group-level fMRI activation mapping. Neuroimage **123**, 185–199 (2015)
3. H. Behjat, N. Leonardi, D. Van De Ville, Statistical parametric mapping of functional MRI data using wavelets adapted to the cerebral cortex, in *Proceedings of the International Symposium on Biomedical Imaging* (2013), pp. 1070–1073
4. H. Behjat, U. Richter, D. Van De Ville, L. Sörnmo, Signal-adapted tight frames on graphs. IEEE Trans. Signal Process. **64**(22), 6017–6029 (2016)
5. J. Benedetto, M. Fickus, Finite normalized tight frames. Adv. Comput. Math. **18**(2), 357–385 (2003)
6. M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst, Geometric deep learning: going beyond Euclidean data. IEEE Signal Process. Mag. **34**(4), 18–42 (2017). https://doi.org/10.1109/MSP.2017.2693418

7. S. Chen, R. Varma, A. Sandryhaila, J. Kovačević, Discrete signal processing on graphs: sampling theory. IEEE Trans. Signal Process. **63**(24), 6510–6523 (2015)
8. F. Chung, *Spectral Graph Theory* (AMS, Providence, 1997)
9. M.K. Chung, S.M. Robbins, K.M. Dalton, R.J. Davidson, A.L. Alexander, A.C. Evans, Cortical thickness analysis in autism with heat kernel smoothing. Neuroimage **25**(4), 1256–1265 (2005)
10. R.R. Coifman, M. Maggioni, Diffusion wavelets. Appl. Comput. Harmon. Anal. **21**(1), 53–94 (2006)
11. M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in *Advances in Neural Information Processing Systems* (2016), pp. 3844–3852
12. J. Diedrichsen, J.H. Balsters, J. Flavell, E. Cussans, N. Ramnani, A probabilistic MR atlas of the human cerebellum. Neuroimage **46**(1), 39–46 (2009)
13. B. Dong, Sparse representation on graphs by tight wavelet frames and applications. Appl. Comput. Harmon. Anal. (2015). https://doi.org/10.1016/j.acha.2015.09.005
14. M. Fiedler, Algebraic connectivity of graphs. Czechoslov. Math. J. **23**(2), 298–305 (1973)
15. F.N. Fritsch, R.E. Carlson, Monotone piecewise cubic interpolation. SIAM J. Numer. Anal. **17**(2), 238–246 (1980)
16. M. Gavish, B. Nadler, R.R. Coifman, Multiscale wavelets on trees, graphs and high dimensional data: theory and applications to semi supervised learning, in *Proceedings of the International Conference on Machine Learning* (2010), pp. 367–374
17. M. Girvan, M.E. Newman, Community structure in social and biological networks. Proc. Natl. Acad. Sci. **99**(12), 7821–7826 (2002)
18. F. Göbel, G. Blanchard, U. von Luxburg, Construction of tight frames on graphs and application to denoising (2014), http://arXiv.org/abs/1408.4012
19. D.J. Hagler Jr., A.P. Saygin, M.I. Sereno, Smoothing and cluster thresholding for cortical surface-based group analysis of fMRI data. Neuroimage **33**(4), 1093–1103 (2006)
20. D. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
21. W. Huang, T.A.W. Bolton, J.D. Medaglia, D.S. Bassett, A. Ribeiro, D.V.D. Ville, A graph signal processing perspective on functional brain imaging. Proc. IEEE **106**(5), 868–885 (2018). https://doi.org/10.1109/JPROC.2018.2798928
22. M. Jansen, G.P. Nason, B.W. Silverman, Multiscale methods for data on graphs and irregular multidimensional situations. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **71**(1), 97–125 (2009)
23. F.I. Karahanoğlu, D. Van De Ville, Dynamics of large-scale fMRI networks: deconstruct brain activity to build better models of brain function. Curr. Opin. Biomed. Eng. **3**, 28–36 (2017). https://doi.org/10.1016/j.cobme.2017.09.008
24. A.M.C. Kelly, L.Q. Uddin, B.B. Biswal, F.X. Castellanos, M.P. Milham, Competition between functional brain networks mediates behavioral variability. Neuroimage **39**(1), 527–537 (2008)
25. S.J. Kiebel, R. Goebel, K.J. Friston, Anatomically informed basis functions. Neuroimage **11**(6), 656–667 (2000)
26. N. Leonardi, D. Van De Ville, Tight wavelet frames on multislice graphs. IEEE Trans. Signal Process. **61**(13), 3357–3367 (2013)
27. N. Logothetis, B. Wandell, Interpreting the BOLD signal. Annu. Rev. Physiol. **66**, 735–769 (2004)
28. Y. Meyer, Principe d'incertitude, bases hilbertiennes et algebres d'operateurs. Seminaire Bourbaki (in French) **662**, 209–223 (1986)
29. S.K. Narang, A. Ortega, Lifting based wavelet transforms on graphs, in *Proceedings of the APSIPA ASC* (2009), pp. 441–444
30. S.K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Trans. Signal Process. **60**(6), 2786–2799 (2012)
31. M. Newman, *Networks* (OUP, Oxford, 2010)
32. A. Ortega, P. Frossard, J. Kovačević, J.M.F. Moura, P. Vandergheynst, On the optimality of ideal filters for pyramid and wavelet signal approximation. Proc. IEEE **106**(5), 808–828 (2018)

33. S. Ozkaya, D. Van De Ville, Anatomically adapted wavelets for integrated statistical analysis of fMRI data, in *Proceedings of the IEEE International Symposium on Biomedical Imaging, Chicago, IL* (2011), pp. 469–472
34. I. Ram, M. Elad, I. Cohen, Generalized tree-based wavelet transform. IEEE Trans. Signal Process. **59**(9), 4199–4209 (2011)
35. I. Ram, M. Elad, I. Cohen, Redundant wavelets on graphs and high dimensional data clouds. IEEE Signal Process. Lett. **19**(5), 291–294 (2012)
36. R. Rustamov, L. Guibas, Wavelets on graphs via deep learning, in *Proceedings of the Advances in Neural Information Processing Systems* (2013), pp. 998–1006
37. A. Sakiyama, Y. Tanaka, Oversampled graph Laplacian matrix for graph filter banks. IEEE Trans. Signal Process. **62**(24), 6425–6437 (2014)
38. A. Sandryhaila, J. Moura, Discrete signal processing on graphs. IEEE Trans. Signal Process. **61**, 1644–1656 (2013)
39. A. Sandryhaila, J.M. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process. Mag. **31**(5), 80–90 (2014)
40. D.I. Shuman, M.J. Faraji, P. Vandergheynst, A multiscale pyramid transform for graph signals. IEEE Trans. Signal Process. **64**(8), 2119–2134 (2016)
41. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
42. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. (2015). https://doi.org/10.1016/j.acha.2015.02.005
43. D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames. IEEE Trans. Signal Process. **63**(16), 4223–4235 (2015)
44. L. Stankovic, M. Dakovic, E. Sejdic, Vertex-frequency analysis: a way to localize graph spectral components [lecture notes]. IEEE Signal Process. Mag. **34**(4), 176–182 (2017)
45. Y. Tanaka, A. Sakiyama, M-channel oversampled graph filter banks. IEEE Trans. Signal Process. **62**(14), 3578–3590 (2014)
46. D.B.H. Tay, J. Zhang, Techniques for constructing biorthogonal bipartite graph filter banks. IEEE Trans. Signal Process. **63**(21), 5772–5783 (2015). https://doi.org/10.1109/TSP.2015.2460216
47. D. Thanou, D.I. Shuman, P. Frossard, Learning parametric dictionaries for signals on graphs. IEEE Trans. Signal Process. **62**(15), 3849–3862 (2014)
48. N. Tremblay, P. Borgnat, Subgraph-based filterbanks for graph signals. IEEE Trans. Signal Process. **64**(15), 3827–3840 (2016)
49. D. Van De Ville, R. Demesmaeker, M.G. Preti, When slepian meets fiedler: putting a focus on the graph spectrum. IEEE Signal Process. Lett. **24**(7), 1001–1004 (2017)
50. U. Von Luxburg, A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
51. Y. Yankelevsky, M. Elad, Dual graph regularized dictionary learning. IEEE Trans. Signal Inf. Process. Netw. **2**(4), 611–624 (2016)

Wavelets on Graphs via Deep Learning



Raif M. Rustamov and Leonidas J. Guibas

Abstract An increasing number of applications require processing of signals defined on weighted graphs. While wavelets provide a flexible tool for signal processing in the classical setting of regular domains, the existing graph wavelet constructions are less flexible—they are guided solely by the structure of the underlying graph and do not take directly into consideration the particular class of signals to be processed. This chapter introduces a machine learning framework for constructing graph wavelets that can sparsely represent a given class of signals. Our construction uses the lifting scheme, and is based on the observation that the recurrent nature of the lifting scheme gives rise to a structure resembling a deep auto-encoder network. Particular properties that the resulting wavelets must satisfy determine the training objective and the structure of the involved neural networks. The training is unsupervised, and is conducted similarly to the greedy pre-training of a stack of auto-encoders. After training is completed, we obtain a linear wavelet transform that can be applied to any graph signal in time and memory linear in the size of the graph. Improved sparsity of our wavelet transform for the test signals is confirmed via experiments both on synthetic and real data.

1 Introduction

Processing of signals on graphs has emerged as a fundamental problem in an increasing number of applications [1]. Indeed, in addition to providing a direct representation of a variety of networks arising in practice, graphs serve as an overarching abstraction

This work was done at Computer science department, Stanford University.

R. M. Rustamov (✉)
Data Science and AI Research, AT&T Labs Research, 1 AT&T Way, Bedminster, NJ 07931, USA
e-mail: raifrustamov@gmail.com

R. M. Rustamov · L. J. Guibas
Computer Science Department, Stanford University, 353 Serra Mall, Stanford, CA 94305, USA
e-mail: guibas@stanford.edu

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_5

for many other types of data. High-dimensional data clouds such as a collection of handwritten digit images, volumetric and connectivity data in medical imaging, laser scanner acquired point clouds and triangle meshes in computer graphics—all can be abstracted using weighted graphs. Given this generality, it is desirable to extend the flexibility of classical tools such as wavelets to the processing of signals defined on weighted graphs.

A number of approaches for constructing wavelets on graphs have been proposed, including, but not limited to the Crovella-Kolaczyk Wavelet Transform (CKWT) [2], Haar-like wavelets [3, 4], diffusion wavelets [5], spectral wavelets [6], tree-based wavelets [7], average-interpolating wavelets [8], and separable filterbank wavelets [9]. However, all of these constructions are guided solely by the structure of the underlying graph, and do not take directly into consideration the particular class of signals to be processed. While this information can be incorporated indirectly when building the underlying graph (e.g. [7, 9]), such an approach does not fully exploit the degrees of freedom inherent in wavelet design. In contrast, a variety of signal class specific and adaptive wavelet constructions exist on images and multidimensional regular domains, see [10] and references therein. Bridging this gap is challenging because obtaining graph wavelets, let alone adaptive ones, is complicated by the irregularity of the underlying space. In addition, theoretical guidance for such adaptive constructions is lacking as it remains largely unknown how the properties of the graph wavelet transforms, such as sparsity, relate to the structural properties of graph signals and their underlying graphs [1].

The goal of our work is to provide a machine learning framework for constructing wavelets on weighted graphs that can sparsely represent a given class of signals. Our construction uses the lifting scheme as applied to the Haar wavelets, and is based on the observation that the update and predict steps of the lifting scheme are similar to the encode and decode steps of an auto-encoder. From this point of view, the recurrent nature of the lifting scheme gives rise to a structure resembling a deep auto-encoder network.

Particular properties that the resulting wavelets must satisfy, such as sparse representation of signals, local support, and vanishing moments, determine the training objective and the structure of the involved neural networks. The goal of achieving sparsity translates into minimizing a sparsity surrogate of the auto-encoder reconstruction error. Vanishing moments and locality can be satisfied by tying the weights of the auto-encoder in a special way and by restricting receptive fields of neurons in a manner that incorporates the structure of the underlying graph. The training is unsupervised, and is conducted similarly to the greedy (pre-)training [11–14] of a stack of auto-encoders.

The advantages of our construction are three-fold. First, when no training functions are specified by the application, we can impose a smoothness prior and obtain a novel general-purpose wavelet construction on graphs. Second, our wavelets are adaptive to a class of signals and after training we obtain a linear transform; this is in contrast to adapting to the input signal (e.g. by modifying the underlying graph [7, 9]) which effectively renders those transforms non-linear. Third, our construction

provides efficient and exact analysis and synthesis operators and results in a critically sampled basis that respects the multiscale structure imposed on the underlying graph.

The work presented in this chapter first appeared as the paper [15]. To the best of our knowledge, this was among the earliest work to utilize neural networks over non-standard domains, second only to the recurrent neural networks formulation in [16]. Since then, there has been growing interest in deep learning over non-standard domains both in machine learning and computer vision/graphics. An extensive overview of this field called Geometric Deep Learning can be found in [17].

This chapter is organized as follows: in Sect. 2 we briefly overview the lifting scheme. Next, in Sect. 3 we provide a general overview of our approach, and fill in the details in Sect. 4. Finally, we present a number of experiments on synthetic and real data in Sect. 5.

2 Lifting Scheme

The goal of wavelet design is to obtain a multiresolution [18] of $L^2(G)$—the set of all functions/signals on graph G. Namely, a nested sequence of approximation spaces from coarse to fine of the form $\mathbf{V}_1 \subset \mathbf{V}_2 \subset ... \subset \mathbf{V}_{\ell_{max}} = L^2(G)$ is constructed. Projecting a signal in the spaces $\mathbf{V}_\ell$ provides better and better approximations with increasing level ℓ. Associated wavelet/detail spaces $\mathbf{W}_\ell$ satisfying $\mathbf{V}_{\ell+1} = \mathbf{V}_\ell \oplus \mathbf{W}_\ell$ are also obtained, where $\oplus$ is used to denote the direct sum of vector spaces.

Scaling functions $\{\phi_{\ell,k}\}$ provide a basis for approximation space $\mathbf{V}_\ell$, and similarly wavelet functions $\{\psi_{\ell,k}\}$ for $\mathbf{W}_\ell$. As a result, for any signal $f \in L^2(G)$ on graph and any level $\ell_0 < \ell_{max}$, we have the wavelet decomposition

$$f = \sum_k a_{\ell_0,k}\phi_{\ell_0,k} + \sum_{\ell=\ell_0}^{\ell_{max}-1} \sum_k d_{\ell,k}\psi_{\ell,k}. \tag{1}$$

The coefficients $a_{\ell,k}$ and $d_{\ell,k}$ appearing in this decomposition are called approximation (also, scaling) and detail (also, wavelet) coefficients respectively. For simplicity, we use a_ℓ and d_ℓ to denote the vectors of all approximation and detail coefficients at level ℓ.

Our construction of wavelets is based on the lifting scheme [19]. Starting with a given wavelet transform, which in our case is the Haar transform (HT), one can obtain lifted wavelets by applying the process illustrated in Fig. 1 (left) starting with $\ell = \ell_{max} - 1$, $a_{\ell_{max}} = f$ and iterating down until $\ell = 1$. At every level the lifted coefficients a_ℓ and d_ℓ are computed by augmenting the Haar coefficients $\bar{a}_\ell$ and $\bar{d}_\ell$ (of the lifted approximation coefficients $a_{\ell+1}$) as follows

$$\begin{aligned} a_\ell &\leftarrow \bar{a}_\ell + U\bar{d}_\ell \\ d_\ell &\leftarrow \bar{d}_\ell - Pa_\ell \end{aligned}$$

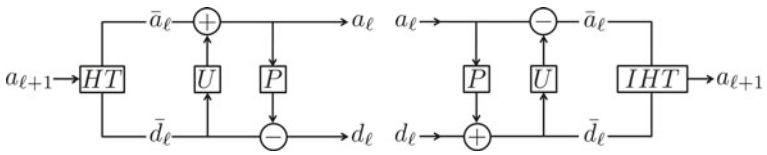


Fig. 1 Lifting scheme: one step of forward (left) and backward (right) transform. Here, a_ℓ and d_ℓ denote the vectors of all approximation and detail coefficients of the lifted transform at level ℓ. U and P are linear update and predict operators. HT and IHT are the Haar transform and its inverse

where update (U) and predict (P) are linear operators (matrices). Note that in adaptive wavelet designs such as ours, the update and predict operators will vary from level to level, but for simplicity of notation we do not indicate this explicitly.

This process is always invertible—the backward transform is depicted, with IHT being the inverse Haar transform, in Fig. 1 (right) and allows obtaining perfect reconstruction of the original signal. While the wavelets and scaling functions are not explicitly computed during either forward or backward transform, it is possible to recover them using the expansion of Eq. (1). For example, to obtain a specific scaling function $\phi_{\ell,k}$, one simply sets all of approximation and detail coefficients to zero, except for $a_{\ell,k} = 1$ and runs the backward transform.

3 Approach

For a given class of signals, our objective is to design wavelets that yield approximately sparse expansions in Eq.(1)—i.e. the detail coefficients are mostly small with a tiny fraction of large coefficients. Therefore, we learn the update and predict operators that minimize some sparsity surrogate of the detail (wavelet) coefficients of given training functions $\{f^n\}_{n=1}^{n_{max}}$.

For a fixed multiresolution level ℓ, and a training function f^n, let $\bar{a}_\ell^n$ and $\bar{d}_\ell^n$ be the Haar approximation and detail coefficient vectors of f^n received at level ℓ (i.e. applied to $a_{\ell+1}^n$as in Fig. 1 (left)). Consider the minimization problem

$$\{U, P\} = \arg\min_{U,P} \sum_n s(d_\ell^n) = \arg\min_{U,P} \sum_n s(\bar{d}_\ell^n - P(\bar{a}_\ell^n + U\bar{d}_\ell^n)), \qquad (2)$$

where s is some sparsity inducing penalty function, such as the L_1-norm. This can be seen as optimizing a linear auto-encoder with encoding step given by $\bar{a}_\ell^n + U\bar{d}_\ell^n$, and decoding step given by multiplication with the matrix P, see Fig 2. Since we would like to obtain a linear wavelet transform, the linearity of the encode and decode steps is of crucial importance. In addition to linearity and the special form of bias terms, our auto-encoders differ from commonly used ones in that *we enforce sparsity on the reconstruction error*, not on the hidden representation. Indeed, in our setting, the

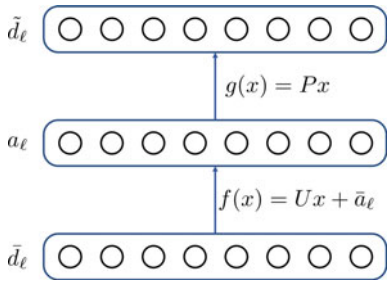


Fig. 2 Optimizing the sparsity of the wavelet transform leads to an auto-encoder training problem over the graph. As with general auto-encoders, the neural network weight matrices U and P are trained to make the reconstructed vector $\tilde{d}_\ell$ close to the input vector $\bar{d}_\ell$. Interestingly, optimizing a sparse penalty of the reconstruction error, such as the L_1-norm, $\|\bar{d}_\ell - \tilde{d}_\ell\|_1$, leads to a sparse wavelet transform

reconstruction errors correspond to the detail coefficients in Eq. (1) and our goal is to make this expansion sparse.

The optimization problem of Eq. 2 suffers from a trivial solution: by choosing update matrix to have large norm (e.g. a large coefficient times identity matrix), and predict operator equal to the inverse of update, one can practically cancel the contribution of the bias terms (namely, $\bar{a}_\ell^n$), obtaining almost perfect reconstruction. Trivial solutions are a well-known problem in the context of auto-encoders, and an effective solution is to tie the weights of the encode and decode steps by setting $U = P^\mathsf{T}$. This also has the benefit of decreasing the number of parameters to learn. We too follow a similar strategy and tie the weights of update and predict steps, but the specific form of tying is dictated by the wavelet properties and will be discussed in Sect. 4.2.

The training is conducted in a manner similar to the greedy pre-training of a stack of auto-encoders [11–14]. Namely, we first train the the update and predict operators at the finest level: here the input to the lifting step are the original training functions—this corresponds to $\ell = \ell_{max} - 1$ and $\forall n, a_{\ell+1}^n = f^n$ in Fig. 1 (left). After training of this finest level is completed, we obtain new approximation coefficients a_ℓ^n which are passed to the next level as the training functions, and this process is repeated until one reaches the coarsest level.

The use of tied auto-encoders is motivated by their success in deep learning revealing their capability to learn useful features from the data under a variety of circumstances. The choice of the lifting scheme as the backbone of our construction is motivated by several observations. First, every invertible 1D discrete wavelet transform can be factored into lifting steps [20], which makes lifting a universal tool for constructing multiresolutions. Second, lifting scheme is always invertible, and provides exact reconstruction of signals. Third, it affords fast (linear time) and memory efficient (in-place) implementation after the update and predict operators are specified. We choose to apply lifting to Haar wavelets specifically because Haar wavelets are easy to define on any underlying space provided that it can be hierarchi-

cally partitioned [3, 4]. Our use of update-first scheme mirrors its common use for adaptive wavelet constructions in image processing literature, which is motivated by its stability; see [21] for a thorough discussion.

4 Construction Details

We consider a simple connected weighted graph G with vertex set V of size N. A signal on the graph is represented by a vector $f \in \mathbb{R}^N$. Let W be the $N \times N$ edge weight matrix (since there are no self-loops, $W_{ii} = 0$), and let S be the diagonal $N \times N$ matrix of vertex weights; if no vertex weights are given, we set $S_{ii} = \sum_j W_{ij}$. For a graph signal f, we define its integral over the graph as a weighted sum,

$$\int_G f = \sum_i S_{ii} f(i).$$

We define the volume of a subset R of vertices of the graph by

$$Vol(R) = \int_R 1 = \sum_{i \in R} S_{ii}.$$

We assume that a hierarchical partitioning (not necessarily dyadic) of the underlying graph into connected regions is provided. We denote the regions at level $\ell = 1, \ldots, \ell_{max}$ by $R_{\ell,k}$; see Fig. 3 where the three coarsest partition levels of a dataset are shown. For each region at levels $\ell = 1, \ldots, \ell_{max} - 1$, we designate arbitrarily all except one of its children (i.e. regions at level $\ell + 1$) as active regions. As will become clear, our wavelet construction yields one approximation coefficient $a_{\ell,k}$ for each region $R_{\ell,k}$, and one detail coefficient $d_{\ell,k}$ for each *active* region $R_{\ell+1,k}$ at level $\ell + 1$. Note that if the partition is not dyadic, at a given level ℓ the number of scaling coefficients (equal to number of regions at level ℓ) will not be the same as the number of detail coefficients (equal to number of active regions at level $\ell + 1$).

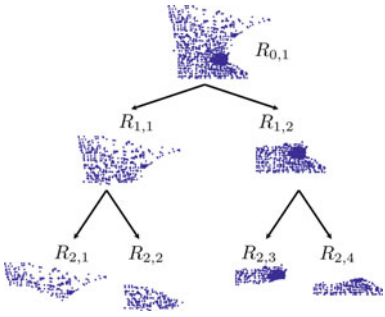


Fig. 3 Hierarchical partitioning into regions of the graph representing the road network of Minnesota. Depicted are the three coarsest levels of the partitioning

We collect all of the coefficients at the same level into vectors denoted by a_ℓ and d_ℓ; to keep our notation lightweight, we refrain from using boldface for vectors.

4.1 Haar Wavelets

Usually, the (unnormalized) Haar approximation and detail coefficients of a signal f are computed as follows. The coefficient $\bar{a}_{\ell,k}$ corresponding to region $R_{\ell,k}$ equals to the average of the function f on that region:

$$\bar{a}_{\ell,k} = Vol(R_{\ell,k})^{-1} \int_{R_{\ell,k}} f.$$

The detail coefficient $\bar{d}_{\ell,k}$ corresponding to an *active* region $R_{\ell+1,k}$ is the difference between averages at the region $R_{\ell+1,k}$ and its parent region $R_{\ell,\mathrm{par}(k)}$, namely $\bar{d}_{\ell,k} = \bar{a}_{\ell+1,k} - \bar{a}_{\ell,\mathrm{par}(k)}$. For perfect reconstruction there is no need to keep detail coefficients for inactive regions, because these can be recovered from the scaling coefficient of the parent region and the detail coefficients of the sibling regions.

In our setting, Haar wavelets are a part of the lifting scheme, and so the coefficient vectors $\bar{a}_\ell$ and $\bar{d}_\ell$ at level ℓ need to be computed from the augmented coefficient vector $a_{\ell+1}$ at level $\ell+1$ (c.f. Fig. 1 (left)). This is equivalent to computing a function's average at a given region from its averages at the children regions. As a result, we obtain the following formula:

$$\bar{a}_{\ell,k} = Vol(R_{\ell,k})^{-1} \sum_{j,\mathrm{par}(j)=k} a_{\ell+1,j} Vol(R_{\ell+1,j}), \tag{3}$$

where the summation is over all the children regions of $R_{\ell,k}$. As before, the detail coefficient corresponding to an active region $R_{\ell+1,k}$ is given by $\bar{d}_{\ell,k} = a_{\ell+1,k} - \bar{a}_{\ell,\mathrm{par}(k)}$. The resulting Haar wavelets are *not normalized*; when comparing or sorting wavelet/scaling coefficients coming from different levels we need to multiply coefficients coming from level ℓ by $2^{-\ell/2}$.

4.2 Auto-Encoder Setup

The choice of the update and predict operators and their tying scheme is guided by a number of properties that wavelets need to satisfy. We discuss these requirements under separate headings.

Vanishing moments The wavelets should have vanishing dual and primal moments—two independent conditions due to biorthogonality of our wavelets. In terms of the approximation and detail coefficients these can be expressed as follows: a) all of the

detail coefficients of a constant function should be zero and b) the integral of the approximation at any level of multiresolution should be the same as the integral of the original function.

Since these conditions are already satisfied by the Haar wavelets, we need to ensure that the update and predict operators preserve them. To be more precise, if $a_{\ell+1}$ is a constant vector, then we have for Haar coefficients that $\bar{a}_\ell = c\mathbf{1}$ and $\bar{d}_\ell = \mathbf{0}$; here c is some constant and $\mathbf{1}$ is a column-vector of all ones. To satisfy a) after lifting, we need to ensure that $d_\ell = \bar{d}_\ell - P(\bar{a}_\ell + U\bar{d}_\ell) = -P\bar{a}_\ell = -cP\mathbf{1} = \mathbf{0}$. Therefore, the rows of predict operator should sum to zero: $P\mathbf{1} = \mathbf{0}$.

To satisfy (b), we need to preserve the first order moment at every level ℓ by requiring

$$\sum_k a_{\ell+1,k} Vol(R_{\ell+1,k}) = \sum_k \bar{a}_{\ell,k} Vol(R_{\ell,k}) = \sum_k a_{\ell,k} Vol(R_{\ell,k}).$$

The first equality is already satisfied (due to the use of Haar wavelets), so we need to constrain our update operator. Introducing the diagonal matrix A_c of the region volumes at level ℓ, we can write

$$0 = \sum_k a_{\ell,k} Vol(R_{\ell,k}) - \sum_k \bar{a}_{\ell,k} Vol(R_{\ell,k}) = \sum_k U\bar{d}_\ell Vol(R_{\ell,k}) = \mathbf{1}^\mathsf{T} A_c U \bar{d}_\ell.$$

Since this should be satisfied for all $\bar{d}_\ell$, we must have $\mathbf{1}^\mathsf{T} A_c U = \mathbf{0}^\mathsf{T}$.

Taking these two requirements into consideration, we impose the following constraints on predict and update weights:

$$P\mathbf{1} = \mathbf{0} \quad \text{and} \quad U = A_c^{-1} P^\mathsf{T} A_f$$

where A_f is the diagonal matrix of the active region volumes at level $\ell + 1$. It is easy to check that

$$\mathbf{1}^\mathsf{T} A_c U = \mathbf{1}^\mathsf{T} A_c A_c^{-1} P^\mathsf{T} A_f = \mathbf{1}^\mathsf{T} P^\mathsf{T} A_f = (P\mathbf{1})^\mathsf{T} A_f = \mathbf{0}^\mathsf{T} A_f = \mathbf{0}^\mathsf{T},$$

as required. We have introduced the volume matrix A_f of regions at the finer level to make the update/predict matrices dimensionless (i.e. insensitive to whether the volume is measured in any particular units); also note the resemblance to Eq. (3).

Locality To make our wavelets and scaling functions localized on the graph, we need to constrain update and predict operators in a way that would disallow distant regions from updating or predicting the approximation/detail coefficients of each other.

Since the update is tied to the predict operator, we can limit ourselves to the latter operator. For a detail coefficient $d_{\ell,k}$ corresponding to the active region $R_{\ell+1,k}$, we only allow predictions that come from the parent region $R_{\ell,\mathrm{par}(k)}$ and the immediate neighbors of this parent region. Two regions of graph are considered neighboring if

their union is a connected graph. This can be seen as enforcing a sparsity structure on the matrix P or as limiting the interconnections between the layers of neurons.

As a result of this choice, it is not difficult to see that the resulting scaling functions $\phi_{\ell,k}$ and wavelets $\psi_{\ell,k}$ will be supported in the vicinity of the region $R_{\ell,k}$. If desired, larger supports can be obtained by allowing the use of second and higher order neighbors of the parent for prediction.

4.3 Optimization

In our setting, due to the relatively small size of the training set and sparse interconnectivity between the layers, an off-the-shelf L-BFGS[1] unconstrained smooth optimization package works very well. In order to make our problem unconstrained, we avoid imposing the equation $P\mathbf{1} = \mathbf{0}$ as a hard constraint. Instead, in each row of P (which corresponds to some active region), the weight corresponding to the parent is eliminated by expressing it as the negative sum of the remaining entries in the row. To obtain a smooth objective, we use L_1-norm with soft absolute value

$$s(x) = \|x\|_1 = \sum_i |x_i| \approx \sum_i \sqrt{\epsilon + x_i^2},$$

where we set $\varepsilon = 10^{-4}$. *The initialization is done by setting all of the weights equal to zero*. This is meaningful, because it corresponds to no lifting at all, and would reproduce the original Haar wavelets.

There has been immense progress in the area of neural network optimization since the original publication of this work. Various versions of stochastic gradient descent have been proposed, ADAM [22] being one of the most popular. We have re-implemented the algorithm in TensorFlow [23] using the ADAM optimizer and found that the results were nearly identical to the ones obtained via L-BFGS. Thus, all of the results presented in this chapter are based on our original optimization procedure.

The automatic differentiation feature of the modern deep learning packages makes it straightforward to train the overall architecture—the process commonly called fine tuning—after the layer-wise training is completed. This optimization would require an aggregate loss over all the partitioning levels. Note that some weighing would be necessary when aggregating L_1-norms coming from different levels. At the minumum, a weight of $2^{-\ell/2}$ for level ℓ contribution would be required, see Sect. 3; we leave this for a future work.

[1] Mark Schmidt, http://www.di.ens.fr/~mschmidt/Software/minFunc.html.

4.4 Training Functions

When training functions are available we directly use them. However, our construction can be applied even if training functions are not specified. In this case we choose smoothness as our prior, and train the wavelets with a set of smooth functions on the graph—namely, we use scaled eigenvectors of graph Laplacian corresponding to the smallest eigenvalues. More precisely, let D be the diagonal matrix with entries $D_{ii} = \sum_j W_{ij}$. The graph Laplacian L is defined as $L = S^{-1}(D - W)$. We solve the symmetric generalized eigenvalue problem $(D - W)\xi = \lambda S\xi$ to compute the smallest eigen-pairs $\{\lambda_n, \xi_n\}_{n=0}^{n_{max}}$.We discard the 0-th eigen-pair which corresponds to the constant eigenvector, and use functions $\{\xi_n/\lambda_n\}_{n=1}^{n_{max}}$ as our training set. The inverse scaling by the eigenvalue is included because eigenvectors corresponding to larger eigenvalues are less smooth (cf. [24]), and so should be assigned smaller weights to achieve a smooth prior.

The number of training functions required to robustly train the neural networks depends on the number of parameters; in our case this is related to the number of the neighbors that a region can have at a given level. In the cases discussed in our experiments, graphs have low-dimensional structure, and the number of neighboring partitions is low—which allows the training to succeed with a small number of training functions. For high-dimensional point clouds a larger number (growing with the intrinsic dimension of the manifold) of training functions will be required.

4.5 Partitioning

Since our construction is based on improving upon the Haar wavelets, their quality will have an effect on the final wavelets. As proved in [4], the quality of Haar wavelets depends on the quality (balance) of the graph partitioning. From practical standpoint, it is hard to achieve high quality partitions on all types of graphs using a single algorithm. However, for the datasets presented in this paper we find that the following approach based on spectral clustering algorithm of [25] works well. Namely, we first embed the graph vertices into $\mathbb{R}^{n_{max}}$ as follows: $i \to (\xi_1(i)/\lambda_1, \xi_2(i)/\lambda_2, \ldots, \xi_{n_{max}}(i)/\lambda_{n_{max}}), \forall i \in V$, where $\{\lambda_n, \xi_n\}_{n=0}^{n_{max}}$ are the eigen-pairs of the Laplacian as in Sect. 4.4, and $\xi_\cdot(i)$ is the value of the eigenvector at the i-th vertex of the graph. To obtain a hierarchical tree of partitions, we start with the graph itself as the root. At every step, a given region (a subset of the vertex set) of graph G is split into two children partitions by running the 2-means clustering algorithm (k-means with $k = 2$) on the above embedding restricted to the vertices of the given partition [3]. This process is continued in recursion at every obtained region. This results in a dyadic partitioning except at the finest level ℓ_{max}.

4.6 Graph Construction for Point Clouds

Our problem setup started with a weighted graph and arrived to the Laplacian matrix L in Sect. 4.4. It is also possible o reverse this process whereby one starts with the Laplacian matrix L and infers from it the weighted graph. This is a natural way of dealing with point clouds sampled from low-dimensional manifolds, a setting common in manifold learning. There is a number of ways for computing Laplacians on point clouds, see [26]; almost all of them fit into the above form $L = S^{-1}(D - W)$, and so, they can be used to infer a weighted graph that can be plugged into our construction.

5 Experiments

Our goal is to experimentally investigate the constructed wavelets for multiscale behavior, meaningful adaptation to training signals, and sparse representation that generalizes to testing signals.

For the first two objectives we visualize the scaling functions at different levels ℓ because they provide insight about the signal approximation spaces $\mathbf{V}_\ell$. The generalization performance can be deduced from comparison to Haar wavelets, because during training we modify Haar wavelets so as to achieve a sparser representation of training signals.

We start with the case of a periodic interval, which is discretized as a cycle graph; 32 scaled eigenvectors (sines and cosines) are used for training. Figure 4 shows the resulting scaling and wavelet functions at level $\ell = 4$. Up to discretization errors, the wavelets and scaling functions at the same level are shifts of each other—showing that our construction is able to learn shift invariance from training functions.

Figure 5a depicts a graph representing the road network of Minnesota, with edges showing the major roads and vertices being their intersections. In our construction we employ unit weights on edges and use 32 scaled eigenvectors of graph Laplacian

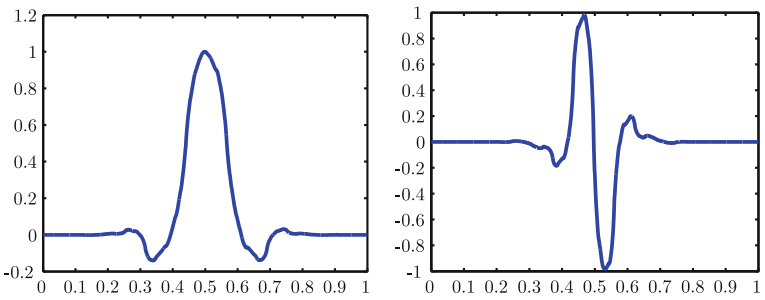
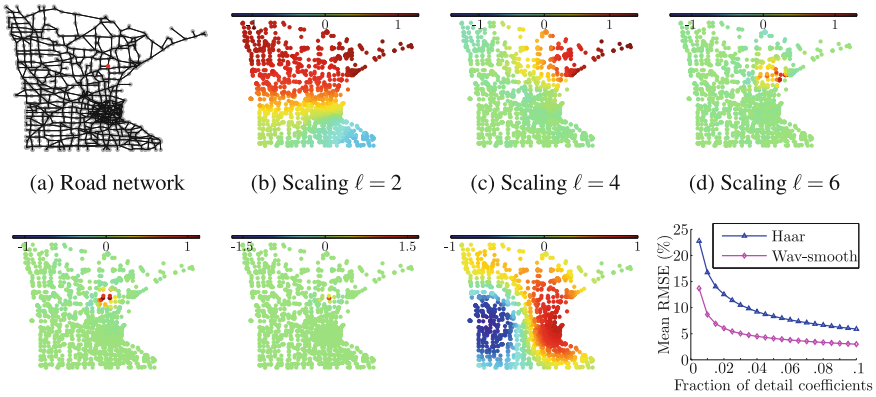


Fig. 4 Scaling and wavelet functions on the periodic interval at level $\ell = 4$



(a) Road network (b) Scaling $\ell = 2$ (c) Scaling $\ell = 4$ (d) Scaling $\ell = 6$

(e) Scaling $\ell = 8$ (f) Scaling $\ell = 10$ (g) Sample function (h) Reconstruction error

Fig. 5 Our construction trained with smooth prior on the network (**a**), yields the scaling functions (**b–f**). A sample continuous function (**g**) out of 100 total test functions. Better average reconstruction results (**h**) for our wavelets (Wav-smooth) indicate a good generalization performance

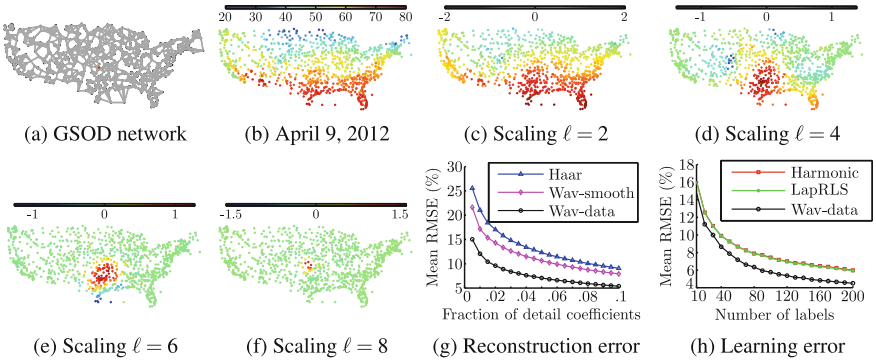
as training functions. The resulting scaling functions for regions containing the red vertex in Fig. 5a are shown at different levels in Fig. 5b–f. The function values at graph vertices are color coded from smallest (dark blue) to largest (dark red). Note that the scaling functions are continuous and show multiscale spatial behavior.

To test whether the learned wavelets provide a sparse representation of smooth signals, we synthetically generated 100 continuous functions using the xy-coordinates (the coordinates have not been seen by the algorithm so far) of the vertices; Fig. 5g shows one of such functions. Figure 5h shows the average error of reconstruction from expansion Eq. (1) with $\ell_0 = 1$ by keeping a specified fraction of largest detail coefficients. The improvement over the Haar wavelets shows that our model generalizes well to unseen signals.

Next, we apply our approach to real-world graph signals. We use a dataset of average daily temperature measurements[2] from meteorological stations located on the mainland US. The longitudes and latitudes of stations are treated as coordinates of a point cloud, from which a weighted Laplacian is constructed using [26] with 5-nearest neighbors; the resulting graph is shown in Fig. 6(a).

The daily temperature data for the year of 2012 gives us 366 signals on the graph; Fig. 6b depicts one such signal. We use the signals from the first half of the year to train the wavelets, and test for sparse reconstruction quality on the second half of the year (and vice versa). Figure 6c–g depicts some of the scaling functions at a number of levels; note that the depicted scaling function at level $\ell = 2$ captures the rough temperature distribution pattern of the US. The average reconstruction error from a specified fraction of largest detail coefficients is shown in Fig. 6g.

[2]National Climatic Data Center, http://www.ftp://ftp.ncdc.noaa.gov/pub/data/gsod/2012/.



(a) GSOD network (b) April 9, 2012 (c) Scaling $\ell = 2$ (d) Scaling $\ell = 4$

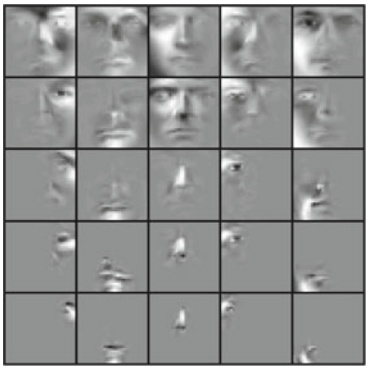
(e) Scaling $\ell = 6$ (f) Scaling $\ell = 8$ (g) Reconstruction error (h) Learning error

Fig. 6 Our construction on the station network (**a**) trained with daily temperature data (e.g. (**b**)), yields the scaling functions (**c**–**f**). Reconstruction results (**g**) using our wavelets trained on data (Wav-data) and with smooth prior (Wav-smooth). Results of semi-supervised learning (**h**)

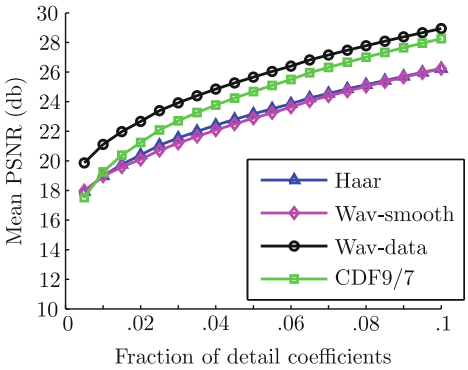
As an application, we employ our wavelets for semi-supervised learning of the temperature distribution for a day from the temperatures at a subset of labeled graph vertices. The sought temperature distribution is expanded as in Eq. (1) with $\ell_0 = 1$, and the coefficients are found by solving a least squares problem using temperature values at labeled vertices. Since we expect the detail coefficients to be sparse, we impose a Lasso penalty on them (i.e. the regularizer $\lambda \sum_{\ell,k} |d_{\ell,k}|$ is added to the least squares objective); to make the problem smaller, all detail coefficients for levels $\ell \geq 7$ are set to zero. We compare to the Laplacian regularized least squares [24] and harmonic interpolation approach [27]. A hold-out set of 25 random vertices is used to assign all the regularization parameters. The experiment is repeated for each of the days (not used to learn the wavelets) with the number of labeled vertices ranging from 10 to 200. Figure 6h shows the errors averaged over all days; our approach achieves lower error rates than the competitors.

Our final example serves two purposes—showing the benefits of our construction in a standard image processing application and better demonstrating the nature of learned scaling functions. Images can be seen as signals on a graph—pixels are the vertices and each pixel is connected to its 8 nearest neighbors. We consider all of the Extended Yale Face Database B [28] images (cropped and down-sampled to 32×32) as a collection of signals on a single underlying graph. We randomly split the collection into half for training our wavelets, and test their reconstruction quality on the remaining half. Figure 7a depicts a number of obtained scaling functions at different levels (the rows correspond to levels $\ell = 4, 5, 6, 7, 8$) in various locations (columns). The scaling functions have a face-like appearance at coarser levels, and capture more detailed facial features at finer levels. Note that the scaling functions show controllable multiscale spatial behavior.

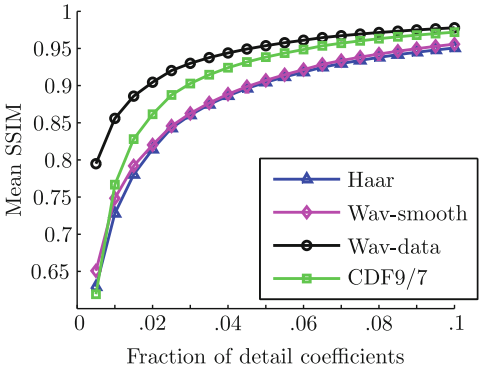
The quality of reconstruction from a sparse set of detail coefficients is plotted in Fig. 7b, c. Here again we consider the expansion of Eq. (1) with $\ell_0 = 1$, and reconstruct using a specified proportion of largest detail coefficients. We also make



(a) Scaling functions



(b) PSNR



(c) SSIM

Fig. 7 The scaling functions (**a**) resulting from training on a face images dataset. These wavelets (Wav-data) provide better sparse reconstruction quality than the CDF9/7 wavelet filterbanks (**b**, **c**)

a comparison to reconstruction using the standard separable CDF 9/7 wavelet filterbanks from bottom-most level; for both of quality metrics, our wavelets trained on data perform better than CDF 9/7. The smoothly trained wavelets do not improve over the Haar wavelets, because the smoothness assumption does not hold for face images.

6 Conclusion

We have introduced an approach to constructing wavelets that take into consideration structural properties of both graph signals and their underlying graphs. An interesting direction for future research would be to randomize the graph partitioning process or

to use bagging over training functions in order to obtain a family of wavelet constructions on the same graph—leading to over-complete dictionaries like in [29]. One can also introduce multiple lifting steps at each level or even add non-linearities as common with neural networks. Our wavelets are obtained by training a structure similar to a deep neural network; interestingly, the recent work of Mallat and collaborators (e.g. [30]) goes in the other direction and provides a wavelet interpretation of deep neural networks. Therefore, we believe that there are ample opportunities for future work in the interface between wavelets and deep neural networks.

Acknowledgements We thank Jonathan Huang for discussions and especially for his advice regarding the experimental section. The authors acknowledge the support of NSF grants FODAVA 808515 and DMS 1228304, AFOSR grant FA9550-12-1-0372, ONR grant N00014-13-1-0341, a Google research award, and the Max Plack Center for Visual Computing and Communications.

References

1. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
2. M. Crovella, E.D. Kolaczyk, Graph wavelets for spatial traffic analysis, in *INFOCOM* (2003)
3. A.D. Szlam, M. Maggioni, R.R. Coifman, J.C. Bremer, Diffusion-driven multiscale analysis on manifolds and graphs: top-down and bottom-up constructions, in *SPIE* (2005), p. 5914
4. M. Gavish, B. Nadler, R.R. Coifman, Multiscale wavelets on trees, graphs and high dimensional data: theory and applications to semi supervised learning, in *ICML* (2010), pp. 367–374
5. R.R. Coifman, M. Maggioni, Diffusion wavelets. Appl. Comput. Harmon. Anal. **21**(1), 53–94 (2006). https://doi.org/10.1016/j.acha.2006.04.004
6. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011). https://doi.org/10.1016/j.acha.2010.04.005
7. I. Ram, M. Elad, I. Cohen, Generalized tree-based wavelet transform. IEEE Trans. Signal Process. **59**(9), 4199–4209 (2011)
8. R.M. Rustamov, Average interpolating wavelets on point clouds and graphs. CoRR (2011). arXiv:1110.2227
9. S.K. Narang, A. Ortega, Multi-dimensional separable critically sampled wavelet filterbanks on arbitrary graphs, in *ICASSP* (2012), pp. 3501–3504
10. M.N. Do, Y.M. Lu, Multidimensional filter banks and multiscale geometric representations. Found. Trends Signal Process. **5**(3), 157–264 (2012)
11. G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006). https://doi.org/10.1162/neco.2006.18.7.1527
12. G.E. Hinton, R. Salakhutdinov, Reducing the dimensionality of data with neural networks. Science **313**, 504–507 (2006). https://doi.org/10.1126/science.1127647
13. Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks, in *Adv. Neural Inf. Process. Syst. 19*, ed. by B. Schölkopf, J. Platt, T. Hoffman (MIT Press, MA, 2007), pp. 153–160
14. M. Ranzato, C. Poultney, S. Chopra, Y. LeCun, Efficient learning of sparse representations with an energy-based model, in *Adv. Neural Inf. Process. Syst. 19*, ed. by B. Schölkopf, J. Platt, T. Hoffman (MIT Press, MA, 2007), pp. 1137–1144
15. R. Rustamov, L.J. Guibas, Wavelets on graphs via deep learning, in *Advances in Neural Information Processing Systems 26*, ed. by C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani,

K.Q. Weinberger (Curran Associates, Inc., 2013), pp. 998–1006. http://papers.nips.cc/paper/5046-wavelets-on-graphs-via-deep-learning.pdf
16. F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model. Trans. Neural Netw. **20**(1), 61–80 (2009). https://doi.org/10.1109/TNN.2008.2005605
17. M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst, Geometric deep learning: going beyond euclidean data. IEEE Signal Process. Mag. **34**(4), 18–42 (2017). https://doi.org/10.1109/MSP.2017.2693418
18. S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*, 3rd edn. (Academic Press, 2008)
19. W. Sweldens, The lifting scheme: a construction of second generation wavelets. SIAM J. Math. Anal. **29**(2), 511–546 (1998)
20. I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps. J. Fourier Anal. Appl. **4**(3), 245–267 (1998)
21. R.L. Claypoole, G. Davis, W. Sweldens, R.G. Baraniuk, Nonlinear wavelet transforms for image coding via lifting. IEEE Trans. Image Process. **12**(12), 1449–1459 (2003)
22. D.P. Kingma, J. Ba, Adam: A method for stochastic optimization (2015)
23. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems (2015). https://www.tensorflow.org/. Software available from tensorflow.org
24. M. Belkin, P. Niyogi, Semi-supervised learning on riemannian manifolds. Mach. Learn. **56**(1–3), 209–239 (2004)
25. A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in *NIPS* (2001), pp. 849–856
26. R.R. Coifman, S. Lafon, Diffusion maps. Applied and Computational Harmonic Analysis **21**(1), 5–30 (2006). https://doi.org/10.1016/j.acha.2006.04.006
27. X. Zhu, Z. Ghahramani, J.D. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in *ICML* (2003), pp. 912–919
28. A. Georghiades, P. Belhumeur, D. Kriegman, From few to many: illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell. **23**(6), 643–660 (2001)
29. X. Zhang, X. Dong, P. Frossard, Learning of structured graph dictionaries, in *ICASSP* (2012), pp. 3373–3376. https://doi.org/10.1109/ICASSP.2012.6288639
30. J. Bruna, S. Mallat, Invariant scattering convolution networks. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1872–1886 (2013)

Oversampled Transforms for Graph Signals



Yuichi Tanaka and Akie Sakiyama

Abstract In this chapter, oversampled transforms for graph signals are introduced. Oversampling is done in two ways: One is oversampled graph Laplacian and the other is oversampled graph transforms. Both are described here. The advantage of the oversampled transforms is that we can take a good trade-off between performance (in context to sparsifying the graph signals) and storage/memory space for transformed coefficients. Furthermore, any graph can be converted into an oversampled bipartite graph by using the oversampled graph Laplacian. It leads to that well-known graph wavelet transforms/filter banks for bipartite graphs can be applied to the signals on any graphs with a slight sacrifice of redundancy. Actual performances are compared through several numerical experiments.

1 Introduction

Transforms and dictionaries for graph signals can be classified by several criteria. In this chapter, we look at them from *redundancy*.

Redundancy represents the ratio between the number of samples in the original domain and that in the transformed domain. Transforms/dictionaries based on redundancy are classified as follows, where N_o and N_t are defined as the number of samples in the original and transformed domains, respectively:

- *Undecimated*: $N_t \gg N_o$. There is no sampling. Redundancy is identical to the number of filters, i.e., the number of rows for the forward transform matrix divided by N_o.
- *Oversampled*: $N_t > N_o$. Sampling ratio is less than the number of filters.
- *Critically-sampled*: $N_t = N_o$. Both of the analysis and synthesis transforms are represented as square matrices.

Y. Tanaka (✉) · A. Sakiyama
Tokyo University of Agriculture and Technology, Fuchu, Japan
e-mail: ytnk@cc.tuat.ac.jp

A. Sakiyama
e-mail: sakiyama@msp-lab.org

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_6

- *Undersampled*: $N_t < N_o$. Sampling ratio is greater than the number of filters. Perfect reconstruction is not possible in general.

In this chapter, we focus on the oversampled transforms [16, 22].

There are numerous methods for undecimated graph transforms (see [8, 17, 20] and references therein) since they are relatively easy to guarantee the perfect reconstruction. However, graph signals are often high-dimensional, and therefore, the undecimated transforms requires huge storage space and/or computation complexity. In contrast, critically-sampled and undersampled transforms need less storage and complexity, with a trade-off for the restriction of the freedom for filter design. So, we take a (relatively good) tradeoff: Oversampled transforms.

Oversampling of graph signals is an interesting topic and it is quite different from that of classical signal processing. This is because we have a *discrete signal* as well as its *underlying graph*; We need to consider the *oversampled graphs*.

In this chapter, an effective graph oversampling method and oversampled graph filter banks are studied. In particular, we describe a method that converts an arbitrary K-colorable graph into one bipartite graph containing all edges of the original graph. This enables us to apply graph wavelets and filter banks for signals on any graphs without graph simplification. Furthermore, the oversampled spectral graph filter banks are introduced that structurally guarantee the perfect reconstruction. The performances of the oversampled transforms are compared through experiments on image processing and denoising of graph signals.

The rest of this chapter is organized as follows. In Sect. 2, an overview of oversampled transforms on graphs are shown. The oversampled graph Laplacian is presented in Sect. 3, along with the theoretical connection with graph theory. In Sect. 4, some examples of graph oversampling are shown. Sect. 5 presents the design method of the oversampled graph filter banks. Experimental results are shown in Sect. 6. Finally, this chapter concludes in Sect. 7.

Notation

In this chapter, signal processing on undirected graphs is only considered. A graph $G = (V, E)$ has a vertex set $V = \{v_0, v_1, \ldots, v_{N-1}\}$ and edge set E, where the number of vertices N is interchangeably used with the cardinality of V, i.e., $N = |V|$. The special notation for a bipartite graph is $G = (L, H, E)$, where L and H are disjoint vertex sets. That means there are no edges within L and H, and E only connects vertices in L and those in H.

$\mathbf{A} \in \mathbb{R}^{N \times N}$ is an adjacency matrix where $[\mathbf{A}]_{ij}$ represents the edge weight between v_i and v_j. The diagonal degree matrix is represented as $[\mathbf{D}]_{ii} = \sum_j [\mathbf{A}]_{ij}$. The combinatorial graph Laplacian is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Since $\mathbf{L}$ is a real symmetric matrix, its eigendecomposition is represented as $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{U} = \left[\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{N-1}\right]$ is the eigenvector matrix and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_0, \lambda_1, \ldots, \lambda_{N-1})$ is the diagonal eigenvalue matrix, i.e., $\mathbf{L}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i$.

The symmetric normalized graph Laplacian is also defined as $\underline{\mathbf{L}} := \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$. $\underline{\mathbf{L}}$ has its eigenvalues within the range $\lambda \in [0, 2]$, and $\lambda_{\max} = 2$ only for bipartite graphs.

The graph signal is $f : V \to \mathbb{R}$ and it can be represented as the vector $\boldsymbol{f} \in \mathbb{R}^N$, where the nth sample $f(n)$ is assumed to be located on the nth vertex of G. The graph Fourier transform (GFT) is defined as the inner product of $\boldsymbol{u}_i$ and $\boldsymbol{f}$:

$$\widetilde{f}(i) = \langle \boldsymbol{u}_i, \boldsymbol{f} \rangle = \sum_{k=0}^{N-1} u_i(k) f(k). \tag{1}$$

2 Oversampling of Graphs and Graph Signals

We consider the following two cases to realize oversampled transforms for graph signals.

Case 1 Oversampling is performed before transformation.
Case 2 Integrated oversampling is performed within transformation.

In Case 1, the input signal is oversampled along with the expansion of the original graph. It leads to that we have to consider the expansion strategy of the graph, that is specific for graph signal processing. In Case 2, we need to design oversampled filter banks or dictionaries for graph signals. Indeed Case 1 and Case 2 can be cascaded. This is illustrated in Fig. 1 and its corresponding synthesis transform is shown in Fig. 2.

2.1 Case 1

For Case 1, let us define the adjacency matrix of the original graph $G_0 = (V_0, E_0)$ as $\mathbf{A}_0$ and its corresponding graph Laplacian $\mathbf{L}_0$, where $|V_0| = N_0$. Formally, the oversampled graph Laplacian $\mathbf{L}_{\mathrm{OS}}$ is represented as follows.

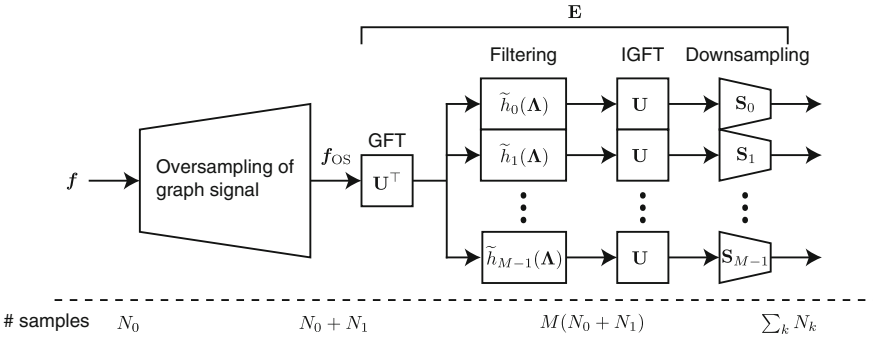


Fig. 1 Framework of oversampled graph transform (analysis side)

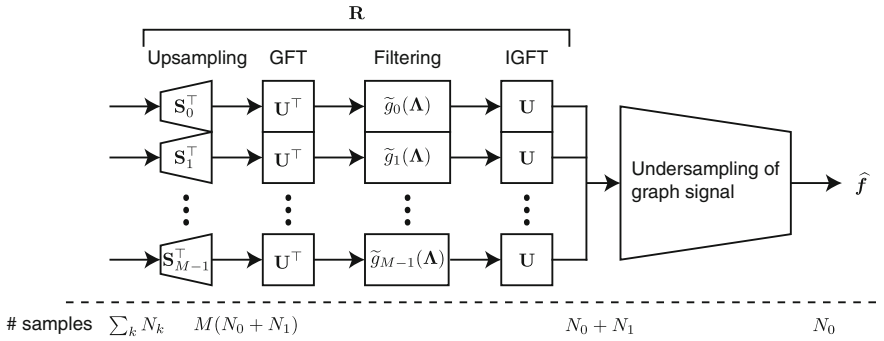


Fig. 2 Framework of oversampled graph transform (synthesis side)

$$\mathbf{L}_{\mathrm{OS}} = \mathbf{D}_{\mathrm{OS}} - \mathbf{A}_{\mathrm{OS}}, \tag{2}$$

where

$$\begin{aligned} \mathbf{D}_{\mathrm{OS}} &= \begin{bmatrix} \mathbf{D}_0 & \\ & \mathbf{D}_1 \end{bmatrix} \\ \mathbf{A}_{\mathrm{OS}} &= \begin{bmatrix} \mathbf{A}_0 & \mathbf{A}_{01} \\ \mathbf{A}_{01}^\top & \mathbf{A}_1 \end{bmatrix}, \end{aligned} \tag{3}$$

in which $\mathbf{A}_1 \in \mathbb{R}^{N_1 \times N_1}$ represents edges within the appended vertices and $\mathbf{A}_{01} \in \mathbb{R}^{N_0 \times N_1}$ represents those between the original and appended vertices. $\mathbf{A}_1$ and $\mathbf{A}_{01}$ can be arbitrarily chosen according to requirements and/or applications.

The input signal is also oversampled in Case 1; Let $\boldsymbol{f}_0 \in \mathbb{R}^{N_0}$ be the original signal. The oversampled signal is represented as

$$\boldsymbol{f}_{\mathrm{OS}} = \begin{bmatrix} \boldsymbol{f}_0^\top & \boldsymbol{f}_1^\top \end{bmatrix}^\top, \tag{4}$$

where $\boldsymbol{f}_1 \in \mathbb{R}^{N_1}$ is an arbitrary oversampled signal corresponding to $\mathbf{A}_1$. It would be an all-zero signal, a part of $\boldsymbol{f}_0$, or interpolated from $\boldsymbol{f}_0$.

$\mathbf{A}_{\mathrm{OS}}$ can actually be designed somewhat arbitrarily, however, the guideline for oversampling is needed for actual graph signal processing systems. In Sect. 3, a design method of $\mathbf{A}_{\mathrm{OS}}$ is introduced that is able to convert any K-colorable graph to an oversampled bipartite graph. It is beneficial for a typical class of graph wavelet transforms that is perfect reconstruction only for the signal on bipartite graphs.

2.2 Case 2

For Case 2, the graph Laplacian used is either $\mathbf{L}_0$ or $\mathbf{L}_{\mathrm{OS}}$. For simplicity, we denote either of them as $\mathbf{L}$. The oversampled transform is represented as

$$\mathbf{E} = \begin{bmatrix} \mathbf{E}_0 \\ \mathbf{E}_1 \\ \vdots \\ \mathbf{E}_{M-1} \end{bmatrix} \in \mathbb{R}^{(\sum_k N_k) \times N} \tag{5}$$

where

$$\mathbf{E}_k = \mathbf{S}_k \mathbf{U} \widetilde{h}_k(\mathbf{\Lambda}) \mathbf{U}^\top \tag{6}$$

in which $\mathbf{S}_k \in \{0, 1\}^{N_k \times N}$ is the sampling matrix, i.e., the submatrix of the identity matrix. Its rows correspond to the indices of the remaining vertices after sampling. $\widetilde{h}_k(\mathbf{\Lambda}) = \mathrm{diag}(\widetilde{h}_k(\lambda_0), \widetilde{h}_k(\lambda_1), \ldots, \widetilde{h}_k(\lambda_{N-1}))$ is the spectral response for the kth spectral graph filter. Furthermore, $\sum_k N_k > N$ for oversampled transforms. By removing $\mathbf{S}_k$, the undecimated transform can be obtained whose redundancy is M, i.e., the number of the transformed coefficients is MN. In the oversampled system, the redundancy is usually lower than M by performing the sampling.

Since $\mathbf{E}$ is a tall matrix, the transformed coefficients can be perfectly recovered if $\mathrm{rank}(\mathbf{E}) \geq N$. However, directly calculating $\mathbf{E}^{-1}$ takes huge computation cost since $\mathbf{L}$ is sometimes a large matrix. Therefore, we often need the symmetric structure, that is, the synthesis transform $\mathbf{R}$ is represented as

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_0 \\ \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_{M-1} \end{bmatrix} \in \mathbb{R}^{N \times (\sum_k N_k)} \tag{7}$$

$$\mathbf{R}_k = \mathbf{U} \widetilde{g}_k(\mathbf{\Lambda}) \mathbf{U}^\top \mathbf{S}_k^\top, \tag{8}$$

where $\widetilde{g}_k(\mathbf{\Lambda}) = \mathrm{diag}(\widetilde{g}_k(\lambda_0), \widetilde{g}_k(\lambda_1), \ldots, \widetilde{g}_k(\lambda_{N-1}))$ is the kth *synthesis* filter. $\mathbf{E}$ and $\mathbf{R}$ have to satisfy the following *perfect reconstruction condition*:

$$\mathbf{RE} = \mathbf{I}_N, \tag{9}$$

where $\mathbf{I}_N$ is the identity matrix of size N.

In Sect. 5, the design procedure of $\widetilde{h}_k(\mathbf{\Lambda})$ and $\widetilde{g}_k(\mathbf{\Lambda})$ that satisfies (9) is shown as an extension of the critically-sampled wavelet transforms for graph signals [11, 12].

2.3 Overall Redundancy

Here, we consider the overall redundancy of the oversampled graph transform. The input signal $\boldsymbol{f}_0 \in \mathbb{R}^{N_0}$ is oversampled to $\boldsymbol{f}_{\mathrm{OS}} \in \mathbb{R}^{N_0+N_1}$, then it is transformed by the oversampled transform whose redundancy is $\sum_k N_k$. As a result, the overall redundancy ρ is represented as

$$\rho = \frac{\sum_k N_k}{N_0}. \tag{10}$$

2.4 Special Case: Critically Sampled Transforms

Here we consider a special situation in Case 2: both $\mathbf{E}$ and $\mathbf{R}$ are square matrices that satisfy (9). This is a *critically sampled transform* for graph signals, i.e., $\rho = 1$.

The most popular critically sampled graph transforms, which has filter responses defined in the graph spectral domain, are designed for bipartite graphs [11, 12]. They are perfect reconstruction if the underlying graph is bipartite and the variation operator is a symmetric normalized graph Laplacian or normalized random walk graph Laplacian. Non-bipartite graphs should be simplified to bipartite ones before transformation by this kind of transforms to guarantee the perfect reconstruction condition.

Figure 3 illustrates the entire transformation for one bipartite graph. Let $B = (L, H, E)$ be a bipartite graph only having edges between vertex sets L and H. The number of samples in each channel is determined on the basis of the graph-coloring result. Down- and upsampling for B is defined in matrix notation as follows:

$$\begin{aligned} \mathbf{S}_{d,0} &= \mathbf{I}_L \in \{0,1\}^{|L|\times N}, \quad \mathbf{S}_{u,0} = \mathbf{S}_{d,0}^\top \\ \mathbf{S}_{d,1} &= \mathbf{I}_H \in \{0,1\}^{|H|\times N}, \quad \mathbf{S}_{u,1} = \mathbf{S}_{d,1}^\top, \end{aligned} \tag{11}$$

where $\mathbf{I}_L$ and $\mathbf{I}_H$ are submatrices of $\mathbf{I}_N$ whose rows correspond to the indices of L and H, respectively. That is, the sampled signal can be represented as $\boldsymbol{f}_{d,0} = \mathbf{S}_{d,0}\boldsymbol{f}$ and so on. The two-channel graph filter bank shown in Fig. 3 is designed to satisfy the following perfect reconstruction condition.

$$\mathbf{T}_v = \mathbf{G}_0\mathbf{S}_{u,0}\mathbf{S}_{d,0}\mathbf{H}_0 + \mathbf{G}_1\mathbf{S}_{u,1}\mathbf{S}_{d,1}\mathbf{H}_1 = c^2\mathbf{I}_N, \tag{12}$$

where

$$\begin{aligned} \mathbf{H}_k &:= \mathbf{U}\widetilde{h}_k(\boldsymbol{\Lambda})\mathbf{U}^\top \\ \mathbf{G}_k &:= \mathbf{U}\widetilde{g}_k(\boldsymbol{\Lambda})\mathbf{U}^\top \end{aligned} \tag{13}$$

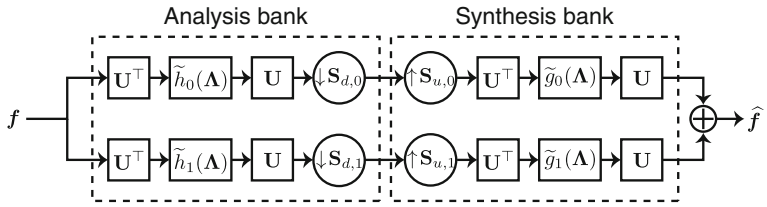


Fig. 3 Critically sampled graph filter bank for signals on bipartite graphs

and $c \in \mathbb{R}$. (12) is further represented as the condition for spectral graph filters as follows.

$$\widetilde{g}_0(\lambda)\widetilde{h}_0(\lambda) + \widetilde{g}_1(\lambda)\widetilde{h}_1(\lambda) = c^2 \tag{14}$$

$$\widetilde{g}_0(\lambda)\widetilde{h}_0(2-\lambda) - \widetilde{g}_1(\lambda)\widetilde{h}_1(2-\lambda) = 0. \tag{15}$$

Several critically sampled graph transforms that satisfy the above perfect reconstruction condition have been proposed, together with some filter designs [11, 12, 16, 17, 22].

- *GraphQMF* [12]: GraphQMF is an orthogonal solution designed from one spectral kernel $\widetilde{h}_0(\lambda)$. The remaining filters are defined as follows.

$$\begin{aligned} \widetilde{h}_1(\lambda) &= \widetilde{h}_0(2-\lambda) \\ \widetilde{g}_0(\lambda) &= \widetilde{h}_0(\lambda) \\ \widetilde{g}_1(\lambda) &= \widetilde{h}_1(\lambda) = \widetilde{h}_0(2-\lambda). \end{aligned} \tag{16}$$

  $\widetilde{h}_0(\lambda)$ has to satisfy the following condition to ensure perfect reconstruction:

$$\widetilde{h}_0^2(\lambda) + \widetilde{h}_0^2(2-\lambda) = c^2. \tag{17}$$

- *GraphBior* [11]: GraphBior, which is a biorthogonal solution, is designed to satisfy

$$\widetilde{h}_1(\lambda) = \widetilde{g}_0(2-\lambda), \;\; \widetilde{g}_1(\lambda) = \widetilde{h}_0(2-\lambda). \tag{18}$$

  This leads to

$$\widetilde{h}_0(\lambda)\widetilde{g}_0(\lambda) + \widetilde{h}_0(2-\lambda)\widetilde{g}_0(2-\lambda) = 2. \tag{19}$$

  A low-pass half-band product filter $\widetilde{p}(\lambda) = \widetilde{h}_0(\lambda)\widetilde{g}_0(\lambda)$ is designed first; then $\widetilde{h}_0(\lambda)$ and $\widetilde{g}_0(\lambda)$ are obtained via spectral factorization similar to the Cohen-Daubechies-Feauveau (CDF) biorthogonal wavelet transform in classical signal processing [5].
- *Frequency Conversion* [17]: A method has been proposed for converting time domain filters $H(\omega)$ into graph spectral filters $H(\lambda)$ through a frequency mapping from $\omega \in [0, \pi]$ to $\lambda \in [0, \lambda_{\max}]$ [17]. In this approach, the perfect reconstruction condition (14) and (15) is always satisfied as long as the set of time domain filters are perfect reconstruction (in the time domain).

3 Oversampled Graph Laplacian

As described in Sect. 2.1, the appended vertices of the oversampled graph Laplacian can be arbitrarily connected to the other vertices. However, an inappropriate choice of graph oversampling causes a performance loss. In this section, we describe an effi-

cient way to construct oversampled graphs that avoids such losses. We also consider an additional constraint; the oversampled graph is bipartite. This is why many existing critically-sampled and oversampled graph transforms are designed for signals on bipartite graphs [11, 12, 16, 17, 22, 25, 26] while there are several exceptions [27, 28].

There have been mainly two methods for applying graph transforms for bipartite graphs into general, i.e., non-bipartite, graphs.

1. Oversampling graphs (this chapter)

   Pros. All edges can be considered.
   Cons. Redundancy becomes (slightly) large.

2. Graph simplification

   Pros. Redundancy can be kept.
   Cons. A part of (and sometimes many) edges should be ignored.

Hereafter, the first approach is introduced. For the second approach, please refer to [13, 21] and references therein.

Since the oversampled graph has to be bipartite, we first construct a *foundation bipartite graph* by removing a part of edges in the original graph. Precisely, $G_f = (L_f, H_f, E_f)$ is made from the original graph, where L_f and H_f are disjoint vertex sets, $L_f \cup H_f = V$, and E_f contains the edges linking E_{L_f} and E_{H_f}. Then, we append vertices and edges in the other bipartite subgraphs to it. In this way, *one oversampled bipartite graph containing all the edges of the bipartite subgraphs* is obtained. While there exist many methods for graph decompositions, any methods can be applied to the following construction.

As an introduction, we start from a simple three-colorable graphs, then it is shown that the similar construction method is also possible for any-colorable graphs.

3.1 Three-Colorable Graphs

First, we describe a way to convert a three-colorable graph into one oversampled bipartite graph containing all edges of the original graph. First of all, three colors are assigned to vertices such that adjacent vertices have different colors. Distinguish these vertices as F_1, F_2 and F_3, respectively.

The three-colorable graph, shown in Fig. 4a, can be decomposed into two bipartite subgraphs: B_1 that contains edges linking $F_{12} := F_1 \cup F_2$ and F_3 (Fig. 4b), and B_2 that contains edges between F_1 and F_2 (Fig. 4c). Hence, the edges in B_2 only have connections on one side of the subsets (F_1 and F_2) of B_1.

To make the oversampled graph, B_1 is viewed as the foundation bipartite graph G_f, i.e., $G_f = B_1 = \{F_{12}, F_3, E_{F_{12} \leftrightarrow F_3}\}$, where $E_{F_k \leftrightarrow F_\ell}$ represents the edges between F_k and F_ℓ, and vertices are appended just *above* each vertex in F_{12} of B_1. The additional vertex sets are represented as F_1' and F_2', respectively. By adding the edges between

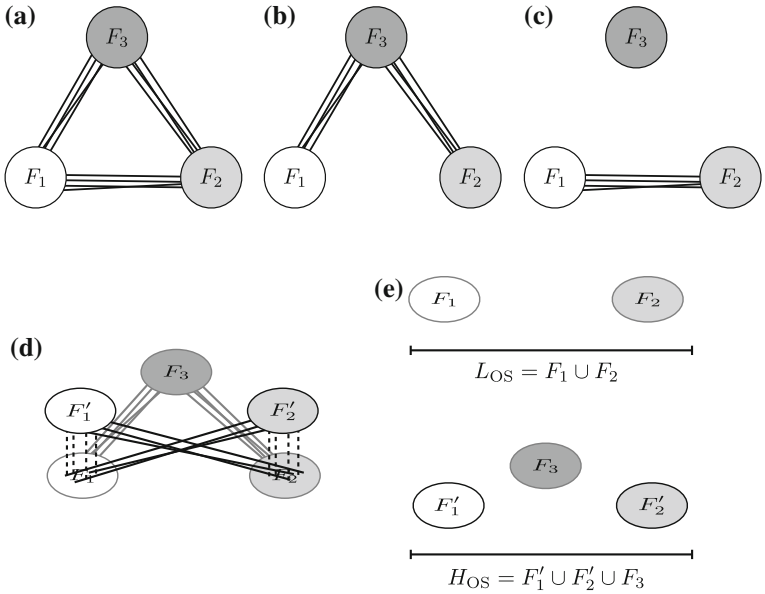


Fig. 4 Bipartite oversampled graph construction for three-colorable graphs. **a** Three-colorable graph whose vertex sets are F_1, F_2 and F_3. **b** Bipartite subgraph B_1. **c** Bipartite subgraph B_2. **d** Oversampled bipartite graph. The gray lines are edges contained in B_1, and the dashed and solid black lines are vertical edges and additional edges according to the edge information of original graph, respectively. **e** Sets L_{OS} and H_{OS} of the oversampled bipartite graph

F_1 and F_2', and also between F_1' and F_2, we can convert the original graph into one bipartite graph that contains all edges in the original graph (Fig. 4d). Additionally, corresponding vertices in F_k and F_k' could be connected each other by vertical edges.

The oversampled graph is a bipartite graph that has the vertex set $V_{\mathrm{OS}} = V_0 \cup F_1' \cup F_2'$ and edge set $E_{\mathrm{OS}} = E_{F_1 \leftrightarrow F_3} \cup E_{F_2 \leftrightarrow F_3} \cup E_{F_1 \leftrightarrow F_2'} \cup E_{F_1' \leftrightarrow F_2}$. Since there are no edges within F_{12} and also within $F_1' \cup F_2' \cup F_3$, $G_{\mathrm{OS}} = (V_{\mathrm{OS}}, E_{\mathrm{OS}})$ is a bipartite graph as shown in Fig. 4e. Even if there are edges $E_{F_1 \leftrightarrow F_1'}$ and $E_{F_2 \leftrightarrow F_2'}$, clearly it is still a bipartite graph.

If some of the vertices in F_{12} only have connections to F_3, they are isolated in B_2. Hence, there is no need to append these vertices to V_{OS}. Therefore, the number of appended vertices, i.e., N_1, is $N_1 = |F_1'| + |F_2'| = |F_1| + |F_2| - |I|$, where $|I|$ is the number of isolated vertices in F_{12}.

3.2 *K*-Colorable Graphs

For K-colorable graphs where $K \geq 4$, the method described above can be extended to make one bipartite graph including all of the edges of the original graph. We assume that the vertices of the original graph $G = (V, E)$ are assigned colors and

divided into K sets $F_1, F_2, \ldots, F_K$. Figure 5 shows two examples of the oversampled bipartite graphs for a five-colorable graph. The oversampled bipartite graph is generated according to the following steps:

1. The foundation bipartite graph $G_f = (L_f, H_f, E_f)$ is made from the original graph. $L_f = \{F_1, F_2, \ldots, F_l\}$ and $H_f = \{F_{l+1}, F_{l+2}, \ldots, F_K\}$, where l is an arbitrary integer value satisfying $1 \leq l \leq K$. E_f is defined as the edge set containing all edges between L_f and H_f (Fig. 5b, e).
2. The *complementary graph* $\overline{G} = (V, E \setminus E_f)$ is computed. $\overline{G}$ has two disjoint graphs: an l-colorable graph $\overline{G}_1 = (L_f, E_{L_f})$ and a $(K-l)$-colorable graph $\overline{G}_2 = (H_f, E_{H_f})$, where E_V represents the edges connecting vertices in V. It is shown in Fig. 5c, f.
3. Appended vertices F_1' are placed directly above each vertex in F_1 of G_f.
4. The edge set $E_{F_1 \leftrightarrow \{F_2, \ldots, F_l\}}$ is extracted from the original graph and it is used to connect F_1' and $\{F_2, \ldots, F_l\}$. Since L_f is disjointed vertices in G_f, $G_{\mathrm{OS}} = (L_f, H_f \cup F_1', E_f \cup E_{F_1' \leftrightarrow \{F_2, \ldots, F_l\}})$ is still a bipartite graph.
5. Steps 3 to 4 are repeated for $F_2, \ldots, F_l$ to yield oversampled sets $F_2', \ldots, F_l'$ and appended new edges to G_{OS}.
6. Similar operations to Steps 3 to 5 can also be applied to $\overline{G}_2$. As a result, the sets $F_{l+1}', \ldots, F_K'$ and the edges in $\overline{G}_2$ are appended to G_{OS}.

Consequently, the sets $F_1', \ldots, F_K'$ and the edges corresponding to $E \setminus E_f$ are added to the foundation bipartite graph. Based on the above operations, an oversampled bipartite graph G_{OS} containing all edges of the original graph is generated as shown in Fig. 5d, g. Note that L_{OS} and H_{OS} of the oversampled graph become

$$\begin{aligned} L_{\mathrm{OS}} &= F_1 \cup \ldots \cup F_l \cup F_{l+1}' \cup \ldots \cup F_K', \\ H_{\mathrm{OS}} &= F_1' \cup \ldots \cup F_l' \cup F_{l+1} \cup \ldots \cup F_K. \end{aligned} \tag{20}$$

Similar to the three-colorable case, vertical edges can be appended and isolated vertices in $\overline{G}$ will be removed. As a result, the number of the vertices in these sets can be represented as

$$|L_{\mathrm{OS}}| = \sum_{i=1}^{l} |F_i| + \sum_{i=l+1}^{K} |F_i'| = N - |I_{H_f}|, \tag{21}$$

$$|H_{\mathrm{OS}}| = \sum_{i=l+1}^{K} |F_i| + \sum_{i=1}^{l} |F_i'| = N - |I_{L_f}|, \tag{22}$$

where $|I_{L_f}|$ and $|I_{H_f}|$ are the number of isolated vertices in $\overline{G}_1$ and $\overline{G}_2$, respectively. The number of appended vertices is, therefore, represented as $2N - |I_{L_f}| - |I_{H_f}|$.

According to the choice of l, there exists $\lfloor \frac{K}{2} \rfloor$ variations of the oversampled graph for K-colorable graphs. For the special case of $l = 1$, L_f is equal to F_1 and $\overline{G}_1$ has no edges as shown in Fig. 5e, f. Therefore, we do not need to append vertices just above L_f, and the oversampled bipartite graph becomes

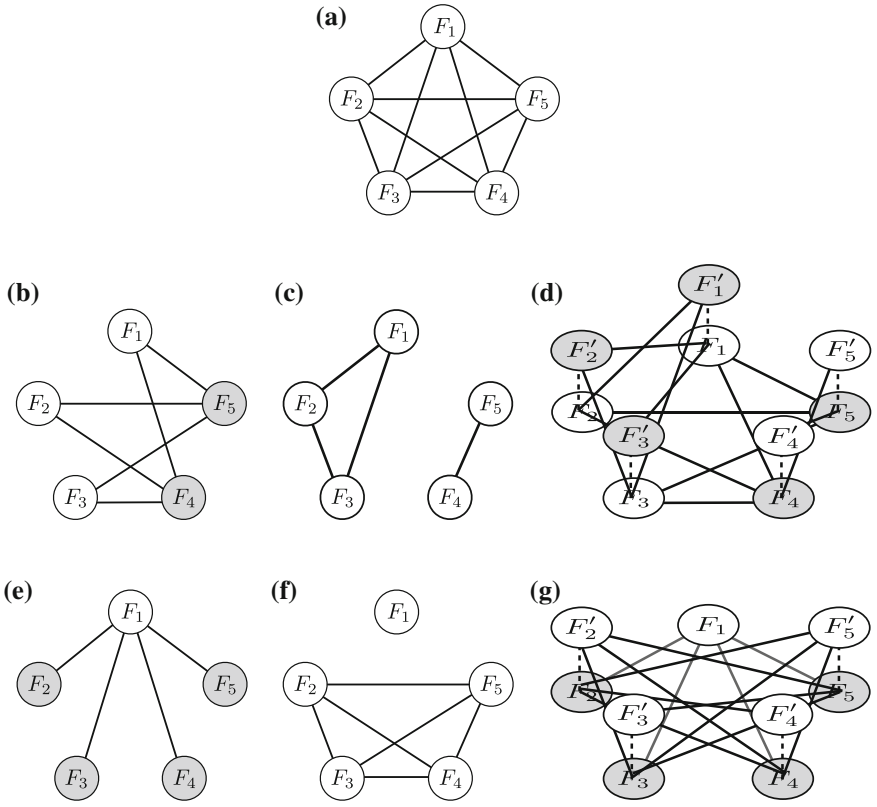


Fig. 5 Examples of oversampled bipartite graphs for a five-colorable graph. The white and gray circles represent sets L_{OS} and H_{OS}, respectively. **a** Original graph. **b** Foundation bipartite graph with $l = 3$. **c** $\overline{G}$ with $l = 3$. **d** Oversampled bipartite graph with $l = 3$. The dashed lines indicate the vertical edges. **e** Foundation bipartite graph with $l = 1$. **f** $\overline{G}$ with $l = 1$. **g** Oversampled bipartite graph with $l = 1$

$$L_{\mathrm{OS}} = F_1 \cup F'_2 \cup \ldots \cup F'_K, \tag{23}$$

$$H_{\mathrm{OS}} = F_2 \cup \ldots \cup F_K. \tag{24}$$

Similarly, when $l = K - 1$, the oversampled bipartite graph becomes

$$L_{\mathrm{OS}} = F_1 \cup \ldots \cup F_{K-1}, \tag{25}$$

$$H_{\mathrm{OS}} = F'_1 \cup \ldots \cup F'_{K-1} \cup F_K. \tag{26}$$

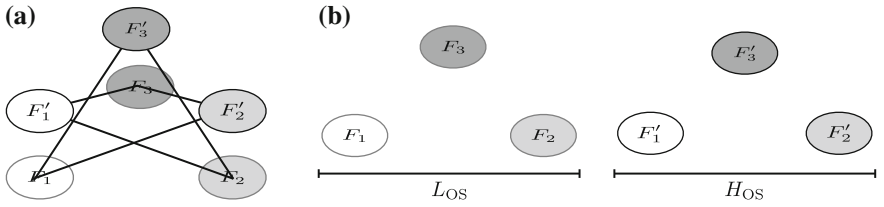


Fig. 6 **a** Bipartite double cover of a three-colorable graph. **b** Its set of L_{OS} and H_{OS}

3.3 Relationship with Bipartite Double Cover

In graph theory, the bipartite double cover of a graph G is defined as the tensor product $G_{\mathrm{BDC}} = G \otimes K_2$, where K_2 is the complete graph of two vertices [2, 7, 18]. G_{BDC} exactly has $2N$ vertices and $2|E|$ edges.

An example of the bipartite double cover of a three-colorable graph (Fig. 4a) is shown in Fig. 6. It is clear that $G_{\mathrm{BDC}} = (\{F_1, \ldots, F_K\}, \{F'_1, \ldots, F'_K\}, E_{\mathrm{BDC}})$, where $E_{\mathrm{BDC}} := E_{\{F_1,\ldots,F_K\}\leftrightarrow\{F'_1,\ldots,F'_K\}}$. The bipartite double cover is equivalent to the oversampled graph introduced in the previous subsection, in the case of $l = K$ without vertical edges.

The adjacency matrix of G_{BDC} can be represented as

$$\mathbf{A}_{\mathrm{BDC}} = \begin{bmatrix} \mathbf{0} & \mathbf{A} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \tag{27}$$

and its symmetric normalized graph Laplacian is

$$\underline{\mathbf{L}}_{\mathrm{BDC}} = \begin{bmatrix} \mathbf{I} & -\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} \\ -\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} & \mathbf{I} \end{bmatrix}. \tag{28}$$

If $\boldsymbol{u}_i$ is an eigenvector of $\underline{\mathbf{L}}$ with the eigenvalue λ_i, one can immediately see that $\frac{1}{\sqrt{2}}[\boldsymbol{u}_i^\top \ \boldsymbol{u}_i^\top]^\top$ and $\frac{1}{\sqrt{2}}[\boldsymbol{u}_i^\top \ -\boldsymbol{u}_i^\top]^\top$ are eigenvectors of $\underline{\mathbf{L}}_{\mathrm{BDC}}$ with eigenvalues $\lambda_{i,\mathrm{BDC}} = \lambda_i$ and $2 - \lambda_i$, respectively.

If we define the oversampled graph signal as $\boldsymbol{f}_{\mathrm{OS}} = [\boldsymbol{f}_0^\top \ \boldsymbol{f}_0^\top]^\top$, i.e., duplicating the original signal values at the corresponding appended vertices, its graph Fourier coefficient associated with λ_i is

$$\boldsymbol{u}_{j,\mathrm{BDC}}^\top \boldsymbol{f}_{\mathrm{OS}}\big|_{\lambda_{j,\mathrm{BDC}}=\lambda_i} = \frac{1}{\sqrt{2}}\begin{bmatrix}\boldsymbol{u}_i^\top & \boldsymbol{u}_i^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{f}_0 \\ \boldsymbol{f}_0\end{bmatrix} = \sqrt{2}\boldsymbol{u}_i^\top \boldsymbol{f}_0 = \sqrt{2}\widetilde{f}_0(i), \tag{29}$$

and that associated with $2 - \lambda_i$ is

$$\boldsymbol{u}_{j,\mathrm{BDC}}^\top \boldsymbol{f}_{\mathrm{OS}}\big|_{\lambda_{j,\mathrm{BDC}}=2-\lambda_i} = \frac{1}{\sqrt{2}}\begin{bmatrix}\boldsymbol{u}_i^\top & -\boldsymbol{u}_i^\top\end{bmatrix}\begin{bmatrix}\boldsymbol{f}_0 \\ \boldsymbol{f}_0\end{bmatrix} = 0. \tag{30}$$

As a result, we can obtain only N nonzero graph Fourier coefficients which are equal to those with the original graph Fourier coefficient. In other words, the graph Fourier spectrum using the bipartite double cover is the same as the one using the original graph.

On the other hand, let us define the adjacency matrix of the foundation bipartite graph G_f of the proposed method as $\mathbf{A}_f$ and that of the remaining graph $\overline{G}$ as $\mathbf{A}_r$. For simplicity, we will consider the expansion method without vertical edges. The adjacency matrix and degree matrix of our approach become

$$\mathbf{A}_{\mathrm{OS}} = \begin{bmatrix} \mathbf{A}_f & \mathbf{A}_r \\ \mathbf{A}_r & \mathbf{0} \end{bmatrix} \tag{31}$$

and

$$\mathbf{D}_{\mathrm{OS}} = \begin{bmatrix} \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_r \end{bmatrix}, \tag{32}$$

where $\mathbf{A} = \mathbf{A}_f + \mathbf{A}_r$ and $\mathbf{D} = \mathbf{D}_f + \mathbf{D}_r$. Its symmetric normalized graph Laplacian is

$$\underline{\mathbf{L}}_{\mathrm{OS}} = \begin{bmatrix} \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}_f\mathbf{D}^{-1/2} & -\mathbf{D}^{-1/2}\mathbf{A}_r\mathbf{D}_r^{-1/2} \\ -\mathbf{D}_r^{-1/2}\mathbf{A}_r\mathbf{D}^{-1/2} & \mathbf{I} \end{bmatrix}. \tag{33}$$

If we assume the oversampled graph Laplacian has eigenvectors $[\boldsymbol{u}_i^\top \ \boldsymbol{u}_i^\top]^\top$ that corresponds to the eigenvalue λ_i of $\underline{\mathbf{L}}$, then it must satisfy

$$\begin{aligned} \underline{\mathbf{L}}_{\mathrm{OS}} \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{u}_i \end{bmatrix} &= \begin{bmatrix} \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}_f\mathbf{D}^{-1/2} & -\mathbf{D}^{-1/2}\mathbf{A}_r\mathbf{D}_r^{-1/2} \\ -\mathbf{D}_r^{-1/2}\mathbf{A}_r\mathbf{D}^{-1/2} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{u}_i \end{bmatrix} \\ &= \lambda_i \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{u}_i \end{bmatrix}. \end{aligned} \tag{34}$$

The constraint can be simplified as

$$(\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}_f\mathbf{D}^{-1/2} - \mathbf{D}^{-1/2}\mathbf{A}_r\mathbf{D}_r^{-1/2})\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i. \tag{35}$$

On the other hand, the original graph Laplacian satisfies

$$\begin{aligned} \lambda_i \boldsymbol{u}_i &= \underline{\mathbf{L}}\boldsymbol{u}_i \\ &= (\mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})\boldsymbol{u}_i \\ &= (\mathbf{I} - \mathbf{D}^{-1/2}(\mathbf{A}_f + \mathbf{A}_r)\mathbf{D}^{-1/2})\boldsymbol{u}_i. \end{aligned} \tag{36}$$

Comparing (35) and (36), $\mathbf{D}^{-1/2}\mathbf{A}_r\mathbf{D}^{-1/2} = \mathbf{D}^{-1/2}\mathbf{A}_r\mathbf{D}_r^{-1/2}$ has to be satisfied. As a result, $\underline{\mathbf{L}}_{\mathrm{OS}}$ has the eigenvector $\frac{1}{\sqrt{2}}[\boldsymbol{u}_i^\top \ \boldsymbol{u}_i^\top]^\top$ associated with $\lambda_{i,\mathrm{OS}} = \lambda_i$ iff $\mathbf{D}_r = \mathbf{D}$, which is the case of the bipartite double cover. In other cases, the eigenvalues and eigenvectors of the oversampled graph are different from those of the original graph.

Hence, we can obtain a different graph Fourier spectrum from that of the original graph by using our approach with $l < K$. Additionally, $\mathbf{A}_r$ has columns and rows whose elements are all zero when $l = K - 1$ or the remaining graph has isolated vertices. In this case, the size of $\underline{\mathbf{L}}_{\mathrm{OS}}$ is less than $2N$.

In summary, the oversampled way shown in this chapter is a bipartite double cover when $l = K$ without vertical edges, and its graph Fourier spectrum is the same as that of the original graph except for a trivial scaling. In contrast, the method with $l < K$ has different eigenvectors from those of the original graph, and its redundancy is less than that of the bipartite double cover.

4 Examples of Graph Oversampling

Here, let us take a look at some examples of graph oversampling.

4.1 Graph Oversampling for Images

Images can be viewed as graph signals by connecting each pixel with its neighboring ones (for example four or eight-neighbors) [4, 12]. Assume the eight-neighbor graph shown in Fig. 7a. Since this graph is four-colorable, it can be decomposed into rectangular (Fig. 7b) and diagonal (Fig. 7c) bipartite subgraphs.

If a graph filter bank for bipartite graphs [11, 12] is applied to the signal on one bipartite graph, the diagonal or rectangular connections will be ignored in a single stage. A multidimensional transform can be applied to multiple bipartite subgraphs to resolve the problem for the graph filter banks [11, 12]. However, we cannot perform the transform that considers the rectangular and diagonal connections simultaneously.

In contrast, the above problem can be (partially) solved by exploiting the graph oversampling. For example, we can append diagonal edges to the rectangular bipartite graph (if this is the foundation graph) while keeping the oversampled graph bipartite.

The overall transform is shown in Fig. 8. We can iterate this process on the LL subband to realize a multilevel image decomposition.

The rectangular oversampled image graph is illustrated in Fig. 7d. In the figure, the gray vertices are appended just above the white vertices of the rectangular bipartite graph, and they have diagonal edges connecting them to the white vertices. If we need the number of lowpass coefficients to be equal to the critically sampled case, we can do so by appending vertices only on the H_{OS} side. The number of the white vertices, i.e., $|L_{\mathrm{OS}}|$, is exactly equal to that of the foundation bipartite graph.

In the second stage, the oversampling way is opposite; we can make the oversampled diagonal bipartite graph (Fig. 7e) for the decomposition. Thus, with the oversampled bipartite graph, images are transformed with the rectangular graphs *plus* diagonal connections as well as diagonal graphs *plus* rectangular connections.

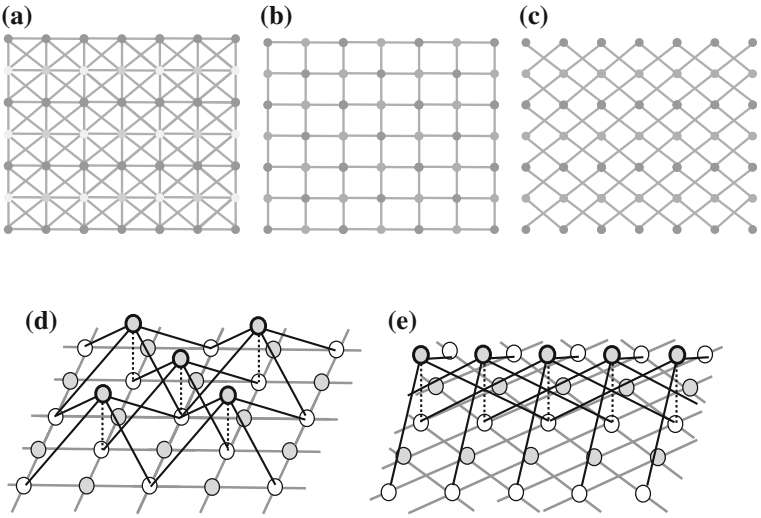


Fig. 7 **a** Image graph. **b** Rectangular bipartite subgraph. **c** Diagonal bipartite subgraph. **d** Oversampled rectangular bipartite graph. **e** Oversampled diagonal bipartite graph. The appended vertices are black circles filled with blue, and the appended edges are black lines

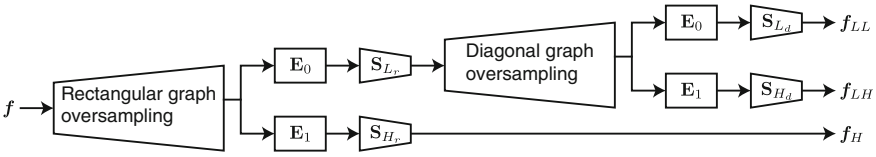


Fig. 8 One-level decomposition of images. $\mathbf{S}_{\chi_r}$ and $\mathbf{S}_{\chi_d}$ where $\chi \in \{L, H\}$ denote the downsampling operations of the rectangular and diagonal graphs, respectively

4.2 Graph Oversampling for Ring Graphs

Ring graphs with an even number of vertices are bipartite, but those with an odd number of vertices are three-colorable as shown in Fig. 9a. We consider the effect of the graph oversampling in this case. Let us assume that the original graph has $2n + 1$ vertices, and F_1, F_2, and F_3 are the white, light gray, and dark gray vertices illustrated in Fig. 9a, respectively. We also assume that F_1 and F_3 each have n vertices and F_2 has only one vertex.

The bipartite subgraphs B_1 and B_2 of this ring graph are shown in Fig. 9c, d. For applying the multidimensional decomposition, a graph filter bank must be applied to each bipartite graph separately. After the signal decomposition using the critically sampled graph filter bank, the LL and HL channels each have n vertices, whereas the LH channel has only one vertex (the HH channel is empty). Therefore, the number of coefficients in each channel is heavily biased.

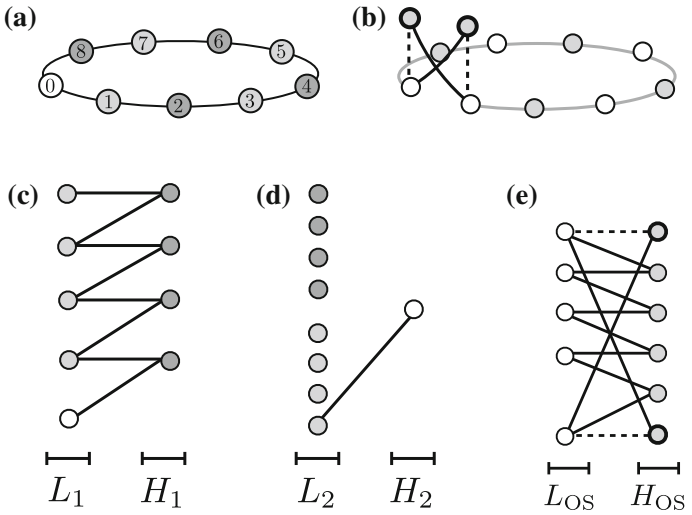


Fig. 9 **a** Ring graph ($n = 4$). **b** Oversampled bipartite graph. **c** Sets L_1 and H_1 of the bipartite subgraph B_1. **d** Sets L_2 and H_2 of the bipartite subgraph B_2. **e** Sets L_{OS} and H_{OS} of the oversampled bipartite graph

It is also clear that the transform for B_1 treats all edges except the one between v_0 and v_1, whereas the transform for B_2 handles only the edge between v_0 and v_1. Additionally, the relationship between v_2 and v_{2n} becomes very weak as a result of the bipartite decomposition, in spite of their 3-hop neighborhood connection in the original ring graph. In this bipartite decomposition, v_2 and v_{2n} are in the $(2n - 2)$-hop neighborhood of B_1 and are not connected to B_2.

The graph oversampling alleviates that effect. The oversampled graph of the ring graph with $2n + 1$ vertices is shown in Fig. 9b, e, respectively. As shown in Fig. 9e, L_{OS} has vertices in F_1 and F_2, and H_{OS} has those in F_1', F_2' and F_3. Therefore, the number of vertices in two disjoint sets are $n + 1$ and $n + 2$. The redundancy is only $(2n + 3)/(2n + 1)$, that is almost critically-sampled for a large n. In the oversampled bipartite graph, all adjacent vertices are connected and all edges of the original graph can be considered in a single stage transform. Furthermore, if we append vertical edges, vertices v_2 and v_{2n} are in a 4-hop neighborhood and have a strong connection like that of the original graph.

4.3 Signal Spread

To demonstrate the advantage of the oversampled bipartite graph, the signal spreads of a critically sampled bipartite graph and an oversampled one are also compared. The original graph in this case was the Petersen graph, and it was decomposed into the two bipartite subgraphs. The input signal is shown in Fig. 10a. The lowpass filtered

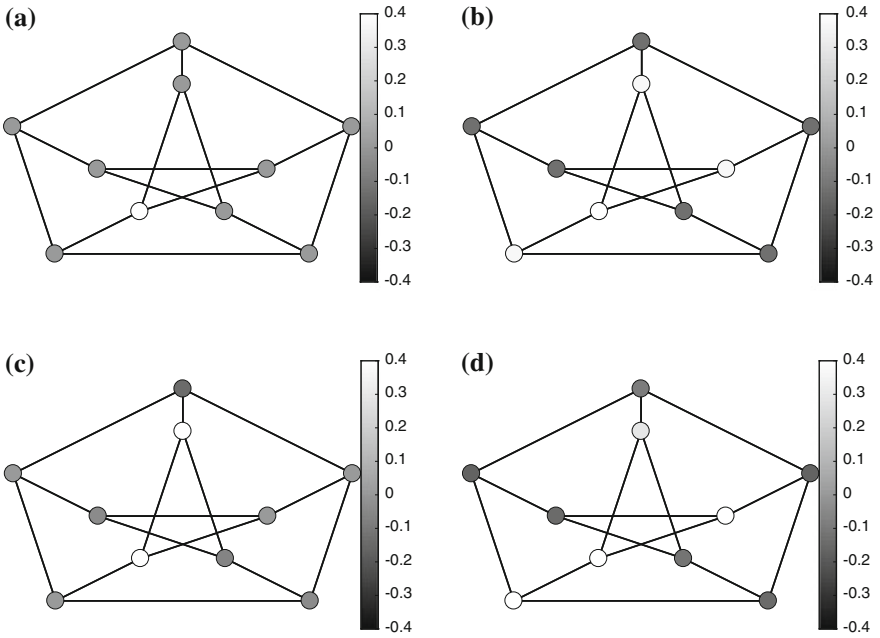


Fig. 10 Signal spread. **a** Input signal. **b** Lowpass filtered signal using the (non-bipartite) original graph. **c** Lowpass filtered signal using bipartite subgraph. **d** Lowpass filtered signal using oversampled bipartite graph

signals are shown in Fig. 10b, c, d. As expected, the spread of the signal after using the oversampled bipartite graph is very similar to that of the original (non-bipartite) graph.

5 Design Method of Oversampled Spectral Graph Filter Banks

In this section, the details of the perfect reconstruction condition are described for the oversampled graph transforms introduced in Sect. 2.2.

5.1 Transfer Function and Perfect Reconstruction Condition

Consider the M-channel graph filter bank shown in Fig. 11. After filtering with $\mathbf{E}_k$ in (6), the first $M/2$ channels pass $|L|$ signals, whereas the last ones keep $|H|$ signals, according to the coloring result of the bipartite graph. In the synthesis side, the transformed coefficients are upsampled then filtering with $\mathbf{R}_k$ in (8) is performed.

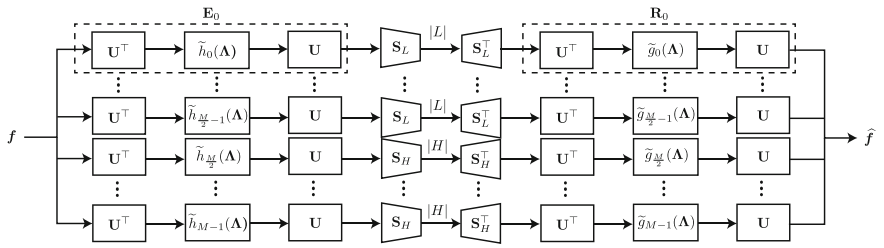


Fig. 11 M-channel oversampled graph filter bank

The reconstructed signal in the kth subband is represented as

$$\widehat{\boldsymbol{f}}_k = \mathbf{U}\widetilde{g}_k(\boldsymbol{\Lambda})\mathbf{U}^\top \mathbf{S}_k^\top \mathbf{S}_k \mathbf{U}\widetilde{h}_k(\boldsymbol{\Lambda})\mathbf{U}^\top \boldsymbol{f}. \tag{37}$$

Therefore, the overall transfer function $\mathbf{T}$ is

$$\begin{aligned}\mathbf{T} &= \sum_{k=0}^{M-1} \mathbf{G}_k \mathbf{S}_k^\top \mathbf{S}_k \mathbf{H}_k \\ &= \frac{1}{2}\sum_{k=0}^{M-1} \mathbf{G}_k \mathbf{H}_k + \frac{1}{2}\sum_{\ell=0}^{M/2-1} (\mathbf{G}_{M/2+\ell}\mathbf{I}'_N\mathbf{H}_{M/2+\ell} - \mathbf{G}_\ell \mathbf{I}'_N \mathbf{H}_\ell),\end{aligned} \tag{38}$$

where

$$[\mathbf{I}'_N]_{mm} = \begin{cases} 1 & \text{if } f(m) \text{ belongs to } H \\ -1 & \text{if } f(m) \text{ belongs to } L. \end{cases} \tag{39}$$

In (38), $\frac{1}{2}\sum_{k=0}^{M-1} \mathbf{G}_k\mathbf{H}_k$ corresponds to the main graph frequency, and the remaining term corresponds to aliasing. Consequently, the perfect reconstruction condition is represented as

$$\begin{aligned}&\frac{1}{2}\sum_{k=0}^{M-1} \mathbf{G}_k\mathbf{H}_k = \mathbf{I} \\ &\frac{1}{2}\sum_{\ell=0}^{M/2-1} (\mathbf{G}_{M/2+\ell}\mathbf{I}'_N\mathbf{H}_{M/2+\ell} - \mathbf{G}_\ell\mathbf{I}'_N\mathbf{H}_\ell) = \mathbf{0}.\end{aligned} \tag{40}$$

By utilizing the aliasing effect [11, 12] $\mathbf{I}'\boldsymbol{u}_i\boldsymbol{u}_i^\top = \boldsymbol{u}_j\boldsymbol{u}_j^\top\mathbf{I}'$, where $\lambda_j = 2 - \lambda_i$, the above condition can be rewritten as the filter condition as follows:

$$\sum_{k=0}^{M-1} \widetilde{g}_k(\lambda)\widetilde{h}_k(\lambda) = 2 \tag{41}$$

$$\sum_{k=0}^{M/2-1} \widetilde{g}_k(\lambda)\widetilde{h}_k(2-\lambda) - \widetilde{g}_{k+M/2}(\lambda)\widetilde{h}_{k+M/2}(2-\lambda) = 0 \tag{42}$$

for any λ. The latter equation is valid if we choose $\widetilde{g}_k(\lambda) = \widetilde{h}_{k+M/2}(2-\lambda)$ and $\widetilde{g}_{k+M/2}(\lambda) = \widetilde{h}_k(2-\lambda)$. Accordingly, (41) becomes

$$\sum_{k=0}^{M/2-1} \widetilde{g}_k(\lambda)\widetilde{h}_k(\lambda) + \widetilde{g}_k(2-\lambda)\widetilde{h}_k(2-\lambda) = 2. \tag{43}$$

As a result, the product filter $\widetilde{p}_k(\lambda) := \widetilde{g}_k(\lambda)\widetilde{h}_k(\lambda)$ must satisfy the following condition:

$$\sum_{k=0}^{M/2-1} \widetilde{p}_k(\lambda) + \widetilde{p}_k(2-\lambda) = 2. \tag{44}$$

5.2 Filter Design

For simplicity, we consider the case of $M = 4$. It can be easily generalized for larger M.

Let us define $\widetilde{q}(\lambda) := \widetilde{p}_0(\lambda) + \widetilde{p}_1(\lambda)$. For $M = 4$, (44) is rewritten as

$$\widetilde{q}(\lambda) + \widetilde{q}(2-\lambda) = 2. \tag{45}$$

This equation is the same as that of a two-channel biorthogonal graph filter bank [11]. Therefore, the design problem boils down to separating the critically sampled product filter $\widetilde{q}(\lambda)$ into lowpass and bandpass (Fig. 12) filters $\widetilde{p}_0(\lambda)$ and $\widetilde{p}_1(\lambda)$ such that the *sum* of filters is $\widetilde{q}(\lambda)$.

Let us assume that a lowpass product filter $\widetilde{p}_0(\lambda)$ is an arbitrarily chosen low-pass filter with K zero at $\lambda = 0$. By changing the variable of $\lambda = 1 + l$ [11], $\widetilde{p}_0(1+l)$ can be expressed as

$$\widetilde{p}_0(1+l) = (1+l)^K \left(\sum_{m=0}^{J_0} \alpha_m l^m \right), \tag{46}$$

where α_m is an arbitrary parameter and $K + J_0$ is the filter order.

Following [11], $\widetilde{q}(1+l)$ has the degree $2K-1$ and its even degree must be zero from the halfband condition. Hence, $\widetilde{q}(1+l)$ is represented as

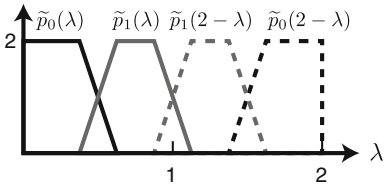


Fig. 12 Four-channel product filter example

$$\widetilde{q}(1+l) = (1+l)^K \left(1 + \sum_{m=1}^{K-1} r_m l^m\right) = 1 + \sum_{n=0}^{K-1} c_{2n+1} l^{2n+1}, \tag{47}$$

which is, of course, the same as that in [11] and has a unique solution satisfying (45). Finally, the remaining product filter $\widetilde{p}_1(\lambda)$ can be defined as follows:

$$\widetilde{p}_1(1+l) = \widetilde{q}(1+l) - \widetilde{p}_0(1+l) = (1+l)^K \left(\sum_{m=0}^{J_1} \beta_m l^m\right). \tag{48}$$

Example: $K = 2$ zeros at $l = -1$
We assume $J_0 = 1$. As in (46) and (48), $\widetilde{p}_0(1+l)$ and $\widetilde{p}_1(1+l)$ are

$$\widetilde{p}_0(1+l) = (1+l)^2 (\alpha_0 + \alpha_1 l) \tag{49}$$

$$\widetilde{p}_1(1+l) = (1+l)^2 (\beta_0 + \beta_1 l), \tag{50}$$

where α_0 and α_1 are arbitrarily chosen parameters. Then, the sum of the product filter $\widetilde{q}(1+l)$ is defined as

$$\widetilde{q}(1+l) = \frac{1}{2}(1+l)^2(2-l), \tag{51}$$

which is an odd-order polynomial and it is the same product filter as that in [11]. To guarantee the perfect reconstruction, β_0 and β_1 must be

$$\begin{aligned} \beta_0 &= 1 - \alpha_0 \\ \beta_1 &= -\left(\alpha_1 + \frac{1}{2}\right). \end{aligned} \tag{52}$$

That is, we can add free parameters (α_0 and α_1) to design a halfband filter, and this will lead to better filter characteristics.

A similar derivation is possible for general M-channel graph filter banks. In that case, the parameters for $(M-2)/2$ product filters can be freely chosen, and the last product filter can be designed so that the entire product filter $\widetilde{q}(\lambda)$ is a maximally flat halfband filter.

5.3 *Design Examples*

As mentioned above, we can use arbitrary parameters to design filters. In what follows, we will use a sequential design method to obtain good filter banks:

1. Design $\widetilde{h}_k(1-l)$ and $\widetilde{g}_k(1-l)$ $(k = 0, \ldots, M/2-2)$ with k_0 and k_1 zeros (where $K = k_0 + k_1$ in (46)–(47)) at $l = 1$ ($\lambda = 2$). They are represented as follows:

$$\begin{aligned}\widetilde{h}_k(1-l) &= (1-l)^{k_0}\sum_{m=0}^{J_k^{(h)}} s_{h,k,m}l^m \\ \widetilde{g}_k(1-l) &= (1-l)^{k_1}\sum_{m=0}^{J_k^{(g)}} s_{g,k,m}l^m\end{aligned} \tag{53}$$

where $s_{h,k,m}$ and $s_{g,k,m}$ are filter coefficients. i.e., the product filter $\widetilde{p}_k(1-l) = \widetilde{g}_k(1-l)\widetilde{h}_k(1-l)$ can be represented as

$$\widetilde{p}_k(1-l) = (1-l)^K\left(\sum_{m=0}^{J_k^{(h)}+J_k^{(g)}} \alpha_{k,m}l^m\right). \tag{54}$$

The numbers of arbitrary parameters in $\widetilde{h}_k(1-l)$ and $\widetilde{g}_k(1-l)$ are $J_k^{(h)}$ and $J_k^{(g)}$, respectively.

The filters are optimized by using the cost function of the stopband attenuation shown below:

$$C(h_k) = w_0\int_{l\in\omega_p}(\sqrt{2}-\widetilde{h}_k(1-l))^2dl + w_1\int_{l\in\omega_s}\widetilde{h}_k^2(1-l)dl, \tag{55}$$

where w_0 and w_1 are weights and ω_p and ω_s are defined as the passband and stopband ($-1 \le \omega_p, \omega_s \le 1$), respectively.

2. Calculate the two-channel halfband filter pair $\widetilde{q}(1-l) = \widetilde{q}(\lambda)$ and $\widetilde{q}(1+l) = \widetilde{q}(2-\lambda)$ with K zeros at $l = 1$ so that the pair satisfies (45).
3. Calculate the bandpass product filter[1]

$$\begin{aligned}\widetilde{p}_{\frac{M}{2}-1}(1-l) &= \widetilde{q}(1-l) - \sum_{k=0}^{M/2-2}\widetilde{p}_k(1-l) \\ &= (1-l)^K\widetilde{p}'_{\frac{M}{2}-1}(1-l).\end{aligned} \tag{56}$$

4. Factorize $\widetilde{p}_{\frac{M}{2}-1}(1-l)$ into two bandpass filters $\widetilde{h}_{\frac{M}{2}-1}(1-l)$ and $\widetilde{g}_{\frac{M}{2}-1}(1-l)$. Test all combinations of roots as long as both bandpass filters have real-valued coefficients, and select the best combination, i.e., the filters minimizing $C(\widetilde{h}_{\frac{M}{2}-1}) + C(\widetilde{g}_{\frac{M}{2}-1})$.

Figure 13 shows an example of oversampled graph filter banks. The arbitrary lowpass filters $\widetilde{h}_0(\lambda)$ and $\widetilde{g}_0(\lambda)$ are designed to have degree 10 and 11, respectively.

[1] There always exists $\widetilde{q}(1-l)$ which satisfies the perfect reconstruction condition (45) [11]. Therefore, $\widetilde{p}_{\frac{M}{2}-1}(1-l)$ also has a unique solution with real coefficients as long as all of the arbitrary design filters have real coefficients. Additionally, since the perfect reconstruction condition is only imposed on $\widetilde{p}_{\frac{M}{2}-1}(1-l)$, $J_k^{(h)}$ and $J_k^{(g)}$ in (53) can be set arbitrarily regardless of K.

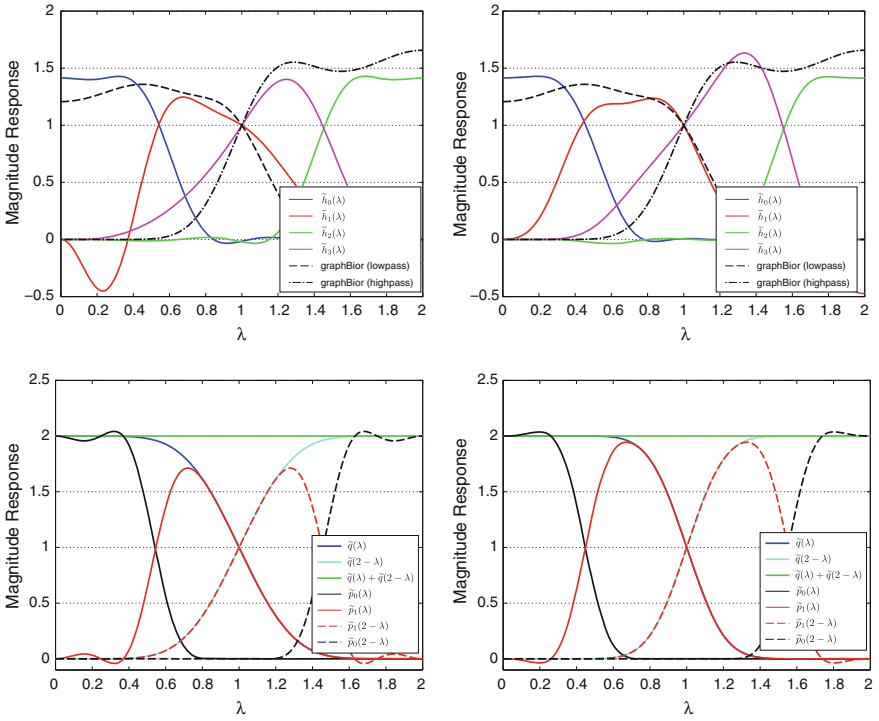


Fig. 13 Four-channel oversampled graph filter banks. From left to right: $(k_0, k_1) = (2, 2)$ and $(4, 4)$. Top row: analysis filter bank (black lines indicate graphBior(6, 6) [11]). Bottom row: halfband filters

We used $-1 \leq \omega_p \leq -0.84$ and $-0.75 \leq \omega_s \leq 1$. For comparison, the frequency responses of the critically sampled graphBior(6, 6) [11] are also plotted. They have 13-taps for the lowpass filter and 12-taps for the highpass filter. It is clear that the oversampled lowpass filter has a sharper transition band and a more uniform response in the passband than the critically sampled graph filter banks have. In the following experiments, we use the oversampled filter bank with $(k_0, k_1) = (4, 4)$ zeros.

6 Experimental Results

In this section, several experiments are shown to numerically compare the performance of oversampled graph transforms.

6.1 Image Processing

First of all, nonlinear approximation of images is performed to compare the graph-based methods and traditional image processing. As the traditional methods, the standard separable CDF 9/7 wavelet filter bank [5] and the Laplacian pyramid for regular signals [3] have been selected. The Laplacian pyramid for regular signals used 9/7 filters and a reconstruction scheme using the pseudo inverse [6].

For graph-based methods, the critically sampled graphBior filter bank [11], the Laplacian pyramid for graph signals [19], and the graph oversampling followed by the graphBior filter bank have also been compared. The graph-based methods used the same filters (graphBior(5, 5) [11]).

The graphBior used an edge-aware image graph [10]. The edge-aware image graphs were made as follows. The links around the edges of the images are classified into *regular* or *less-reliable* links. They were determined by checking that the difference in pixel intensity between the edge pixels is more than or less than a certain threshold. The weights of less-reliable links are set to zero or reduced to a value lower than those of the regular links (Fig. 7a, b, c). For example, the edge-aware image graphs of Fig. 14a are shown in Fig. 14b, c. The graph Laplacian pyramid used the same graph and downsampling operation as graphBior for the lowpass channel.

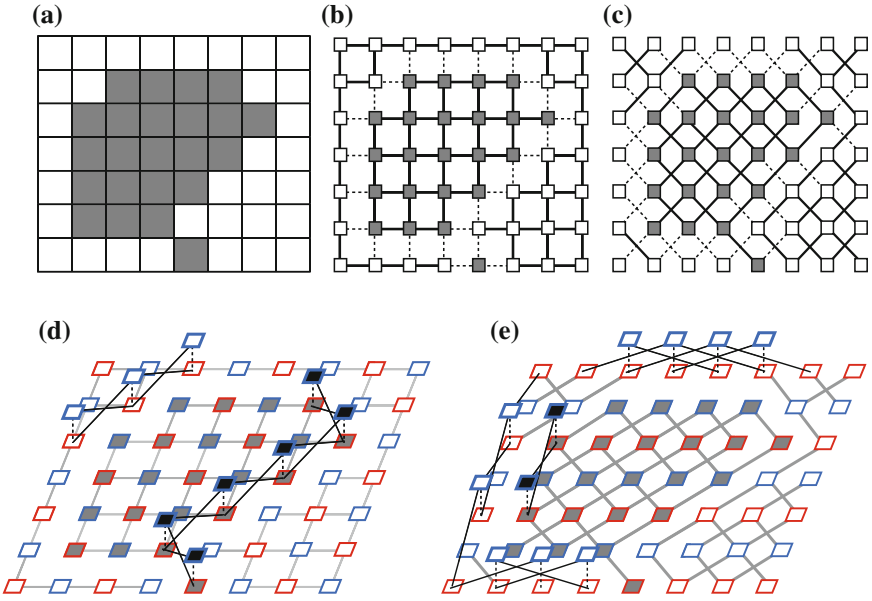


Fig. 14 **a** Original image. **b** Edge-aware rectangular bipartite graph. The solid and dashed lines are regular and less-reliable links, respectively. **c** Edge-aware diagonal bipartite graph. **d** Oversampled edge-aware rectangular bipartite graph. The black lines are additional edges. The dashed black lines indicate vertical edges. The red vertices contain the lowpass component, and the blue vertices contain the highpass component after downsampling. **e** Oversampled edge-aware diagonal bipartite graph

Table 1 Reconstruction of Images Using NLA: PSNR (dB)

Fraction of highpass coeffs.	0.00	0.01	0.02	0.04	0.08	0.16
Ballet						
9/7 filter [5]	24.57	31.76	35.83	43.10	52.87	60.05
Laplacian pyramid [3]	24.57	30.49	33.71	38.27	45.74	55.29
GraphBior [12]	31.66	42.39	45.98	50.69	56.10	61.37
Graph laplacian pyramid [19]	31.66	41.37	44.16	47.70	52.00	56.96
GraphBior + OSGL (without vertical edges)	30.78	45.12	50.12	55.04	59.35	63.80
GraphBior + OSGL (with vertical edges)	**33.01**	**49.97**	**53.51**	**57.02**	**60.35**	**64.58**
Synthetic						
9/7 filter	30.44	37.95	42.96	50.81	**66.73**	**110.10**
Laplacian pyramid	30.44	36.30	39.43	44.23	51.64	66.61
GraphBior	32.92	40.77	44.18	49.34	58.94	76.88
Graph laplacian pyramid	32.92	39.77	42.35	45.60	50.48	58.83
GraphBior + OSGL (without vertical edges)	33.22	40.40	43.99	49.83	60.20	81.62
GraphBior + OSGL (with vertical edges)	**34.29**	**43.36**	**46.99**	**52.53**	61.53	82.38
Cameraman						
9/7 filter	20.66	23.38	25.30	27.61	30.82	35.73
Laplacian pyramid	20.66	23.73	25.15	27.05	29.81	33.76
GraphBior	21.74	25.82	27.42	29.58	32.46	37.07
Graph laplacian pyramid	21.74	25.28	26.66	28.49	30.76	33.95
GraphBior + OSGL (without vertical edges)	21.29	24.47	25.92	27.97	30.69	35.11
GraphBior + OSGL (with vertical edges)	**21.75**	**26.26**	**27.78**	**29.81**	**32.72**	**37.12**

(continued)

Table 1 (continued)

Fraction of highpass coeffs.	0.00	0.01	0.02	0.04	0.08	0.16
Coins						
9/7 filter	23.23	26.49	28.29	30.73	34.32	40.17
Laplacian pyramid	23.23	26.34	27.94	30.11	33.11	37.66
GraphBior	24.78	28.87	30.54	32.77	35.92	**40.88**
Graph laplacian pyramid	24.78	28.43	29.76	31.55	33.94	37.37
GraphBior + OSGL (without vertical edges)	24.56	27.21	28.70	30.73	33.93	38.63
GraphBior + OSGL (with vertical edges)	**25.15**	**29.11**	**30.75**	**33.09**	**36.22**	40.75

The graph oversampling used the oversampled edge-aware image graph [14]. In this case, on the basis of the edge-aware image bipartite graph, the vertices and the links are added along the edges: (1) diagonal direction regular links are added to the edge-aware rectangular graph and (2) rectangular direction regular links are added to the edge-aware diagonal graph. For instance, the oversampled graphs for the rectangular and diagonal bipartite graphs of Fig. 14a are shown in Fig. 14d, e. The critically sampled graph filter banks are applied to the signals on these graphs using the method described in Sect. 4.1.

Table 1 lists the PSNRs of the reconstructed images, i.e., reconstructions from all lowpass coefficients and some fraction of the highpass coefficients after the three-level decomposition. Since the fraction of highpass coefficients is relative to the size of the original image, the number of the lowpass and highpass coefficients used for the reconstruction is the same for all methods. However, the ratio of the remaining highpass coefficients to the total number of highpass coefficients varies since the Laplacian pyramid and the proposed method are redundant transforms. In spite of this, the oversampled transform performed better than the other methods, including graphBior, in most cases. It can be seen that the vertical edges provide significant gains.

Figures 15 and 16 show images reconstructed from all lowpass coefficients and 3% of the highpass coefficients. The standard CDF 9/7 and Laplacian pyramid for regular signals did not take into account the edge information, and as a result, the reconstructed images were blurred around the edges. Since the graph-based transforms consider the rectangular and/or diagonal edges, they preserve the edges well. We can see that blurring and ringing artifacts around the edges in the reconstructed

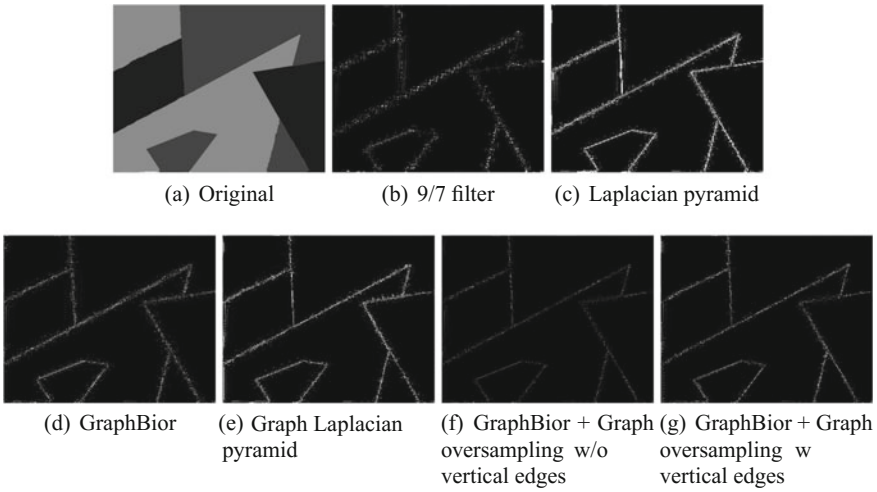


Fig. 15 *Synthetic* images from all lowpass coefficients and 3% of the highpass coefficients after a three-level decomposition. Squared errors between the original and reconstructed images are shown

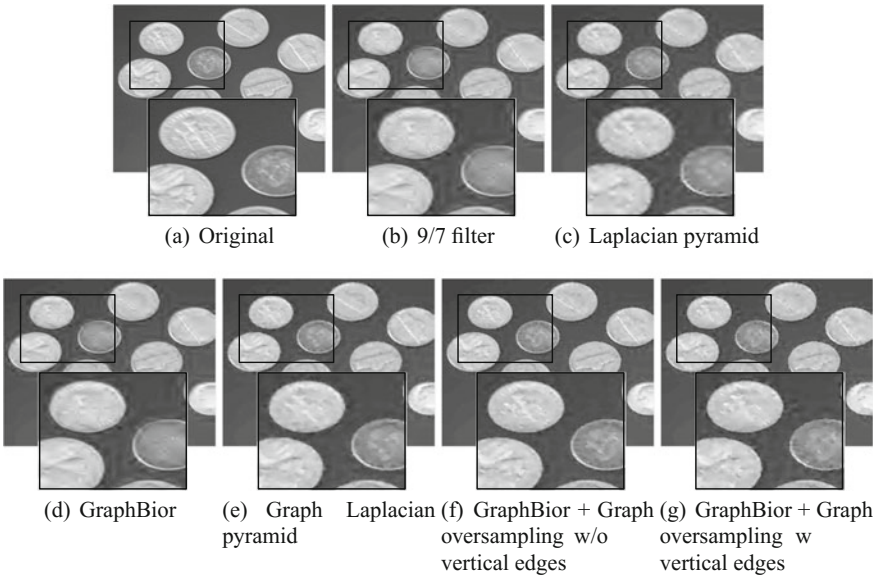


Fig. 16 Reconstructed *Coins* images from all lowpass coefficients and 3% of the highpass coefficients after a three-level decomposition. Zoomed-in parts are specified by black squares

image of the oversampled transform are greatly suppressed compared with other graph-based transforms.

6.2 Denoising of Graph Signals

The detailed experiments of graph signal denoising are shown. Two synthetic input signals on the Minnesota Traffic and Yale Coat of Arms graphs are, respectively, shown in Figs. 17a and 18a. Both signals are localized in the vertex domain, i.e., they have two signal values $f[i] \in \{1, -1\}$ and the regions for $f[i] = 1$ and -1 are concentrated according to the coordinates of the vertices. The input signals are corrupted by additive white Gaussian noise with the standard deviation σ.

For the proposed method, we applied the critically sampled graphBior filter bank (abbreviated as graphBior with OSGLM) [11] or the four-channel oversampled graph filter bank (abbreviated as OSGFB with OSGLM) [22] on oversampled graphs.

We compared the above two methods with the regular one-dimensional wavelet *sym8* with one-level and five-level decompositions, graphBior(6, 6) (graphBior with CSGLM) [11], the Laplacian pyramid for graph signals (GLP) [19], the spectral graph wavelet transform (SGWT) with three scales [8], graphBior with the bipartite double cover (graphBior with BDC), and the four-channel oversampled graph filter bank with the bipartite graph decomposition (OSGFB with CSGLM) [22, 23].

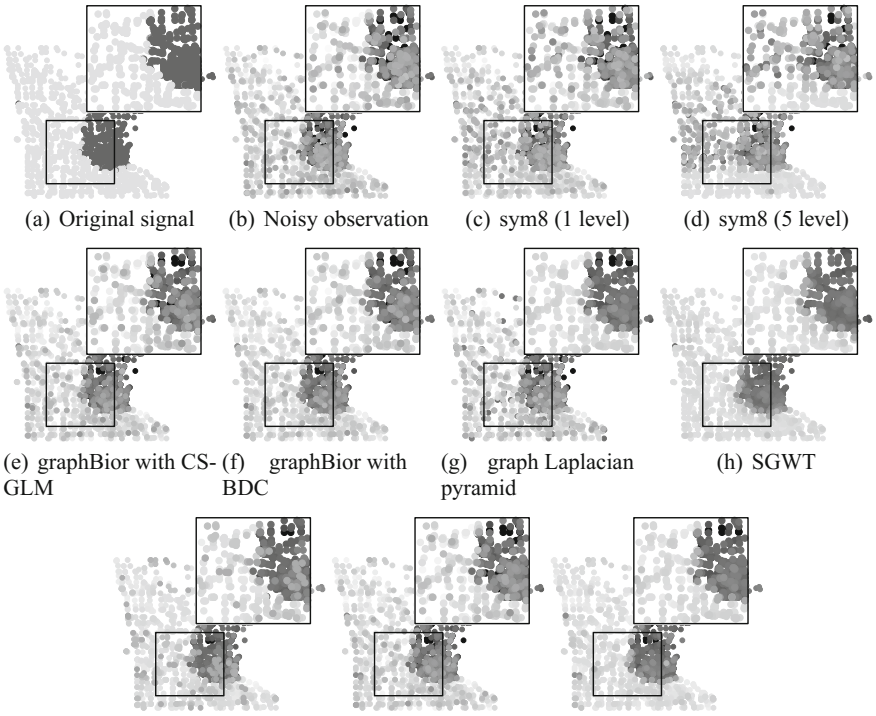


Fig. 17 Denoising results of Minnesota Traffic graph. Zoomed-in parts are specified by black squares

Since *sym8* treated the signal as a vector, it did not take into account the structure of the signals. For a fair comparison, the graph Laplacian pyramid used the same bipartite graphs and downsampling operation as those of graphBior for the lowpass channel.

All of the graph-based methods performed one-level transforms. That is, graphBior, the oversampled graph filter bank and the graph Laplacian pyramid performed two-dimensional transforms by using two subgraphs, whereas the proposed methods performed one-dimensional transforms by using the oversampled bipartite graph. The lowest frequency subband was kept, and the other high-frequency subbands were hard-thresholded with the threshold $T = 3\sigma$.

Table 2 compares the SNRs after denoising. The graph-based transforms outperformed the regular wavelet transforms. OSGFB with OSGLM shows better performance than other graph-based transforms in most cases. It was especially superior to OSGFB with CSGLM and SGWT on the Minnesota Traffic graph, in spite of it having less redundancy. In comparison with the methods using graphBior filters, graphBior with BDC and graphBior with OSGLM have significantly better SNR.

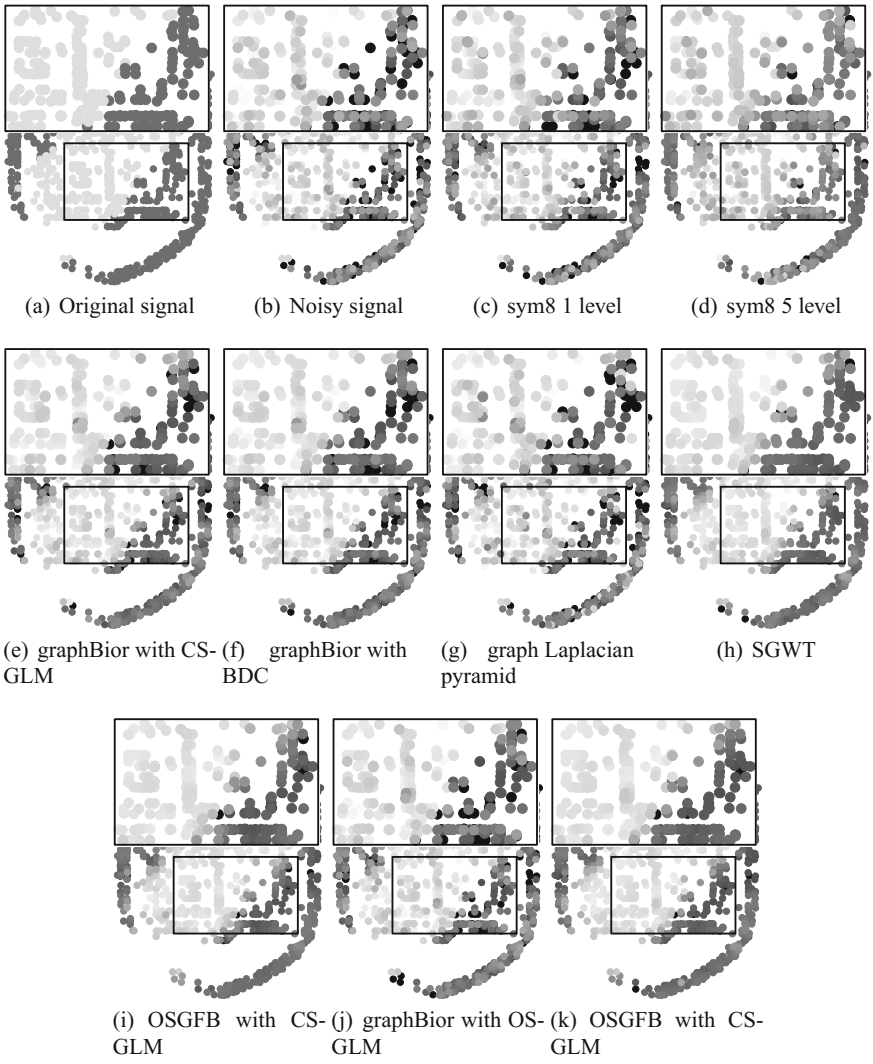


Fig. 18 Denoised results of the Yale Coat of Arms graph. Zoomed-in parts are specified by black squares

Moreover, graphBior with OSGLM had similar levels of performance as graphBior with BDC, especially for the strong noise case, despite that its redundancy is less than graphBior with BDC.

Figures 17 and 18 show the denoised signals of the Minnesota Traffic graph and the Yale Coat of Arms graph for $\sigma = 1/2$, respectively. Since the regular signal processing did not take into account the structure of the signals, the signals denoised by *sym8* were still noisy. We can see that OSGFB with OSGLM performed better than the other transforms.

Table 2 Denoising Results: SNR (dB)

Minnesota traffic graph							
σ	1/32	1/16	1/8	1/4	1/2	1	Redundancy
Noisy	30.15	24.08	18.06	12.02	5.99	−0.02	–
Sym8 (1 level)	30.17	24.25	18.65	11.94	6.23	1.59	1.00
Sym8 (5 levels)	30.22	24.07	17.99	11.07	5.76	3.13	1.00
Graphior + CSGL	31.44	25.61	19.97	14.19	8.50	2.63	1.00
GLP	31.39	25.68	20.02	14.24	8.51	2.61	2.05
SGWT	33.35	27.76	22.08	15.05	10.33	**8.82**	4.00
OSGFB + CSGL	34.75	28.78	21.84	15.26	10.29	4.24	4.00
GraphBior + BDC	32.54	26.75	20.81	14.79	8.92	3.03	2.00
GraphBior + OSGL	32.46	26.76	20.88	14.94	9.00	3.11	1.37
OSGFB + OSGL	**35.08**	**29.34**	**23.17**	**17.63**	**12.31**	7.04	2.74
Yale coat of arms							
σ	1/32	1/16	1/8	1/4	1/2	1	Redundancy
Noisy	30.15	24.10	18.08	12.01	6.04	0.00	–
Sym8 (1 level)	29.64	23.67	18.02	11.36	6.19	1.82	1.00
Sym8 (5 levels)	29.41	23.33	17.04	10.19	4.98	2.21	1.00
GraphBior + CSGL	29.85	24.24	18.75	13.16	8.66	3.94	1.00
GLP	30.10	24.71	19.21	13.58	8.80	4.00	1.79
SGWT	29.40	23.70	18.24	12.95	8.85	6.47	4.00
OSGFB + CSGL	30.11	24.66	19.05	13.98	10.25	7.45	4.00
GraphBior + BDC	31.55	25.43	20.01	14.39	9.23	3.88	2.00
GraphBior + OSGL	31.33	25.32	19.90	14.29	9.23	4.11	1.78
OSGFB + OSGL	**31.77**	**26.80**	**21.22**	**15.21**	**10.49**	**7.48**	3.56

7 Conclusion

This chapter introduced the oversampled graph transforms. By using the oversampled graph Laplacian, any graphs can be converted into an oversampled bipartite graphs. It has a close relationship with bipartite double cover. The perfect reconstruction condition and the design method of the oversampled graph filter banks were also shown that can be used arbitrary graph filters as a component. In numerical experiments, it has been shown that the oversampled graph transforms can have a good trade-off between the performance and redundancy. Since there are several new approaches in graph transforms and sampling [1, 9, 15, 24, 27], new methods to design oversampled graph transforms will be investigated in the future.

Acknowledgements This work was supported in part by JST PRESTO Grant Number JPMJPR1656 and JSPS KAKENHI Grant Number JP16H04362.

References

1. A. Anis, A. Ortega, Critical sampling for wavelet filterbanks on arbitrary graphs. Proc. Int. Conf. Acoust. Speech, Signal Process, 3889–3893 (2017)
2. R.A. Brualdi, F. Harary, Z. Miller, Bigraphs versus digraphs via matrices. J. Graph Theory **4**(1), 51–73 (1980)
3. P. Burt, E. Adelson, The Laplacian pyramid as a compact image code. IEEE Trans. Commun. **31**(4), 532–540 (1983)
4. G. Cheung, E. Magli, Y. Tanaka, M. Ng, Graph spectral image processing. Proc. IEEE **106**(5), 907–930 (2018)
5. A. Cohen, I. Daubechies, J.C. Feauveau, Biorthogonal bases of compactly supported wavelets. Commun. Pure Appl. Math. **45**(5), 485–560 (1992)
6. M.N. Do, M. Vetterli, Framing pyramids. IEEE Trans. Signal Process. **51**(9), 2329–2342 (2003)
7. A.L. Dulmage, N.S. Mendelsohn, Coverings of bipartite graphs. Canadian J. Math. **10**(4), 516–534 (1958)
8. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011). http://wiki.epfl.ch/sgwt
9. Y. Jin, D.I. Shuman, An M-channel critically sampled filter bank for graph signals. Proc. IEEE Int. Conf. Acoust. Speech, Signal Process, 3909–3913 (2017)
10. S.K. Narang, Y.H. Chao, A. Ortega, Graph-wavelet filterbanks for edge-aware image processing, in *Proceedings of SSP'12* (2012), pp. 141–144
11. S.K. Narang, A. Ortega, Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs. IEEE Trans. Signal Process. **61**(19), 4673–4685 (2013). http://biron.usc.edu/wiki/index.php/Graph_Filterbanks
12. S.K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data. IEEE Trans. Signal Process. **60**(6), 2786–2799 (2012). http://biron.usc.edu/wiki/index.php/Graph_Filterbanks
13. H.Q. Nguyen, M.N. Do, Downsampling of signals on graphs via maximum spanning trees. IEEE Trans. Signal Process. **63**(1), 182–191 (2015)
14. A. Sakiyama, Y. Tanaka, Edge-aware image graph expansion methods for oversampled graph Laplacian matrix, in *Proceedings of ICIP'14* (2014), pp. 2958–2962
15. A. Sakiyama, K. Watanabe, Y. Tanaka, A. Ortega, Two-channel critically-sampled graph wavelets with spectral domain sampling (2018). arXiv:1804.08811

16. A. Sakiyama, Y. Tanaka, Oversampled graph Laplacian matrix for graph filter banks. IEEE Trans. Signal Process. **62**(24), 6425–6437 (2014)
17. A. Sakiyama, K. Watanabe, Y. Tanaka, Spectral graph wavelets and filter banks with low approximation error. IEEE Trans. Signal Inf. Process. Netw. **2**(3), 230–245 (2016)
18. E. Sampathkumar, On tensor product graphs. J. Aust. Math. Soc. **20**(3), 268–273 (1975)
19. D.I. Shuman, M.J. Faraji, P. Vandergheynst, A framework for multiscale transforms on graphs (2013). arXiv:1308.4942
20. D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames. IEEE Trans. Signal Process. **63**(16), 4223–4235 (2015). http://documents.epfl.ch/users/s/sh/shuman/www/publications.html
21. D.I. Shuman, M.J. Faraji, P. Vandergheynst, A multiscale pyramid transform for graph signals. IEEE Trans. Signal Process. **64**(8), 2119–2134 (2016)
22. Y. Tanaka, A. Sakiyama, *M*-channel oversampled graph filter banks. IEEE Trans. Signal Process. **62**(14), 3578–3590 (2014)
23. Y. Tanaka, A. Sakiyama, *M*-channel oversampled perfect reconstruction filter banks for graph signals, in *Proceedings of ICASSP'14* (2014), pp. 2604–2608
24. Y. Tanaka, Spectral domain sampling of graph signals. IEEE Trans. Signal Process. **66**(14), 3752–3767 (2018)
25. D.B.H. Tay, Y. Tanaka, A. Sakiyama, Near orthogonal oversampled graph filter banks. IEEE Signal Process. Lett. **23**(2), 277–281 (2015)
26. D.B.H. Tay, Y. Tanaka, A. Sakiyama, Almost tight spectral graph wavelets with polynomial filters. IEEE J. Sel. Topics Signal Process. **11**(6), 812–824 (2017)
27. O. Teke, P.P. Vaidyanathan, Extending classical multirate signal processing theory to graphs–Part II: *M*-channel filter banks. IEEE Trans. Signal Process. **65**(2), 423–437 (2016)
28. N. Tremblay, P. Borgnat, Subgraph-based filterbanks for graph signals. IEEE Trans. Signal Process. **64**(15), 3827–3840 (2016)

Local-Set-Based Graph Signal Sampling and Reconstruction



Yuantao Gu and Xiaohan Wang

Abstract For a graph signal in the low-frequency subspace, the missing data can be reconstructed through the sampled data by exploiting the smoothness of the graph signal. In this chapter, the concepts of local set and centerless local set are introduced and several iterative methods are presented to reconstruct bandlimited graph signal from decimated data or measured signal. These algorithms are built on frame theory and the concepts of (centerless) local sets, based on which several frames and contraction operators are provided. We then prove that the reconstruction methods converge to the original signal under certain conditions and demonstrate the new methods lead to a significantly faster convergence compared with the baseline method. Furthermore, the correspondence between graph signal sampling and time-domain irregular sampling is analyzed comprehensively, which may be helpful to future works on graph signals. Numerical experimental results demonstrate the effectiveness of the reconstruction methods in various sampling geometries, imprecise priori knowledge of cutoff frequency, and noisy scenarios.

1 Introduction

1.1 Background, Motivation, and Organization

In recent years, the increasing demands for signal and information processing in irregular domains have resulted in an emerging field of signal processing on graphs [1, 2]. Bringing a new perspective for analyzing data associated with graphs, graph signal

Y. Gu (✉)
Department of Electronic Engineering, Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China
e-mail: gyt@tsinghua.edu.cn

X. Wang
Huawei Corp., Shanghai, China
e-mail: wangxiaohan3@huawei.com

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*, Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_7

processing has found potential applications in sensor networks, image processing, semi-supervised learning, and recommendation systems.

Smooth signals or approximately smooth signals over graph are common in practical applications [2–5], especially for those cases in which the graph topologies are constructed to enforce the smoothness property of signals [6]. It is a natural problem to reconstruct smooth signals from partial observations on a graph in practical applications [2, 3]. The commonly used observation model is to decimate some entries on only a part of the vertices, i.e. samples of the graph signal.

In addition to the traditional scheme of decimation, the graph signal may be sampled by using local measurement. For example, in a scenario of environment monitoring by wireless sensor networks, especially, a sensor network with the hierarchical architecture that is partitioned into multiple clusters. In each cluster, there is a node acting as the head and gathering data from all sensors inside the cluster. The collected data within a cluster are aggregated by the cluster head, which plays the role as a local measurement and can be naturally obtained. Retrieving the raw data of all the nodes using the measured data from all the clusters can be modeled as a problem of smooth graph signal reconstruction from local measurements.

In this chapter, we will introduce some efficient methods to solve the problem of reconstructing a bandlimited graph signal from known samples. The smooth signal is supposed to be within a low-frequency subspace. Several iterative methods are demonstrated to recover the missing entries from known decimations or measurements.

In order to improve the convergence rate of bandlimited graph signal reconstruction, iterative weighting reconstruction (IWR) and iterative propagating reconstruction (IPR) are introduced based on a new concept of local set [7]. As the foundation of reconstruction methods, several local-set-based frames and contraction operators are introduced. Both IWR and IPR are theoretically proved to uniquely reconstruct the original signal under certain conditions. Compared with existing methods, the condition of the new reconstruction methods is easy to determine by local parameters. The correspondence between graph signal sampling and time-domain irregular sampling is analyzed comprehensively, which will be helpful to future works on graph signals. Experiments show that IWR and IPR converge significantly faster than available methods. Besides, experiments on several topics including sampling geometry and robustness are conducted.

Based on generalizing the graph signal sampling scheme from *decimation* to *measurement*, we then introduce a new algorithm named iterative local measurement reconstruction (ILMR) to reconstruct the original signal from limited measurements [8]. It is proved that if certain conditions are satisfied the bandlimited signal can always be exactly reconstructed from its local measurements. Moreover, we demonstrate that the traditional decimation scheme, which samples by vertex, along with its corresponding reconstruction algorithm is a special case of this work. Based on the performance analysis of ILMR, we find that the local measurement scheme is more robust than decimation in noisy scenarios. As a consequence, the optimal local weights in different noisy environments are discussed. The proposed sampling scheme has several advantages. First, it will benefit in the situation where

local measurements are easier to obtain than the samples of specific vertices. Second, the proposed local measurement scheme is more robust against noise.

The rest of this chapter is organized as follows. In the later part of this section, some basic concepts on graph signal processing and frame theory are introduced. In Sect. 2, the ideas of local set and centerless local set are presented and the traditional sampling scheme is generalized to local measurement. In Sect. 3, some important definitions are introduced and local-set-based frames are proved. In Sect. 4, two local-set-based reconstruction methods IWR and IPR are proposed and their convergence behavior is analyzed, respectively. In Sect. 5, the reconstruction algorithm ILMR is proposed and its convergence is proved. Section 6 shows the relationship between graph signal sampling and time-domain irregular sampling and Sect. 7 presents some numerical experiments. The chapter is concluded in Sect. 8.

1.2 Basic Concepts

1.2.1 Graph Laplacian and Graph Fourier Analysis

An undirected graph is denoted as $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes a set of N vertices and $\mathcal{E}$ denotes the edge set. If one real number is associated with each vertex, these numbers of all the vertices are collectively referred as a graph signal. A graph signal can also be regarded as a mapping $f : \mathcal{V} \to \mathbb{R}$.

The graph Laplacian is extensively exploited in spectral graph theory [9] and signal processing on graphs [1]. For an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, its Laplacian is

$$\mathbf{L} = \mathbf{D} - \mathbf{A},$$

where $\mathbf{A}$ is the adjacency matrix of the graph and $\mathbf{D}$ is a diagonal degree matrix with the diagonal elements as the degrees of corresponding vertices.

The Laplacian is a real symmetric matrix, and all the eigenvalues are nonnegative. Supposing $\{\lambda_k\}$ are the eigenvalues, and $\{\mathbf{u}_k\}$ are the corresponding eigenvectors, the graph Fourier transform is defined as the expansion of a graph signal $\mathbf{f}$ in terms of $\{\mathbf{u}_k\}$, as

$$\hat{f}(k) = \langle \mathbf{f}, \mathbf{u}_k \rangle = \sum_{i=1}^{N} f(i) u_k(i),$$

where $f(i)$ denotes the entry of $\mathbf{f}$ associated with vertex i. Similar with classical Fourier analysis, eigenvalues $\{\lambda_k\}$ are regarded as frequencies of the graph, and $\hat{f}(k)$ is regarded as the frequency component corresponding to λ_k. The frequency components associated with smaller eigenvalues can be called low-frequency part, and those associated with larger eigenvalues form the high-frequency part.

1.2.2 Bandlimited Graph Signal Sampling and Reconstruction

For a graph signal $\mathbf{f} \in \mathbb{R}^N$ on a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, $\mathbf{f}$ is called ω-bandlimited if the spectral support of $\mathbf{f}$ is within $[0, \omega]$. That is, the frequency components corresponding to eigenvalues larger than ω are all zero. The subspace of ω-bandlimited signals on graph $\mathcal{G}$ is a Hilbert space called Paley-Wiener space, denoted as $PW_\omega(\mathcal{G})$ [10].

We consider the sampling and reconstruction of bandlimited signals on undirected and unweighted graphs. Suppose that for a bandlimited graph signal $\mathbf{f} \in PW_\omega(\mathcal{G})$, only $\{f(u)\}_{u\in\mathcal{S}}$ on the sampling set $\mathcal{S} \subseteq \mathcal{V}$ are known, the problem of graph signal reconstruction from decimation is to obtain the original signal $\mathbf{f}$ from the sampled data. The general case of reconstruction from measurement will be explained later.

There are many useful theoretical results on the problem of bandlimited graph signal sampling and reconstruction.

Definition 1 (*uniqueness set*, [10]) A set of vertices $\mathcal{S} \subseteq \mathcal{V}(\mathcal{G})$ is a *uniqueness set* for space $PW_\omega(\mathcal{G})$ if it holds for all $\mathbf{f}, \mathbf{g} \in PW_\omega(\mathcal{G})$ that $f(u) = g(u), \forall u \in \mathcal{S}$ implies $\mathbf{f} = \mathbf{g}$.

According to this definition, any $\mathbf{f} \in PW_\omega(\mathcal{G})$ could be uniquely determined by its entries on a uniqueness set $\mathcal{S}$. As a consequence, $\mathbf{f}$ may be exactly recovered if the sampling set is a uniqueness set. Readers are suggested to refer to [3, 10, 11] for more details on uniqueness set.

A method called iterative least square reconstruction (ILSR) is proposed to reconstruct bandlimited graph signals in [12] as the following theorem.

Theorem 1 ([12]) *If the sampling set $\mathcal{S}$ is a uniqueness set for $PW_\omega(\mathcal{G})$, then the original signal $\mathbf{f}$ can be reconstructed using the sampled data $\{f(u)\}_{u\in\mathcal{S}}$ by ILSR method,*

$$\mathbf{f}^{(k+1)} = \mathcal{P}_\omega(\mathbf{f}^{(k)} + \mathbf{J}^{\mathrm{T}}\mathbf{J}(\mathbf{f}_{du} - \mathbf{f}^{(k)})), \tag{1}$$

where $\mathcal{P}_\omega$ is the projection operator onto $PW_\omega(\mathcal{G})$, and $\mathbf{J}$ denotes the downsampling operator and $\mathbf{f}_{du}$ is the downsampled signal.

ILSR is derived from the method of projection onto convex sets (POCS). Its convergence is proved using the fixed point theorem of contraction mapping.

1.2.3 Frame Theory and Signal Reconstruction

The problem of signal sampling and reconstruction is closely related to frame theory.

Definition 2 (*frame and frame bound*) A family of elements $\{\mathbf{f}_i\}_{i\in\mathcal{I}}$ is a *frame* for a Hilbert space $\mathcal{H}$, if there exist constants $0 < A \leq B$ such that

$$A\|\mathbf{f}\|^2 \leq \sum_{i\in\mathcal{I}} |\langle \mathbf{f}, \mathbf{f}_i\rangle|^2 \leq B\|\mathbf{f}\|^2, \quad \forall \mathbf{f} \in \mathcal{H},$$

where A and B are called *frame bounds*.

Definition 3 (*frame operator*) For a frame $\{\mathbf{f}_i\}_{i\in\mathcal{I}}$, *frame operator* $\mathbf{S}:\mathcal{H}\to\mathcal{H}$ is defined as

$$\mathbf{Sf}=\sum_{i\in\mathcal{I}}\langle\mathbf{f},\mathbf{f}_i\rangle\mathbf{f}_i.$$

One may readily read that $A\mathbf{I}\preceq\mathbf{S}\preceq B\mathbf{I}$ for $\mathcal{H}$, where $\mathbf{I}$ denotes the identity operator and $A\mathbf{I}\preceq\mathbf{S}$ means that $\mathbf{S}-A\mathbf{I}$ is positive semidefinite. Consequently, $\mathbf{S}$ is always invertible and its inverse could be expanded into series in some special cases. For instance, one has

$$\mathbf{f}=\mathbf{S}^{-1}\mathbf{Sf}=\mu\sum_{j=0}^{\infty}(\mathbf{I}-\mu\mathbf{S})^j\mathbf{Sf},$$

where μ is a scalar satisfying $\|\mathbf{I}-\mu\mathbf{S}\|<1$. This inspires that $\mathbf{f}$ could be iteratively reconstructed from any initial point $\mathbf{f}^{(0)}$ by

$$\begin{aligned}\mathbf{f}^{(k+1)}&=\mu\mathbf{Sf}+(\mathbf{I}-\mu\mathbf{S})\mathbf{f}^{(k)}\\&=\mathbf{f}^{(k)}+\mu\mathbf{S}(\mathbf{f}-\mathbf{f}^{(k)}),\end{aligned}\tag{2}$$

with the error bound satisfying

$$\|\mathbf{f}^{(k)}-\mathbf{f}\|\le\|\mathbf{I}-\mu\mathbf{S}\|^k\|\mathbf{f}^{(0)}-\mathbf{f}\|.$$

Obviously, recursion (2) cannot be entitled *reconstruction* because the original signal to be recovered is involved in the iteration. However, it provides a prototype for practical methods, which will be discussed in Sect. 4.

The parameter μ, which could be deemed as a step-size, determines the convergence rate. If one chooses $\mu=1/B$, then $\|\mathbf{I}-\mu\mathbf{S}\|\le 1-A/B<1$, and the error bound of iteration (2) will shrink with the exponential of $(1-A/B)$. A better choice is $\mu=2/(A+B)$, then $\|\mathbf{I}-\mu\mathbf{S}\|\le(B-A)/(B+A)$, which leads to a faster convergence rate [13]. For more information on frames, the reader are recommended to refer to [14].

The following theorem demonstrates that a set of graph signals related to a uniqueness set becomes a frame for $PW_\omega(\mathcal{G})$, which is a quite important foundation of this chapter.

Theorem 2 ([10]) *If the sampling set $\mathcal{S}$ is a uniqueness set for $PW_\omega(\mathcal{G})$, then $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathcal{S}}$ is a frame for $PW_\omega(\mathcal{G})$, where $\mathcal{P}_\omega(\cdot)$ is the projection operator onto $PW_\omega(\mathcal{G})$, and $\boldsymbol{\delta}_u$ is a δ-function whose entries satisfying*

$$\delta_u(v)=\begin{cases}1, & v=u;\\ 0, & v\ne u.\end{cases}$$

2 Local Sets and Generalized Sampling Scheme

2.1 Local Sets and Centerless Local Sets

The concepts of *local sets* and *centerless local sets* for graph signal processing are first proposed in [7, 8], respectively.

Definition 4 (*local sets*, [7]) For a sampling set $\mathcal{S}$ on graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, assume that $\mathcal{V}$ is divided into disjoint local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ associated with the sampled vertices. For each $u \in \mathcal{S}$, denote the subgraph of $\mathcal{G}$ restricted to $\mathcal{N}(u)$ by $\mathcal{G}_{\mathcal{N}(u)}$, which is composed of vertices in $\mathcal{N}(u)$ and edges between them in $\mathcal{E}$. For each $u \in \mathcal{S}$, its local set satisfies $\mathcal{N}(u) \ni u$, and the subgraph $\mathcal{G}_{\mathcal{N}(u)}$ is connected. Besides, $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ should satisfy

$$\mathcal{N}(u) \cap \mathcal{N}(v) = \emptyset, \quad \forall u, v \in \mathcal{S} \text{ and } u \neq v,$$

and

$$\bigcup_{u\in\mathcal{S}} \mathcal{N}(u) = \mathcal{V}.$$

Definition 5 (*centerless local sets*, [8]) For a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, assume that disjoint local sets $\{\mathcal{N}_i\}_{i\in\mathcal{I}}$ is a partition of $\mathcal{V}$, where $\mathcal{I}$ denotes the index set of divisions. Each subgraph $\mathcal{G}_{\mathcal{N}_i}$, which denotes the subgraph of $\mathcal{G}$ restricted to $\mathcal{N}_i$, is connected. Besides, $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ should satisfy

$$\mathcal{N}_i \cap \mathcal{N}_j = \emptyset, \quad \forall i, j \in \mathcal{I} \text{ and } i \neq j,$$

and

$$\bigcup_{i\in\mathcal{I}} \mathcal{N}_i = \mathcal{V}.$$

According to their definitions, both the *local sets* and the *centerless local sets* are disjoint partitions of a connected graph. The difference is that the local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ are defined with respected to a sampling set $\mathcal{S}$, while the centerless local sets $\{\mathcal{N}_i\}_{i\in\mathcal{I}}$ is no more than a partition. Please refer to Fig. 1 for visualization.

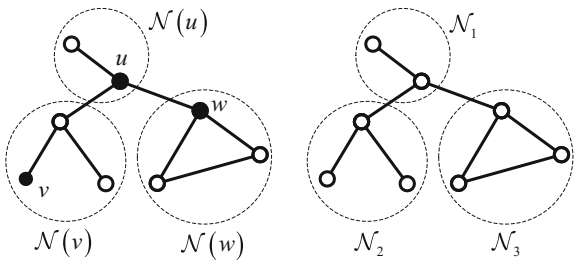


Fig. 1 Illustrations of an example graph, one of the divisions of local sets (left), and that of centerless local sets (right)

Remark 1 One should notice that the local sets and the centerless local sets play different roles in graph signal sampling and reconstruction. In the traditional decimation scheme, the local sets are designed for specific reconstruction algorithms and have no effect in the sampling process. The theoretical foundation and the local-set-based reconstruction algorithms will be introduced in Sect. 4, respectively. On the contrary, the centerless local sets are elaborated for a new type of generalized sampling, or local measurement, and determine the performance of reconstruction, which will be presented in Sect. 5.

In order to describe the property of (centerless) local sets, we introduce several measures, which are useful in the following analysis.

Definition 6 (*maximal size*, [7]) The *maximal size* of local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ is defined as

$$N_{\max} = \max_{u\in\mathcal{S}} |\mathcal{N}(u)|,$$

where $|\cdot|$ denotes cardinality.

Definition 7 (*maximal multiple number*, [7]) Denote

$$\mathcal{T}(u) = \mathrm{SPT}(\mathcal{G}_{\mathcal{N}(u)})$$

as the shortest-path tree of $\mathcal{G}_{\mathcal{N}(u)}$ rooted at u. For v connected to u in $\mathcal{T}(u)$, $\mathcal{T}_u(v)$ is the subtree which v belongs to when u and its associated edges are removed from $\mathcal{T}(u)$. The *maximal multiple number* of $\mathcal{N}(u)$ is defined as

$$K(u) = \max_{(u,v)\in\mathcal{E}(\mathcal{T}(u))} |\mathcal{T}_u(v)|,$$

where $\mathcal{E}(\mathcal{T}(u))$ is the edge set of graph $\mathcal{T}(u)$.

Remark 2 By the definition of $K(u)$, it is ready to check that

$$K(u) \le |\mathcal{N}(u)| - d_{\mathcal{N}(u)}(u) \le |\mathcal{N}(u)| - 1, \tag{3}$$

where $d_{\mathcal{N}(u)}(u)$ is the degree of u in the subgraph $\mathcal{G}_{\mathcal{N}(u)}$. For simplicity, one may introduce an approximation for easy calculation of $K(u)$ by

$$\tilde{K}(u) = |\mathcal{N}(u)| - d_{\mathcal{N}(u)}(u). \tag{4}$$

Definition 8 (*radius*, [7]) The *radius* of $\mathcal{N}(u)$ is the maximal distance from u to any other vertex in $\mathcal{G}_{\mathcal{N}(u)}$, which is denoted as

$$R(u) = \max_{v\in\mathcal{N}(u)} \mathrm{dist}(v, u).$$

Definition 9 (*diameter*, [8]) For a centerless local set $\mathcal{N}_i$, its diameter is defined as the largest distance of two vertices in $\mathcal{G}_{\mathcal{N}_i}$, i.e.,

$$D_i = \max_{u,v\in\mathcal{N}_i} \operatorname{dist}(u, v).$$

2.2 Generalized Sampling: From Decimation to Measurement

We consider a new sampling scheme of measuring by using the concept of centerless local sets. In this scheme, all the vertices in a graph are partitioned into centerless local sets. In each set, there is no specific sampling vertex, but all vertices in this set contribute to produce a measurement. Accordingly, a *local weight* is defined to balance the contribution of all vertices in this set and to obstruct the energy from other parts of the graph.

Definition 10 (*local weight*, [8]) A local weight $\boldsymbol{\varphi}_i \in \mathbb{R}^N$ associated with a centerless local set $\mathcal{N}_i$ satisfies

$$\varphi_i(v) \begin{cases} \geq 0, v \in \mathcal{N}_i \\ = 0, v \notin \mathcal{N}_i \end{cases}$$

and

$$\sum_{v\in\mathcal{N}_i} \varphi_i(v) = 1.$$

We arrive at the definition of *local measurement* by linearly combining the signals in each centerless local set using preassigned local weights.

Definition 11 (*local measurement*, [8]) For given centerless local sets and the associated local weights $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i\in\mathcal{I}}$, a set of local measurements for a graph signal $\mathbf{f}$ is $\{f_{\boldsymbol{\varphi}_i}\}_{i\in\mathcal{I}}$, where

$$f_{\boldsymbol{\varphi}_i} \triangleq \langle \mathbf{f}, \boldsymbol{\varphi}_i \rangle = \sum_{v\in\mathcal{N}_i} f(v)\varphi_i(v).$$

The sampling schemes of decimation and of local measurement are visualized in Fig. 2. Compared with decimation, local measurement can be regarded as a generalized sampling scheme. The local measurement scheme is to obtain linear combinations of the signals in each local set, while the decimation scheme is to obtain the signals on selected vertices in the sampling set $\mathcal{S}$. Both sampling schemes take the inner products of the original signal and specified local weights. Decimation can be regarded as a special case of local measurement, in which only the sampled vertices have weight 1 and other vertices in centerless local sets have weights 0.

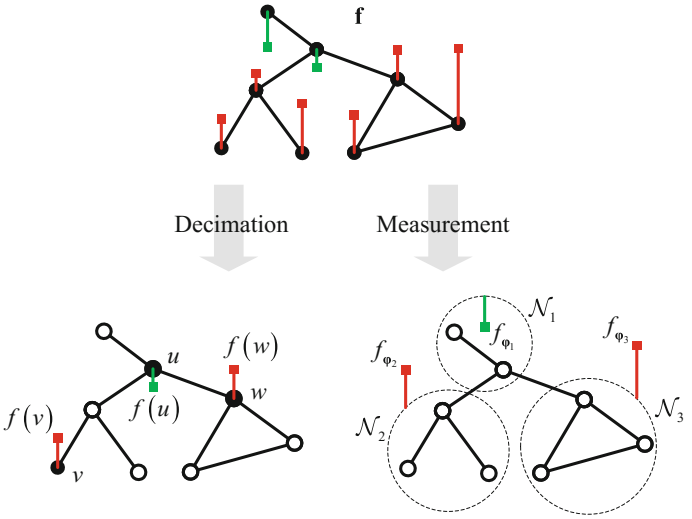


Fig. 2 An illustration of the traditional sampling (decimation) scheme versus the generalized sampling (local measurement) scheme. For each centerless local set, a local measurement is produced by a linear combination of signals associated with vertices within this set

Remark 3 We highlight that the sets, weights, and measurements are *local* rather than *global*, which comes from some natural observations. It is partially because locality and local operations are basic features of graphs and complex networks. Moreover, signal processing on graphs may be dependent on distributed implementation, where local operations are more feasible than global ones.

3 Local-Set-Based Frame and Contraction

Based on frame theory and the introduced local sets, we define two operators named *local propagation* and *local-measurement-based propagation*. Both of them are proved to have contraction property. Then several local-set-based frames are introduced, as the theoretical foundation of the local-set-based reconstruction methods in Sect. 4.

3.1 Local Propagation and Contraction

Definition 12 (*local propagation*, [7]) For a given sampling set $\mathcal{S}$ and associated local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ on a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, the local propagation **G** is defined by

$$\mathbf{Gf} = \mathcal{P}_\omega \left(\sum_{u \in \mathcal{S}} f(u) \boldsymbol{\delta}_{\mathcal{N}(u)} \right) \tag{5}$$

$$= \sum_{u \in \mathcal{S}} f(u) \mathcal{P}_\omega \left(\boldsymbol{\delta}_{\mathcal{N}(u)} \right), \tag{6}$$

where $\boldsymbol{\delta}_{\mathcal{N}(u)}$ denotes the δ-function of set $\mathcal{N}(u)$ with entries

$$\delta_{\mathcal{N}(u)}(v) = \begin{cases} 1, & v \in \mathcal{N}(u); \\ 0, & v \notin \mathcal{N}(u). \end{cases}$$

As its name shows, operation $\mathbf{G}$ first propagates the energy locally and evenly to the local set that each sampled vertex belongs to, and then projects the new signal to be ω-bandlimited, please refer to (5). These two steps could be merged into one, by a bandlimited local propagation of $\mathcal{P}_\omega \left(\boldsymbol{\delta}_{\mathcal{N}(u)} \right)$, please refer to (6). Local propagation, which provides a fast solution to adequately fill all unknown entries by sampled data, makes the local-set-based reconstruction feasible.

As an important theoretical foundation, the following lemma gives the condition that $(\mathbf{I} - \mathbf{G})$ is a contraction mapping.

Lemma 1 ([7]) *For a given set $\mathcal{S}$ and associated local sets $\{\mathcal{N}(u)\}_{u \in \mathcal{S}}$ on a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, $\forall \omega < 1/Q_{\max}^2$, the operator $(\mathbf{I} - \mathbf{G})$ is a contraction mapping for $PW_\omega(\mathcal{G})$, where*

$$Q_{max} = \max_{u \in \mathcal{S}} \sqrt{K(u) R(u)}, \tag{7}$$

where $K(u)$ and $R(u)$ denote the maximal multiple number and the radius of $\mathcal{N}(u)$, respectively.

3.2 Local Measurement-Propagation and Contraction

Definition 13 ([8]) For a given centerless local sets and the associated weights $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i \in \mathcal{I}}$ on a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, an operator $\widehat{\mathbf{G}}$ is defined by

$$\widehat{\mathbf{G}}\mathbf{f} = \mathcal{P}_\omega \left(\sum_{i \in \mathcal{I}} \langle \mathbf{f}, \boldsymbol{\varphi}_i \rangle \boldsymbol{\delta}_{\mathcal{N}_i} \right) \tag{8}$$

$$= \sum_{i \in \mathcal{I}} \langle \mathbf{f}, \boldsymbol{\varphi}_i \rangle \mathcal{P}_\omega(\boldsymbol{\delta}_{\mathcal{N}_i}), \tag{9}$$

where $\boldsymbol{\delta}_{\mathcal{N}_i}$ is defined as

$$\delta_{\mathcal{N}_i}(v) = \begin{cases} 1, & v \in \mathcal{N}_i; \\ 0, & v \notin \mathcal{N}_i. \end{cases} \tag{10}$$

For a graph signal, the proposed operator is to calculate the local measurement in each centerless local set, then to assign the local measurement to all the vertices in that set, and finally to filter out the component beyond the bandwidth, i.e., (8). Equivalently, it denotes a linear combination of all low-frequency parts of $\{\boldsymbol{\delta}_{\mathcal{N}_i}\}_{i\in\mathcal{I}}$, with the combination coefficients as the local measurements of corresponding local sets, i.e., (9).

The following lemma shows that the proposed operator is bounded in $PW_\omega(\mathcal{G})$ under certain conditions.

Lemma 2 ([8]) *For given centerless local sets and the associated weights* $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i\in\mathcal{I}}$, $\forall \mathbf{f} \in PW_\omega(\mathcal{G})$, *the following inequality holds,*

$$\|\mathbf{f} - \widehat{\mathbf{G}}\mathbf{f}\| \leq C_{\max}\sqrt{\omega}\|\mathbf{f}\|,$$

where

$$C_{\max} = \max_{i\in\mathcal{I}} \sqrt{|\mathcal{N}_i| D_i},$$

$|\cdot|$ *denotes cardinality, and* D_i *is defined in Definition 9.*

Lemma 2 shows that the operator $(\mathbf{I} - \widehat{\mathbf{G}})$ is a contraction mapping in $PW_\omega(\mathcal{G})$ if ω is less than $1/C_{\max}^2$.

The process of local propagation and local-measurement-based propagation are illustrated in Fig. 3.

3.3 Weighted Frame and Local Set Frame

Based on frame theory and the definition of local set, we could prove that the weighted lowpass δ-function set is a frame for $PW_\omega(\mathcal{G})$ and estimate its bounds.

Lemma 3 ([7]) *For a given sampling set* $\mathcal{S}$ *and associated local sets* $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ *on a graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$, $\forall\omega < 1/Q_{\max}^2$, $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathcal{S}}$ *is a frame for* $PW_\omega(\mathcal{G})$ *with bounds* $(1-\gamma)^2/N_{max}$ *and* 1, *where* Q_{max} *is defined in* (7) *and*

$$\gamma = Q_{\max}\sqrt{\omega}. \tag{11}$$

Beyond Lemma 3, we further explore the weighted lowpass δ-functions is also a frame for $PW_\omega(\mathcal{G})$ by appropriate weights.

Lemma 4 ([7]) *For a given sampling set* $\mathcal{S}$ *and associated local sets* $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ *on a graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$, $\forall\omega < 1/Q_{\max}^2$, $\{\sqrt{|\mathcal{N}(u)|}\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathcal{S}}$ *is a frame for* $PW_\omega(\mathcal{G})$ *with bounds* $(1-\gamma)^2$ *and* $(1+\gamma)^2$, *where* $Q_{\max}$ *and* γ *are defined in* (7) *and* (11), *respectively.*

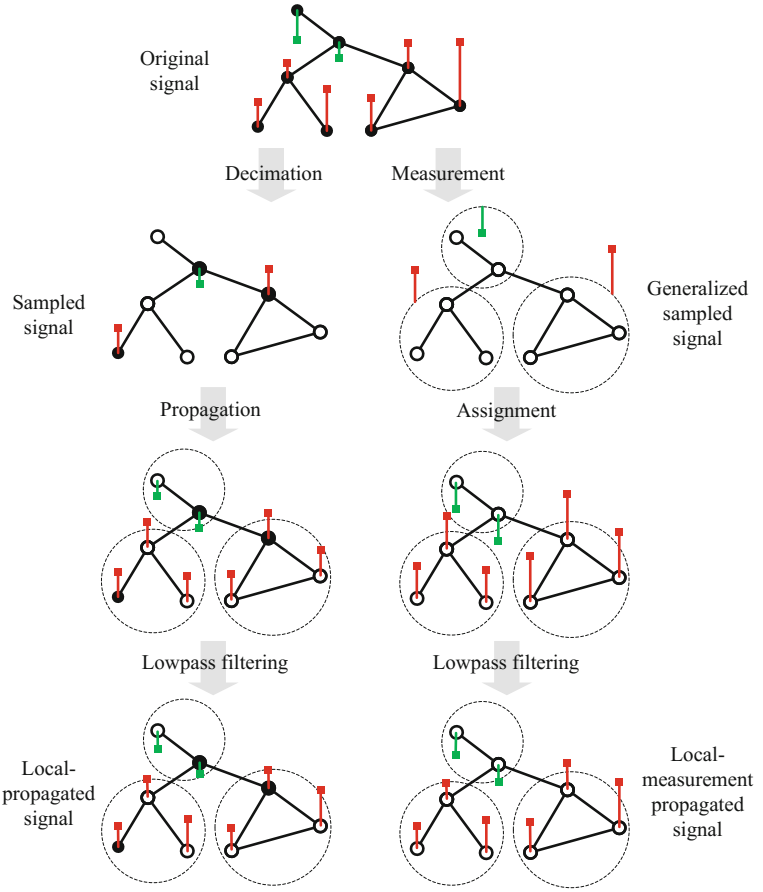


Fig. 3 The illustrations of local propagation and local-measurement-based propagation

Bandlimited graph signals can be iteratively reconstructed using a frame for $PW_\omega(\mathcal{G})$, but the frame bounds play critical roles on the convergence rate. By given appropriate weights to the elements in a frame, a new frame is obtained with a sharper bounds estimation, which may lead to a faster convergence. The related algorithms will be proposed in Sect. 4.

To end up this section, we present a general theoretical result which may inspire further study on frame-theory-based graph signal processing.

Proposition 1 ([7]) *For a given sampling set* $\mathcal{S}$ *and associated local sets* $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ *on a graph* $\mathcal{G}(\mathcal{V},\mathcal{E})$, $\forall\omega < 1/Q_{\max}^2$, $\{\mathcal{P}_\omega(\boldsymbol{\delta}_{\mathcal{N}(u)})\}_{u\in\mathcal{S}}$ *is a frame for* $PW_\omega(\mathcal{G})$ *with bounds* $(1-\gamma)^2$ *and* $N_{\max}$, *where* $Q_{\max}$ *and* γ *are defined in* (7) *and* (11), *respectively.*

Table 1 The frames in space $PW_\omega(\mathcal{G})$, $\forall\omega < 1/Q^2_{\max}$, and their bounds

Frame	Lower bound	Upper bound
$\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathbb{S}}$	$(1-\gamma)^2/N_{\max}$	1
$\{\sqrt{\lvert\mathcal{N}(u)\rvert}\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathbb{S}}$	$(1-\gamma)^2$	$(1+\gamma)^2$
$\{\mathcal{P}_\omega(\boldsymbol{\delta}_{\mathcal{N}(u)})\}_{u\in\mathbb{S}}$	$(1-\gamma)^2$	$N_{\max}$

Proposition 1 implies a strong relationship between frame and sampling reconstruction. In fact local propagation is not a standard frame operator, because two signal sets $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathbb{S}}$ and $\{\mathcal{P}_\omega(\boldsymbol{\delta}_{\mathcal{N}(u)})\}_{u\in\mathbb{S}}$ are involved. However, under the same condition with the contraction of operator $(\mathbf{I}-\mathbf{G})$, both sets can be proved to be frame, and either of them can be used to reconstruct the original signal by the corresponding frame operator. All frames discussed in this section are listed in Table 1.

4 Local-Set-Based Graph Signal Reconstruction

Using frame theory, ILSR is first represented in the frame-based framework. Based on the introduced local sets, two methods Iterative Weighting Reconstruction (IWR) and Iterative Propagating Reconstruction (IPR) are presented in this section. They have been proved of convergence with theoretical analysis.

4.1 Iterative Least Square Reconstruction

In [7], the ILSR algorithm [12] in the form of (1) was presented into frame-based framework for the first time. According to Definition 3 and Lemma 3, frame operator associated with frame $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathbb{S}}$ is

$$\mathbf{Sf} = \sum_{u\in\mathbb{S}} \langle \mathbf{f}, \mathcal{P}_\omega(\boldsymbol{\delta}_u)\rangle \mathcal{P}_\omega(\boldsymbol{\delta}_u). \tag{12}$$

For $\mathbf{f} \in PW_\omega(\mathcal{G})$, one has $\mathcal{P}_\omega(\mathbf{f}) = \mathbf{f}$ and yields

$$\langle \mathbf{f}, \mathcal{P}_\omega(\boldsymbol{\delta}_u)\rangle = \langle \mathcal{P}_\omega(\mathbf{f}), \boldsymbol{\delta}_u\rangle = \langle \mathbf{f}, \boldsymbol{\delta}_u\rangle = f(u).$$

Consequently, frame operator (12) is reduced to

$$\mathbf{Sf} = \sum_{u\in\mathbb{S}} f(u)\mathcal{P}_\omega(\boldsymbol{\delta}_u). \tag{13}$$

Algorithm 1 Iterative Least Square Reconstruction.

1: Input: Graph $\mathcal{G}$, cutoff frequency ω, sampling set $\mathcal{S}$, sampled data $\{f(u)\}_{u\in\mathcal{S}}$;
2: Output: Reconstructed signal $\mathbf{f}^{(k)}$;
3: Initialization:
4: $$\mathbf{f}^{(0)} = \mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}} f(u)\boldsymbol{\delta}_u\right);$$
5: **repeat**
6: $$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}} (f(u) - f^{(k)}(u))\boldsymbol{\delta}_u\right);$$
7: **until** The stop condition is satisfied.

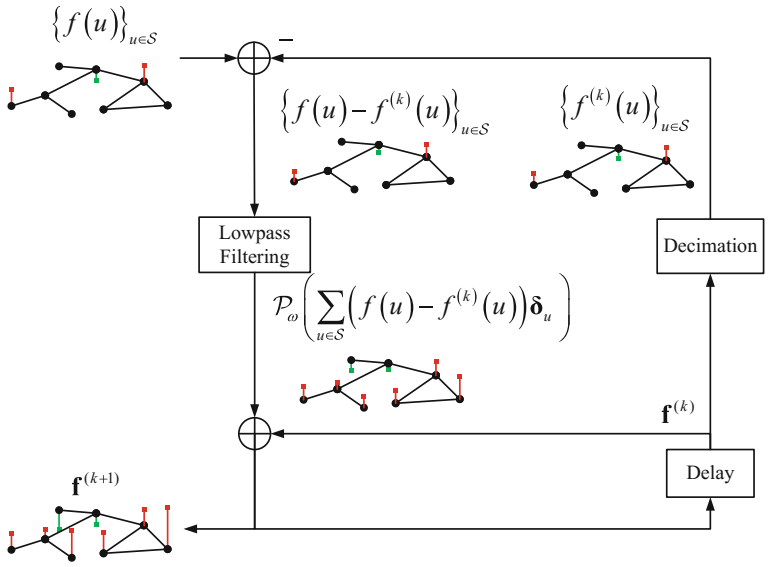


Fig. 4 Illustration of the iterations of ILSR

Utilizing (13) in (2), one may read that the original signal, whose unsampled values are never needed in the iterative reconstruction, could be exactly recovered from its entries on a uniqueness set. The reformulated ILSR method is displayed in Algorithm 1. The iteration process is visualized in Fig. 4.

4.2 Iterative Weighting Reconstruction

Using the weighted frame, an algorithm named iterative weighting reconstruction (IWR) is presented in Proposition 2 and its convergence is proved.

Proposition 2 ([7]) *For a given sampling set $\mathcal{S}$ and associated local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ on a graph $\mathcal{G}(\mathcal{V},\mathcal{E})$, $\forall \mathbf{f} \in PW_\omega(\mathcal{G})$, where $\omega < 1/Q_{\max}^2$, $\mathbf{f}$ can be reconstructed by the sampled data $\{f(u)\}_{u\in\mathcal{S}}$ through the IWR method in Algorithm 2, with the error bound satisfying*

$$\|\mathbf{f}^{(k)} - \mathbf{f}\| \le \left(\frac{2\gamma}{1+\gamma^2}\right)^k \|\mathbf{f}^{(0)} - \mathbf{f}\|,$$

where $Q_{\max}$ and γ are defined in (7) and (11), respectively.

Algorithm 2 Iterative Weighting Reconstruction.

1: Input: Graph $\mathcal{G}$, cutoff frequency ω, sampling set $\mathcal{S}$, neighbor sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$, sampled data $\{f(u)\}_{u\in\mathcal{S}}$;
2: Output: Reconstructed signal $\mathbf{f}^{(k)}$;
3: Initialization:
4: $$\mathbf{f}^{(0)} = \frac{1}{1+\gamma^2}\mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}} |\mathcal{N}(u)| f(u)\boldsymbol{\delta}_u\right);$$
5: **repeat**
6: $$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \frac{1}{1+\gamma^2}\mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}} |\mathcal{N}(u)|(f(u) - f^{(k)}(u))\boldsymbol{\delta}_u\right);$$
7: **until** The stop condition is satisfied.

The iteration process of IWR is visualized in Fig. 5. The idea of IWR is to attach different weights to sampled vertices. The weights for vertex u is larger if its local set $\mathcal{N}(u)$ has more vertices, in other words, the vertex u is more isolated or the region around u has a lower sampling density. On the contrary, if the sampled vertices in a region are very dense, less importance is allocated to them.

Corollary 1 ([7]) *In Lemma 4 and Proposition 2, Q_{max} can be replaced by $\tilde{Q}_{max}$, which is defined as*

$$\tilde{Q}_{max} = \max_{u\in\mathcal{S}} \sqrt{\tilde{K}(u)R(u)}.$$

According to (3), for any $u \in \mathcal{S}$ we have $\tilde{K}(u) \ge K(u)$, and then $\tilde{Q}_{\max} \ge Q_{\max}$. In fact, $K(u)$ is not easy to obtain for each given subgraph $\mathcal{G}_{\mathcal{N}(u)}$. However, $\tilde{K}(u)$ is convenient to get and $\tilde{Q}_{\max}$ is a practical choice, even though the bound is not as accurate.

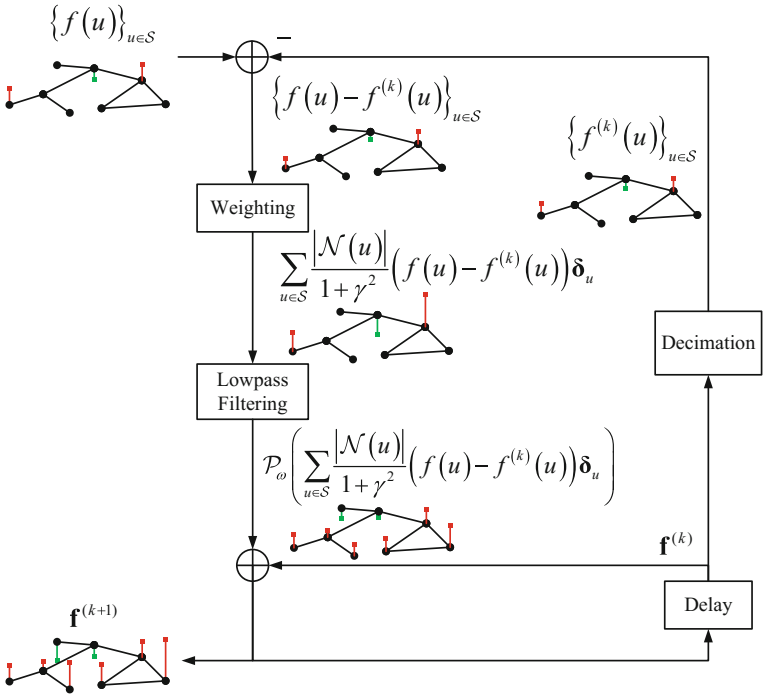


Fig. 5 Illustration of the iterations of IWR

4.3 Iterative Propagating Reconstruction

Iterative propagating reconstruction (IPR) is proposed as the result of the contraction of the local propagation operator.

Proposition 3 ([7]) *For a given sampling set* $\mathcal{S}$ *and associated local sets* $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ *on a graph* $\mathcal{G}(\mathcal{V},\mathcal{E})$, $\forall \mathbf{f}\in PW_\omega(\mathcal{G})$, *where* $\omega < 1/Q_{\max}^2$, $\mathbf{f}$ *can always be reconstructed by its samples* $\{f(u)\}_{u\in\mathcal{S}}$ *through the IPR method in Algorithm 3, with the error bound satisfying*

$$\|\mathbf{f}^{(k)}-\mathbf{f}\| \le \gamma^k \|\mathbf{f}^{(0)}-\mathbf{f}\|,$$

where $Q_{\max}$ *and* γ *are defined in* (7) *and* (11), *respectively.*

Other than basic frame operators in IWR, IPR is based on the contraction of the local propagation operator, in which two frames are involved. Strictly speaking, IPR is not a frame-based method. IPR is closely related to local sets, which is one of the main contributions of this work. The iteration process of IPR is visualized in Fig. 6.

Algorithm 3 Iterative Propagating Reconstruction.

1: Input: Graph $\mathcal{G}$, cutoff frequency ω, sampling set $\mathcal{S}$, neighbor sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$, sampled data $\{f(u)\}_{u\in\mathcal{S}}$;
2: Output: Reconstructed signal $\mathbf{f}^{(k)}$;
3: Initialization:
4: $$\mathbf{f}^{(0)} = \mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}} f(u)\boldsymbol{\delta}_{\mathcal{N}(u)}\right);$$
5: **repeat**
6: $$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \mathcal{P}_\omega\left(\sum_{u\in\mathcal{S}}(f(u) - f^{(k)}(u))\boldsymbol{\delta}_{\mathcal{N}(u)}\right);$$
7: **until** The stop condition is satisfied.

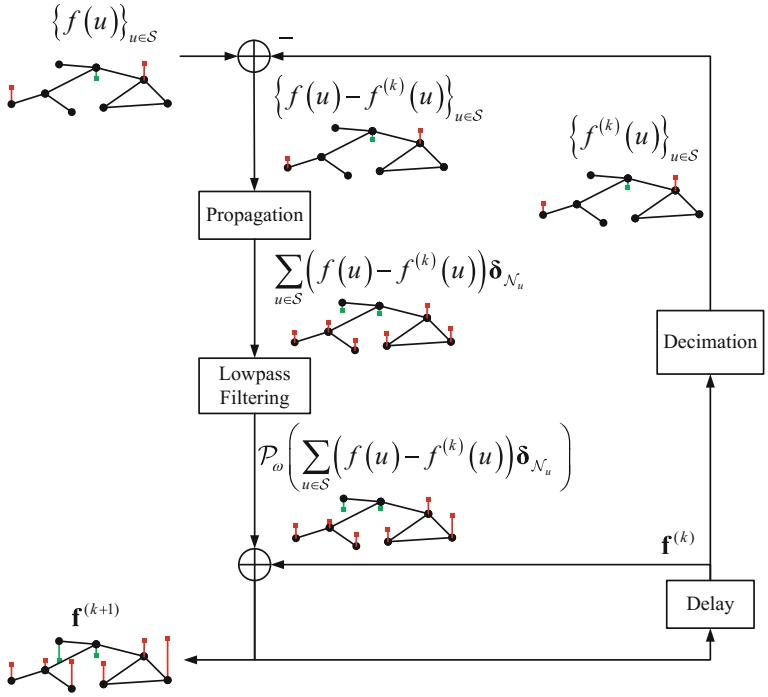


Fig. 6 Illustration of the iterations of IPR

Remark 4 Similar to Proposition 2, $Q_{\max}$ can also be replaced by $\tilde{Q}_{\max}$ in Lemma 2 and Proposition 3, which is more practical to obtain.

Since $\frac{2\gamma}{1+\gamma^2} > \gamma$ when $0 < \gamma < 1$, the theoretical guarantee of IPR decays faster than that of IWR. When γ approaches to 1, the two theoretical guarantees are close to each other.

4.4 Discussions

In this subsection, we will first talk about the determination of cutoff frequency. Considering that both IWR and IPR are based on a division of the graph, i.e., local sets, we will then show two special sampling set and local sets, and discuss the choice of local sets for general cases.

4.4.1 Intuitive Explanation of the Above Algorithms

As illustrated in Figs. 4, 5, and 6, the differences among ILSR, IWR, and IPR lie in the way of their dealing with the residuals at the sampled vertices. In each iteration the sampled residual $\sum_{u\in\mathcal{S}}(f(u)-f^{(k)}(u))\boldsymbol{\delta}_u$ is directly projected onto the ω-bandlimited space $PW_\omega(\mathcal{G})$ in ILSR. In IWR, the sampled residuals are multiplied by weights $|\mathcal{N}(u)|$ and then projected onto the low-frequency space. For IPR, the sampled residuals are copied and assigned to the vertices in the corresponding local sets and then the projection procedure is conducted.

Because of the weighting or propagating procedure, for each step the increment of IWR or IPR is larger than that of ILSR. It may intuitively explain why the proposed two algorithms both converge faster than ILSR. Besides, it is easy to see from Figs. 5 and 6 that the graph signal composed of the propagated residuals seems closer to a low-frequency signal than the weighted residual, which means that the increment of IPR remains more than that of IWR after the projection to the low-frequency subspace. It may explain why IPR converges even faster than IWR.

4.4.2 Cutoff Frequency

According to Propositions 2 and 3, the estimated convergence bounds of IWR and IPR are related to the cutoff frequency and the topology of local sets. Adequately estimating the cutoff frequency of the raw signal may accelerate the convergence of reconstruction. For given $\mathcal{S}$ and $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$, the maximal multiple number and radius are to be calculated. Consequently, $Q_{\max}$ is determined. Therefore, a smaller known ω leads to a smaller γ, then sharper error bounds of convergence are obtained for both IWR and IPR, which may lead to a faster convergence. The choice of local sets also affects the convergence performance, which will be discussed in the following section.

In the local-set-based methods IWR and IPR, the theoretical maximal cutoff frequency, below which the raw signal could be recovered, is much easier to obtain. For given sampling set $\mathcal{S}$ and associated local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$, the maximal multiple number and radius can be obtained locally, then $Q_{\max}$ and the cutoff frequency are easy to be determined.

4.4.3 Special Sampling Set and Local Sets

If the sampling set $\mathcal{S} = \mathcal{V}$, then $|\mathcal{N}(u)| = 1$, $K(u) = 0$, and $Q_{\max} = 0$. According to Propositions 2 and 3, any $\mathbf{f}$ satisfying $\omega < \infty$ can be reconstructed by IWR and IPR, which is a natural result.

Another extreme case is the sampling set $\mathcal{S}$ contains only one vertex and the corresponding local set is all the vertices in the graph. In this case, only constant signals can be reconstructed from the sampled data, which means that only one discrete frequency $\omega = 0$ can satisfy the condition $\omega < 1/Q_{\max}^2$. It is easy to understand because only the signals with the same value for all the vertices can be reconstructed from only a single sample. The analysis above is always true no matter which vertex is chosen as the sampled one. Therefore, the following corollary gives an estimation of the smallest positive eigenvalue of a graph.

Corollary 2 ([7]) *For a graph Laplacian, its smallest positive eigenvalue λ_{min} satisfies*

$$\lambda_{min} \geq \max_{u \in \mathcal{V}(\mathcal{G})} \frac{1}{K(u)R(u)},$$

where $K(u)$ and $R(u)$ are defined in Definitions 7 and 8, respectively, in which the local set contains all the vertices of graph $\mathcal{G}$, i.e., $\mathcal{G}_{\mathcal{N}(u)} = \mathcal{G}$.

4.4.4 On the Evaluation of Local Sets

According to the sufficient condition in Propositions 2 and 3, a sampling set and the associated local sets with a smaller $Q_{\max}$ usually lead to a wider range of bandlimited signal which can be guaranteed to reconstruct. Besides, for a given ω, a smaller $Q_{\max}$ leads to a better error bound of convergence, i.e., a smaller γ (for IPR) or $2\gamma/(1+\gamma^2)$ (for IWR). Therefore, when the graph topology is given, it is necessary to find a proper sampling set $\mathcal{S}$ and the corresponding vertex division $\{\mathcal{N}(u)\}_{u \in \mathcal{S}}$, which makes the quantity $Q_{\max}$ as small as possible.

However, since the sufficient condition we give is rather conservative and not very sharp, minimizing $Q_{\max}$ is only a rough way to obtain a better division of local sets. In other words, there may be some better evaluation of local sets than $Q_{\max}$. Finding the optimal division of local sets is still an open problem and needs more comprehensive study. Therefore we have not focused on how to construct local sets to minimize $Q_{\max}$ in this paper. It may be done better when a sharper sufficient condition is provided, which will be studied in the future work. In this paper, only a special choice of sampling set and the associated local sets with $Q_{\max} = 1$ are presented in the following text.

4.4.5 A Special Case of One-hop Sampling

In this section, we present a special case of *dense* sampling where all entries to be recovered are directly connected to the sampled vertices. One may read that such dense sampling facilitates the local sets partition.

Corollary 3 ([7]) *For a given sampling set* $\mathcal{S}$*, if the local sets* $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$ *satisfies*

$$\max_{v\in\mathcal{N}(u)} dist(u,v) \leq 1, \quad \forall u \in \mathcal{S}, \tag{14}$$

the sufficient condition of recovery in Propositions 2 and 3 can be refined as $\omega < 1$ *and* $\gamma = \sqrt{\omega}$*.*

A greedy method is proposed and described in Algorithm 4, which can produce the one-hop sampling set and the associated local sets at the same time and satisfy the condition of (14). The reason for selecting the vertex with the largest degree and its neighbors is that more vertices can be removed in each step, which may lead to a sampling set with fewer vertices. One may accept that this is a rather economical choice of sampling set when there is no restriction on the number or location of the sampling vertices, because both $K(u)$ and $R(u)$ are small simultaneously.

Algorithm 4 A Greedy Method for a One-hop Sampling Set.

1: Input: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$;
2: Output: One-hop sampling set $\mathcal{S}$, local sets $\{\mathcal{N}(u)\}_{u\in\mathcal{S}}$;
3: Initialization: $\mathcal{S}=\emptyset$;
4: **repeat**
5: Find the largest-degree vertex, $u = \arg\max_{v\in\mathcal{V}} d_{\mathcal{G}}(v)$;
6: Add u into the sampling set, $\mathcal{S}=\mathcal{S}\cup\{u\}$;
7: The one-hop local set $\mathcal{N}(u)=\{u\}\cup\{v\in\mathcal{V}|(u,v)\in\mathcal{E}\}$;
8: Remove the edges, $\mathcal{E}=\mathcal{E}\backslash\{(p,q)|p\in\mathcal{N}(u), q\in\mathcal{V}\}$;
9: Remove the vertices, $\mathcal{V}=\mathcal{V}\backslash\mathcal{N}(u)$ and $\mathcal{G}(\mathcal{V},\mathcal{E})$;
10: **until** $\mathcal{V}=\emptyset$.

5 Local-Measurements-Based Graph Signal Reconstruction

We will show that under certain conditions the original signal $\mathbf{f}$ can be uniquely and exactly reconstructed from the local measurements $\{f_{\boldsymbol{\varphi}_i}\}_{i\in\mathcal{J}}$.

5.1 Iterative Local Measurement Reconstruction

Based on Lemma 2, it is shown in Proposition 4 that the original signal can be reconstructed from its local measurements.

Proposition 4 ([8]) *Given centerless local sets and the associated weights* $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i\in\mathcal{J}}$, $\forall \mathbf{f} \in PW_\omega(\mathcal{G})$, *where* ω *is less than* $1/C_{\max}^2$, $\mathbf{f}$ *can be reconstructed from its local measurements* $\{f_{\boldsymbol{\varphi}_i}\}_{i\in\mathcal{J}}$ *through an iterative local measurement reconstruction (ILMR) algorithm in Algorithm 5, with the error at the* k*th iteration satisfying*

$$\|\mathbf{f}^{(k)} - \mathbf{f}\| \leq \gamma^k \|\mathbf{f}^{(0)} - \mathbf{f}\|,$$

where

$$\gamma = C_{\max}\sqrt{\omega}. \tag{15}$$

Proposition 4 shows that a signal $\mathbf{f}$ is uniquely determined and can be reconstructed by its local measurements $\{f_{\boldsymbol{\varphi}_i}\}_{i\in\mathcal{J}}$ if $\{\boldsymbol{\varphi}_i\}_{i\in\mathcal{J}}$ are known. The quantity $(f_{\boldsymbol{\varphi}_i} - \langle \mathbf{f}^{(k)}, \boldsymbol{\varphi}_i\rangle)$ is the estimate error between the original measurement and the reconstructed measurement at the kth iteration. According to the definition of $\widehat{\mathbf{G}}$ encoded in Definition 13, the iteration (18) can be rewritten as

$$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \widehat{\mathbf{G}}(\mathbf{f} - \mathbf{f}^{(k)}). \tag{16}$$

Therefore, in each iteration of ILMR, the new increment of the interpolated signal is obtained by first assigning the estimate errors to all vertices in the associated centerless local sets, and then projecting it onto the ω-bandlimited subspace.

Algorithm 5 Iterative Local Measurement Reconstruction.

1: Input: Graph $\mathcal{G}$, cutoff frequency ω, centerless local sets $\{\mathcal{N}_i\}_{i\in\mathcal{J}}$, local weights $\{\boldsymbol{\varphi}_i\}_{i\in\mathcal{J}}$, local measurements $\{f_{\boldsymbol{\varphi}_i}\}_{i\in\mathcal{J}}$;

2: Output: Reconstructed signal $\mathbf{f}^{(k)}$;

3: Initialization:

4: $$\mathbf{f}^{(0)} = \mathcal{P}_\omega\left(\sum_{i\in\mathcal{J}} f_{\boldsymbol{\varphi}_i}\boldsymbol{\delta}_{\mathcal{N}_i}\right); \tag{17}$$

5: **repeat**

6: $$\mathbf{f}^{(k+1)} = \mathbf{f}^{(k)} + \mathcal{P}_\omega\left(\sum_{i\in\mathcal{J}} (f_{\boldsymbol{\varphi}_i} - \langle \mathbf{f}^{(k)}, \boldsymbol{\varphi}_i\rangle)\boldsymbol{\delta}_{\mathcal{N}_i}\right); \tag{18}$$

7: **until** The stop condition is satisfied.

The procedures of ILMR in each iteration are illustrated in Fig. 7. Except for the difference of decimation and local measurement, the basic idea of ILMR is similar to that of IPR. In the assignment or propagating step, ILMR assigns the estimate errors of local measurements to vertices within the local sets, while IPR propagates the estimate errors of the decimated signal on the sampled vertices to other vertices

in the local sets. In fact, ILMR degenerates to IPR if the local weight concentrates on only one vertex (the sampled vertex) in each local set, in which case the local measurement degenerates to decimation.

The sufficient conditions and error bounds for ILMR and IPR are also different. Suppose the (centerless) local sets divisions in ILMR and IPR are exactly the same, i.e. the sampling set $\mathcal{S}$ in IPR can be written as $\{u_i\}_{i\in\mathcal{I}}$, where $\mathcal{I}$ is the index set in ILMR, then $\mathcal{N}_i$ equals $\mathcal{N}(u_i)$ for all $i \in \mathcal{I}$. According to Definitions 7 and 8, we have $R(u_i) \leq D_i$ and $K(u_i) \leq |\mathcal{N}(u_i)| = |\mathcal{N}_i|$. Therefore, $C_{\max}$ is not less than $Q_{\max}$. It implies that a more strict condition is needed for ILMR. It is reasonable because the sufficient condition for ILMR to guarantee the reconstruction is for all of the choices of local weights, which include decimation as a special case. However, since both sufficient conditions in Propositions 3 and 4 are not tight and there is still room for refinement, such a comparison only provides a rough analysis.

Remark 5 The projection operator $\mathcal{P}_\omega(\cdot)$ can be approximated by a polynomial expansion of the Laplacian, which is localized. As a consequence, ILMR can be approximately implemented in a localized way. In detail, the projection operator is written as

$$\mathcal{P}_\omega(\mathbf{f}) = \mathbf{U}\mathrm{diag}\left\{\hat{h}(\lambda_1), \ldots, \hat{h}(\lambda_N)\right\}\mathbf{U}^{\mathrm{T}}\mathbf{f},$$

where $\hat{h}(\cdot)$ denotes the lowpass filter

$$\hat{h}(\lambda) = \begin{cases} 1, & \lambda \leq \omega; \\ 0, & \text{elsewhere.} \end{cases}$$

Utilizing a polynomial approximation of $\hat{h}(\cdot)$ (e.g. Chebyshev polynomial expansion [10, 21]), one has

$$\hat{h}(\lambda) \approx \sum_{j=0}^{k} \alpha_j \lambda^j, 0 \leq \lambda \leq \lambda_N,$$

where $\{\alpha_j\}$ denote the coefficients and k is the order of the approximation, which is usually far smaller than N. Therefore the projection is approximated by a polynomial expansion of the Laplacian

$$\mathcal{P}_\omega(\mathbf{f}) \approx \mathbf{U}\mathrm{diag}\left\{\sum_{j=0}^{k}\alpha_j\lambda_1^j, \ldots, \sum_{j=0}^{k}\alpha_j\lambda_N^j\right\}\mathbf{U}^{\mathrm{T}}\mathbf{f} = \sum_{j=0}^{k}\alpha_j\mathbf{L}^j\mathbf{f}.$$

Because the Laplacian operator can be conducted by each vertex and its neighbors, the projection operator is approximately localized.

Remark 6 For potential applications, if the local measurements come from the result of some repeatable physical operations, the local weights are even not necessarily known when conducting ILMR. In detail, if $\{\boldsymbol{\varphi}_i\}_{i\in\mathcal{I}}$ is unknown but fixed, i.e., the

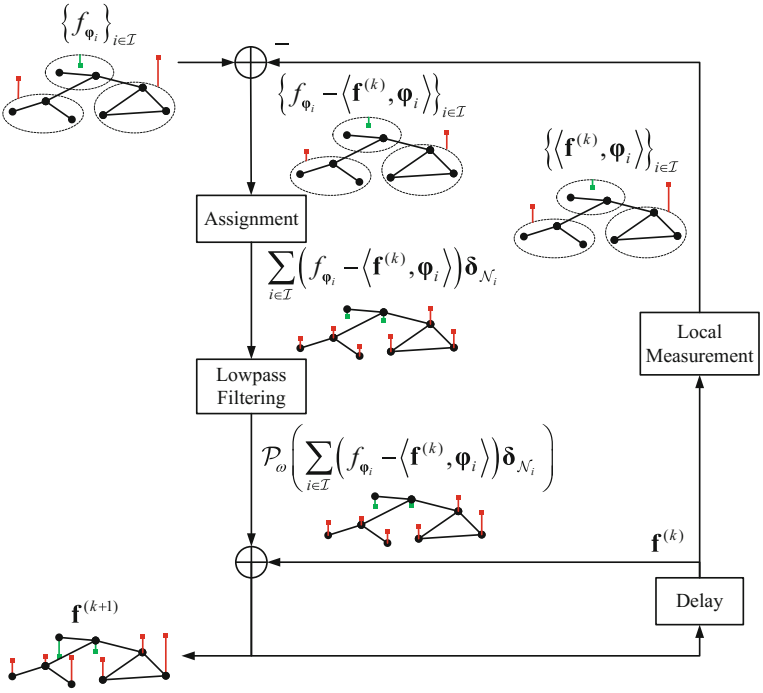


Fig. 7 Illustration of the iterations of ILMR

local measurement operation in Fig. 7 is a black box, $\langle \mathbf{f}^{(k)}, \boldsymbol{\varphi}_i \rangle$ can also be obtained by conducting the physical operations in each iteration. Therefore, the original signal can still be reconstructed by ILMR without exactly knowing $\{\boldsymbol{\varphi}_i\}_{i\in\mathcal{J}}$. This is a rather interesting result, and may facilitate graph signal reconstruction in specific scenarios.

Remark 7 If the bandlimited space is described as a subspace with a known dimensionality, rather than the cutoff frequency ω, the perfect reconstruction is achievable as a closed form by solving linear equations. However, the value of the iterative algorithm relies on its locality, which is important in graph related problems. Furthermore, iterative algorithms can be applied in potential online and distributed scenarios.

5.2 *Performance Analysis*

In this subsection, we study the error performance of ILMR when the original signal is corrupted by additive noise. We first derive the reconstruction error for incorrect measurement. Then the expected reconstruction error is calculated under the assumption of independent Gaussian noises and the optimal local weight is obtained in the

sense of minimizing the expected reconstruction error bound. Finally, in a special case of *i.i.d.* Gaussian perturbation, a greedy method for the centerless local sets partition and the selection of optimal local weights are provided.

5.2.1 Reconstruction Error in the Noisy Scenario

Suppose that the observed signal associated with each vertex is corrupted by additive noise. The corrupted signal is denoted as $\tilde{\mathbf{f}} = \mathbf{f} + \mathbf{n}$, where $\mathbf{n}$ denotes the noise. In the kth iteration of ILMR, the corrupted local measurements $\{\langle \tilde{\mathbf{f}}, \boldsymbol{\varphi}_i \rangle\}_{i\in\mathcal{J}}$ are utilized to produce the temporary reconstruction of $\tilde{\mathbf{f}}^{(k)}$.

The following proposition gives a reconstruction error bound of $\tilde{\mathbf{f}}^{(k)}$.

Proposition 5 ([8]) *For given centerless local sets and the associated weights* $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i\in\mathcal{J}}$, $\mathbf{f} \in PW_\omega(\mathcal{G})$ *is corrupted by additive noise* $\mathbf{n}$. *If* ω *is less than* $1/C_{\max}^2$, *in the kth iteration the output of ILMR using the corrupted local measurements* $\{\langle \tilde{\mathbf{f}}, \boldsymbol{\varphi}_i \rangle\}_{i\in\mathcal{J}}$ *satisfies*

$$\|\tilde{\mathbf{f}}^{(k)} - \mathbf{f}\| \leq \frac{\tilde{n}}{1-\gamma} + \gamma^{k+1} \left(\|\mathbf{f}\| + \|\mathbf{n}\|\right), \tag{19}$$

where γ *is defined as (15),* $\tilde{n}$ *is defined as*

$$\tilde{n} = \sum_{i\in\mathcal{J}} \sqrt{|\mathcal{N}_i|} \cdot |n_i|, \tag{20}$$

and n_i *is the equivalent noise of centerless local set* $\mathcal{N}_i$, *defined as*

$$n_i = \langle \mathbf{n}, \boldsymbol{\varphi}_i \rangle = \sum_{v\in\mathcal{N}_i} n(v)\varphi_i(v). \tag{21}$$

From (19) it can be seen that in the noisy scenario the reconstruction error is controlled by the sum of two parts. The first one is a weighted sum of the equivalent noises of all the local sets, while the second one is decaying with the increase of iteration number. The first part is crucial as the iteration goes on. Thus minimizing the first part, which is determined by both partition of centerless local sets and local weights, improves the performance of ILMR in the noisy scenario.

5.2.2 Gaussian Noise and Optimal Local Weights

For a given partition $\{\mathcal{N}_i\}_{i\in\mathcal{J}}$, some prior knowledge of unknown noise $\mathbf{n}$ brings the possibility to design optimal local weights. In fact, the optimal local weights can also be studied in other criterions, e.g. the fastest convergence. Here we consider the

optimal local weights in the sense of minimizing the expected reconstruction error bound. We assume the noises associated with different vertices are independent.

Suppose the noise follows zero-mean Gaussian distribution, i.e., $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is a diagonal matrix and the noise of vertex v satisfies $n(v) \sim \mathcal{N}(0, \sigma^2(v))$. Then $\tilde{n}$ defined in (20) is a random variable.

For centerless local set $\mathcal{N}_i$, according to (21), the equivalent noise n_i also follows a Gaussian distribution $n_i \sim \mathcal{N}(0, \sigma_i^2)$, where

$$\sigma_i^2 = \sum_{v \in \mathcal{N}_i} \sigma^2(v) \varphi_i^2(v). \tag{22}$$

Then $|n_i|$ follows the half-normal distribution with its expectation satisfying

$$\mathrm{E}\{|n_i|\} = \sigma_i \sqrt{\frac{2}{\pi}}.$$

According to (20), the expectation of $\tilde{n}$ is

$$\mathrm{E}\{\tilde{n}\} = \sqrt{\frac{2}{\pi}} \sum_{i \in \mathcal{I}} \sqrt{|\mathcal{N}_i|} \sigma_i. \tag{23}$$

Then the following corollary is ready to obtain.

Corollary 4 ([8]) *For given centerless local sets and the associated weights* $\{(\mathcal{N}_i, \boldsymbol{\varphi}_i)\}_{i \in \mathcal{I}}$*, the original signal* $\mathbf{f} \in PW_\omega(\mathcal{G})$*, assuming the noise associated with vertex* v *follows independent Gaussian distribution* $\mathcal{N}(0, \sigma^2(v))$*, if* ω *is less than* $1/C_{\max}^2$*, the expected reconstruction error of ILMR in the* k*th iteration satisfies*

$$\mathrm{E}\left\{\|\tilde{\mathbf{f}}^{(k)} - \mathbf{f}\|\right\} \leq \frac{1}{1-\gamma} \sqrt{\frac{2}{\pi}} \sum_{i \in \mathcal{I}} \sqrt{|\mathcal{N}_i|} \sigma_i + \mathcal{O}\left(\gamma^{k+1}\right), \tag{24}$$

where γ *is defined as (15), and* σ_i *is defined as (22).*

By minimizing the right hand side of (24), the optimal choice of local weights can be derived.

Corollary 5 ([8]) *For a given division of centerless local sets* $\{\mathcal{N}_i\}_{i \in \mathcal{I}}$*, if the noises associated with the vertices are independent and follow zero-mean Gaussian distributions* $n(v) \sim \mathcal{N}(0, \sigma^2(v))$*, then the optimal local weights* $\{\boldsymbol{\varphi}_i\}_{i \in \mathcal{I}}$ *are*

$$\varphi_i(v) = \begin{cases} \dfrac{(\sigma^2(v))^{-1}}{\sum_{v \in \mathcal{N}_i} (\sigma^2(v))^{-1}}, & v \in \mathcal{N}_i; \\ 0, & v \notin \mathcal{N}_i. \end{cases} \tag{25}$$

The above analysis shows that in the sense of minimizing the expected reconstruction error, the optimal local weight associated with vertex v within $\mathcal{N}_i$ is inversely proportional to the noise variance of v. This is evident because more information are reserved in the sampling process if a larger local weight is assigned to a vertex with smaller noise variance. However, it should be noted that compared with the optimal local measurement, assigning all the weights in $\mathcal{N}_i$ to the vertex with the smallest noise variance, i.e. the optimal decimation, is not the best choice. In fact, the optimal choice of local measurements is consistent with the well-known inverse variance weighting in statistics [15].

Therefore, local measurement reduces the disturbance of noise and reconstruct the original signal more precisely. In other words, for given partition of centerless local sets, graph signal reconstruction from local measurements with the optimal weights performs better than reconstruction from decimation, even when the vertices with the smallest noise variance are chosen in the latter sampling scheme.

5.2.3 A Special Case of Independent and Identical Distributed Gaussian Noise

Specifically, if noise variances are the same for all the vertices, i.e., $\sigma(v)$ equals σ for any $v \in \mathcal{V}$, $\tilde{n}$ can be approximately written in a more explicit form. For $\mathcal{N}_i$, the optimal local weight is equal for all the vertices in $\mathcal{N}_i$. Thus $\varphi_i(v)$ equals $1/|\mathcal{N}_i|$ for $v \in \mathcal{N}_i$, and in this case, $\sqrt{|\mathcal{N}_i|}n_i$ follows a Gaussian distribution,

$$\sqrt{|\mathcal{N}_i|}n_i \sim \mathcal{N}(0, \sigma^2).$$

Then $\sqrt{|\mathcal{N}_i|} \cdot |n_i|$ follows the half-normal distribution with the same parameter σ. The above analysis shows that each term of the sum in (20) follows independent and identical half-normal distribution, with its expectation and variance satisfying

$$\begin{aligned} \mathrm{E}\left\{\sqrt{|\mathcal{N}_i|} \cdot |n_i|\right\} &= \sigma\sqrt{\frac{2}{\pi}}, \\ \mathrm{Var}\left\{\sqrt{|\mathcal{N}_i|} \cdot |n_i|\right\} &= \sigma^2\left(1 - \frac{2}{\pi}\right). \end{aligned}$$

Assuming that the number of local sets $|\mathcal{I}|$ is large, by the central limit theorem, $\tilde{n}$ follows a Gaussian distribution approximately,

$$\tilde{n} \sim \mathcal{N}\left(|\mathcal{I}|\sigma\sqrt{\frac{2}{\pi}}, |\mathcal{I}|\sigma^2\left(1 - \frac{2}{\pi}\right)\right).$$

Then we have the following corollary.

Corollary 6 ([8]) *For given centerless local sets* $\{\mathcal{N}_i\}_{i\in\mathcal{J}}$ *and the associated weights* $\varphi_i(v) = 1/|\mathcal{N}_i|$ *for* $v \in \mathcal{N}_i$*, the original signal* $\mathbf{f} \in PW_\omega(\mathcal{G})$*, assuming the noise associated with each vertex follows* i.i.d *Gaussian distribution* $\mathcal{N}(0, \sigma^2)$*, if* ω *is less than* $1/C_{\max}^2$*, the expected reconstruction error of ILMR in the kth iteration satisfies*

$$\mathrm{E}\left\{\|\tilde{\mathbf{f}}^{(k)} - \mathbf{f}\|\right\} \leq \frac{|\mathcal{J}|\sigma}{1-\gamma}\sqrt{\frac{2}{\pi}} + \mathcal{O}\left(\gamma^{k+1}\right), \tag{26}$$

where γ *is defined as (15).*

According to (26), the error bound is affected by the number of centerless local sets $|\mathcal{J}|$. A division with fewer sets may reduce the expected reconstruction error. However, it should be noted that the number of centerless local sets cannot be too small to satisfy the condition

$$\gamma = C_{\max}\sqrt{\omega} = \max_{i\in\mathcal{J}} \sqrt{|\mathcal{N}_i|D_i\omega} < 1,$$

which is determined by the cutoff frequency of the original graph signal. Besides, the factor $1/(1-\gamma)$ in (26) implies that a smaller $C_{\max}$, which leads to a smaller γ, also reduces the error bound. A rough calculation can be given to balance the two factors. If there are not too many vertices in each $\mathcal{N}_i$, we have that $C_{\max}$ approximates to $N_{\max}$, where $N_{\max}$ is the largest cardinality of centerless local sets. Since $N_{\max}|\mathcal{J}|$ approximates to N, we have

$$\frac{1}{1-\gamma}|\mathcal{J}| \approx \frac{1}{1-\sqrt{\omega}N_{\max}} \cdot \frac{N}{N_{\max}}.$$

To minimize the above quantity, a near optimal $N_{\max}$ is

$$N_{\max} = \frac{1}{2\sqrt{\omega}}, \tag{27}$$

i.e., γ approximates to $1/2$. It provides a strategy to partition centerless local sets. For given cutoff frequency ω, an approximated $N_{\max}$ can be chosen according to (27), then the graph is divided into local sets to make sure that $|\mathcal{N}_i|$ is not more than $N_{\max}$ and the number of local sets is as small as possible.

For a given $N_{\max}$, a greedy algorithm is proposed to make the division of centerless local sets, as shown in Algorithm 6. The greedy algorithm is to iteratively remove connected vertices with the smallest degrees from the original graph into the new set, until the cardinality of the new set reaches $N_{\max}$ or there is no connected vertex. The reason for choosing the smallest-degree vertex is that such a vertex is more likely to be on the border of a graph.

Algorithm 6 A greedy method to partition centerless local sets with maximal cardinality.

Input: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Maximal cardinality $N_{\max}$;
Output: Centerless local sets $\{\mathcal{N}_i\}_{i\in\mathcal{I}}$;
Initialization: $i = 0$;
repeat
  Find one vertex with the smallest degree in $\mathcal{G}$, $u \in \arg\min_{v\in\mathcal{V}} d_{\mathcal{G}}(v)$;
  Update $i = i + 1$, $\mathcal{N}_i = \{u\}$;
  Obtain the neighbor set of $\mathcal{N}_i$, $\mathcal{S}_i = \{v \in \mathcal{G} | v \sim w, w \in \mathcal{N}_i, v \notin \mathcal{N}_i\}$;
  repeat
    Find one vertex with the smallest degree in $\mathcal{S}_i$, $u \in \arg\min_{v\in\mathcal{S}_i} d_{\mathcal{G}}(v)$;
    Update $\mathcal{N}_i = \mathcal{N}_i \cup \{u\}$;
    Update $\mathcal{S}_i = \{v \in \mathcal{G} | v \sim w, w \in \mathcal{N}_i, v \notin \mathcal{N}_i\}$;
  until $|\mathcal{N}_i| = N_{\max}$ or $\mathcal{S}_i = \emptyset$
  Remove the edges, $\mathcal{E} = \mathcal{E}\backslash\{(p, q) | p \in \mathcal{N}_i, q \in \mathcal{V}\}$;
  Remove the vertices, $\mathcal{V} = \mathcal{V}\backslash\mathcal{N}_i$ and $\mathcal{G} = \mathcal{G}(\mathcal{V}, \mathcal{E})$;
until $\mathcal{V} = \emptyset$

6 Relationship with Time Domain Results

Bandlimited signal sampling and reconstruction on graph is closely related to irregular sampling [16, 17] or nonuniform sampling [18] in the time domain, which sheds light on the analysis of graph signal. There have existed several iterative reconstruction methods and theoretical analysis of time-domain irregular sampling [16, 19–21], some of which are related to the frame theory [16, 20, 22]. Some further works extend the results to high dimensional spaces [16] and manifolds [23, 24].

By exploiting the similarities between time-domain irregular sampling and graph signal sampling, some results of this work have consistent formulation with the corresponding results in the time domain. The reconstruction methods also have correspondences in the time domain.

Results on time-domain irregular sampling show that $\{\mathcal{T}_{t_i}\mathrm{sinc}_\Omega\}_{t_i\in\mathcal{S}}$ is a frame for $\mathcal{B}^2_\Omega$ if the sampling set $\mathcal{S}$ satisfies some particular conditions, where $\mathcal{T}_{t_i} f(t) = f(t - t_i)$ denotes the translation of $f(t)$, sinc_Ω denotes the sinc function whose bandwidth is Ω and $\mathcal{B}^2_\Omega$ denotes the space of Ω-bandlimited square integrable signal.

Correspondingly, for the graph signal $\mathbf{f} \in PW_\omega(\mathcal{G})$, $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathcal{S}}$ is a frame under some conditions. The result is consistent with that in the time domain. The correspondence between irregular sampling in the time domain and that on graph is illustrated in Fig. 8. In the graph signal sampling problem, $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathcal{S}}$ corresponds to the frame $\{\mathcal{T}_{t_i}\mathrm{sinc}_\Omega\}_{t_i\in\mathcal{S}}$ in the time domain. The essence of the two problem is very similar and theoretical results on sampling and reconstruction of graph signals can be enlightened by irregular sampling in the time domain.

The ideas of graph signal reconstruction methods ILSR, IWR and IPR have correspondences in time-domain, which are Marvasti method [20], adaptive weights method [16] and Voronoi method [21], respectively. However, a graph is discrete

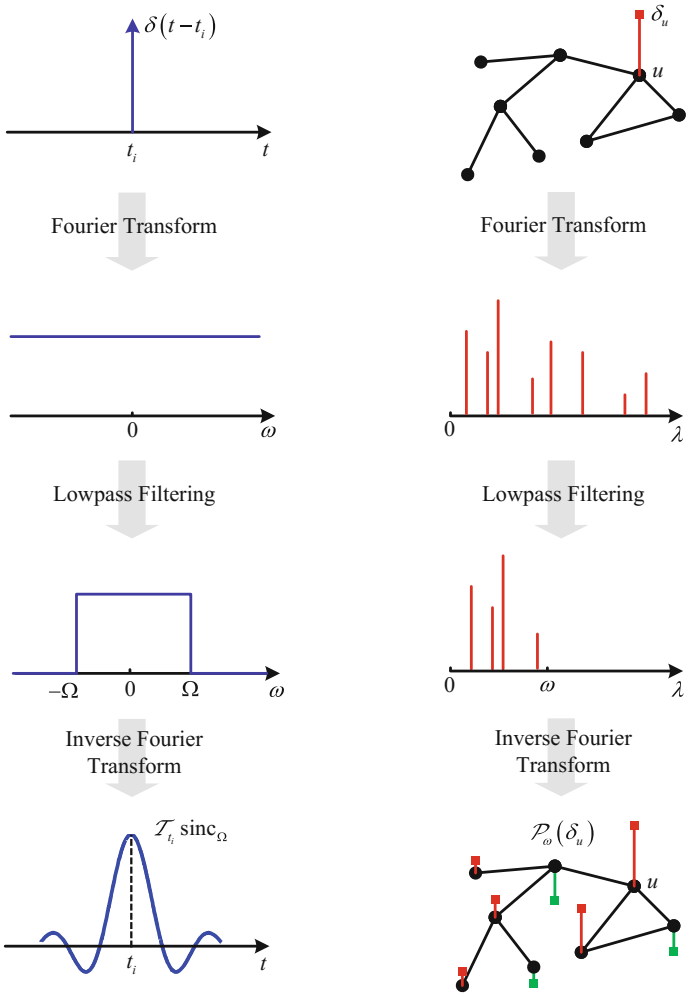


Fig. 8 The correspondence between irregular sampling in the time domain and that on graph. Both $\mathcal{T}_{t_i}\,\mathrm{sinc}_\Omega$ and $\mathcal{P}_\omega(\boldsymbol{\delta}_u)$ are the projections of δ-functions onto bandlimited spaces. Under certain conditions, for all the sampled points or vertices, these kinds of signals $\{\mathcal{T}_{t_i}\,\mathrm{sinc}_\Omega\}_{t_i\in\mathbb{S}}$ and $\{\mathcal{P}_\omega(\boldsymbol{\delta}_u)\}_{u\in\mathbb{S}}$ become frames. Consequently, the original signals can be reconstructed by the sampled data

and the local topology of each sampling vertex is irregular, which leads to some new problems related to local sets in the sampling and reconstruction of graph signals.

There is also corresponding time-domain result [21] for local measurement based graph signal reconstruction. Bandlimited time-domain signals can be reconstructed from irregular sampled local averages, which is

$$f_i = \int f(x)u_i(x)\mathrm{d}x = \langle u_i, f\rangle,$$

Table 2 The correspondence between irregular sampling in the time domain and that on graph

Terms	Time domain	Vertex domain
Signal	$f(t)$	$\mathbf{f}$
Cutoff frequency	Ω	ω
Low-frequency space	$\mathcal{B}_{\Omega}^{2}$	$PW_{\omega}(\mathcal{G})$
Shifted impulse	$\delta(t-t_i)$	$\boldsymbol{\delta}_u$
Shifted sinc function	$\mathcal{T}_{t_i}\text{sinc}_{\Omega}$	$\mathcal{P}_{\omega}(\boldsymbol{\delta}_u)$
Neighborhood	$[(t_{i-1}+t_i)/2, (t_i+t_{i+1})/2)$	$\mathcal{N}(u)$
Neighbor indicator	$1_{[(t_{i-1}+t_i)/2,(t_i+t_{i+1})/2)}$	$\boldsymbol{\delta}_{\mathcal{N}(u)}$
Weight	$\sqrt{(t_{i+1}-t_{i-1})/2}$	$\sqrt{\lvert\mathcal{N}(u)\rvert}$
Reconstruction method	Marvasti method	ILSR
Reconstruction method	Adaptive weights method	IWR
Reconstruction method	Voronoi method	IPR
Reconstruction method	Gröchenig method	ILMR

where the ith averaging function u_i satisfies

$$\text{supp}u_i \subset [x_i - \delta/2, x_i + \delta/2], 0 \leq u_i(x) \leq 1, \int u_i(x)\mathrm{d}x = 1$$

and x_i is the ith sampling point.

The averaging function u_i is the time-domain correspondence of the local weight ϕ_i in ILMR. The local average f_i is the time-domain correspondence of local measurement f_{ϕ_i}. ILMR can be regarded as the graph-based version of the reconstruction algorithm in [21].

The correspondence between time-domain irregular sampling and graph signal sampling is shown in Table 2.

7 Numerical Experiments

The Minnesota road graph [25] is chosen as the graph, which has 2640 vertices and 6604 edges, to test the proposed reconstruction algorithms. The bandlimited signal is generated by first generating a random signal and then removing its high-frequency components.[1]

[1]The MATLAB codes for the proposed methods and all experiments are available at http://gu.ee.tsinghua.edu.cn/publications.

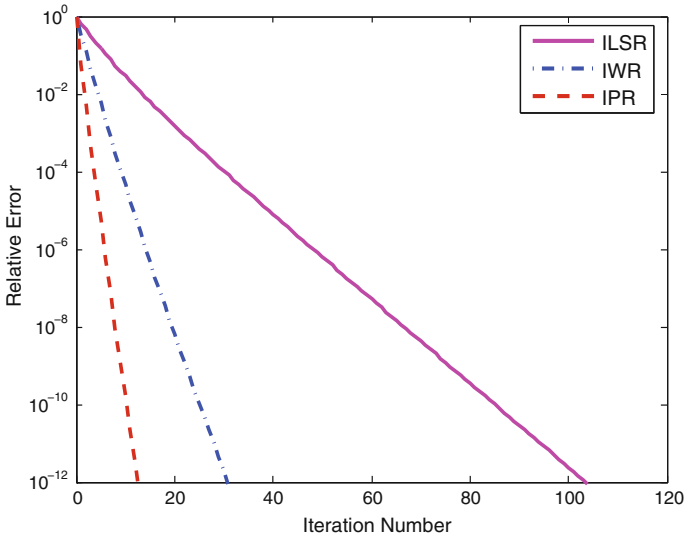


Fig. 9 Convergence curves of ILSR, IWR, and IPR

7.1 Local-Set-Based Reconstruction Algorithms

In the first experiment, the convergence rates of the three algorithms are compared. A one-hop sampling set satisfying (14) is chosen as the sampling set. The one-hop sampling set and the corresponding local sets are obtained by the greedy method in Algorithm 4. This sampling set has 872 vertices, which is about one third of all. The cutoff frequency is 0.25. The convergence curves of ILSR, IWR, and IPR are illustrated in Fig. 9. It is obvious that the convergence rate of the proposed algorithms is significantly improved compared with the reference. Furthermore, IPR is better than IWR on the convergence rate. Both observations are in accordance with the analysis in Sect. 4.

The second experiment is to verify that the choice of sampling set may affect the performance of convergence. Two different sampling sets are used to reconstruct the same bandlimited original signal. Both of the sampling sets have the same amount of vertices. The first sampling set is the one-hop set satisfying (14), with 872 sampled vertices and $Q_{\max} = 1$. For the second sampling set, 872 vertices are selected uniformly at random among all the vertices. Each unsampled vertex belongs to the local set associated with its nearest sampled vertex. Then $K(u)$ and $R(u)$ can be obtained for each local set, and we have $Q_{\max} = \sqrt{40}$ with the corresponding $K(u) = 8$ and $R(u) = 5$. The convergence curves of the two sampling sets using the three reconstruction methods are illustrated in Fig. 10. For all the algorithms, the convergence is faster by using the sampled data of the one-hop sampling set than by using the randomly chosen vertex set. It means that the sampling geometry has influence on

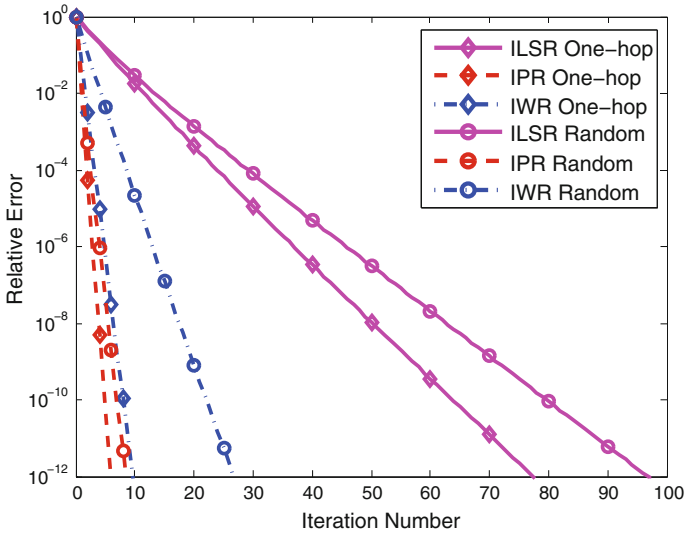


Fig. 10 Convergence curves of ILSR, IWR, and IPR, on one-hop sampling set and randomly chosen sampling set

the reconstruction. A sampling set and the local sets with smaller $Q_{\max}$ may converge faster.

Suppose there is noise involved in the observation of sampled graph signal. The last experiment focuses on the robustness to the observation noise of the three algorithms. In this experiment the noise is generated as independent identical distributed Gaussian sequence. As shown in Fig. 11, the steady-state error decreases as the SNR increases. The three methods have almost the same performance against observation noise.

7.2 Local-Measurement-Based Reconstruction Algorithm

The centerless local sets are generated by the greedy method in Algorithm 6 using given $N_{\max}$. Five kinds of local weights are tested including

1. uniform weight, where $\varphi_i(v)$ equals $1/|\mathcal{N}_i|, \forall v \in \mathcal{N}_i$;
2. random weight, where

$$\varphi_i(v) = \frac{\varphi_i'(v)}{\sum_{u\in\mathcal{N}_i}\varphi_i'(u)}, \quad \forall v \in \mathcal{N}_i, \varphi_i'(u) \sim \mathcal{U}(0,1),$$

   and $\mathcal{U}(0, 1)$ denotes the uniform distribution;
3. Dirac delta weight, where $\boldsymbol{\varphi}_i$ equals $\boldsymbol{\delta}_u$ for a randomly chosen $u \in \mathcal{N}_i$;

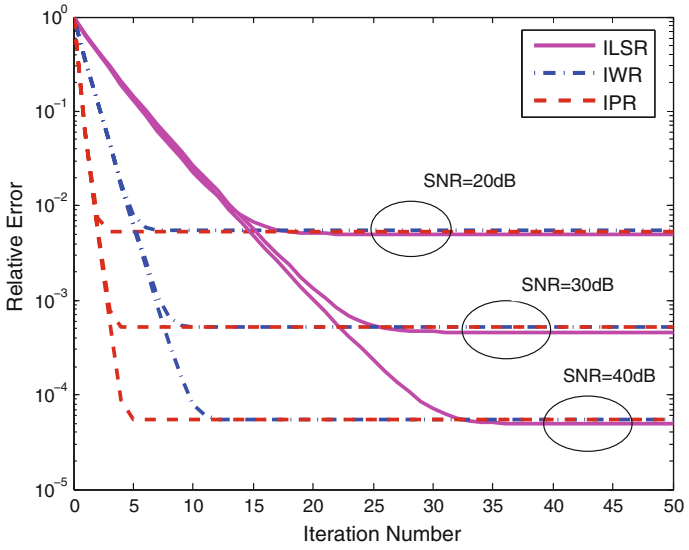


Fig. 11 Convergence curves of three reconstruction methods with various observation SNR

4. the optimal weight, where

$$\varphi_i(v) = \frac{(\sigma^2(v))^{-1}}{\sum_{v\in\mathcal{N}_i}(\sigma^2(v))^{-1}}, \quad \forall v \in \mathcal{N}_i;$$

5. the optimal Dirac delta weight, where $\boldsymbol{\varphi}_i$ equals $\boldsymbol{\delta}_u$ for

$$u = \arg\min_{u\in\mathcal{N}_i} \sigma^2(u).$$

Notice that case 3 and case 5 degenerate ILMR to IPR.

In the first experiment, the convergence of the proposed ILMR is verified for various centerless local sets partition and local weights. The graph is divided into 709 and 358 centerless local sets for $N_{\max}$ equals 4 and 8, respectively. Three kinds of local weights are tested including case 1, 2, and 3. The averaged convergence curves are plotted in Fig. 12 for 100 randomly generated original graph signals. According to Fig. 12, the convergence is accelerated when the graph is divided into more local sets and has a smaller $N_{\max}$. It is intuitive because more local sets will bring more measurements and increase the sampling rate, which provides more information in the reconstruction. According to (15), for the same ω, a smaller $N_{\max}$ leads to a smaller γ, and guarantees a faster convergence. The experimental result also shows that in the noise-free scenario, reconstruction with uniform weight converges slightly faster than that with random weight. However, both above cases converge much faster than reconstruction with Dirac delta weight. This means that local-measurement-

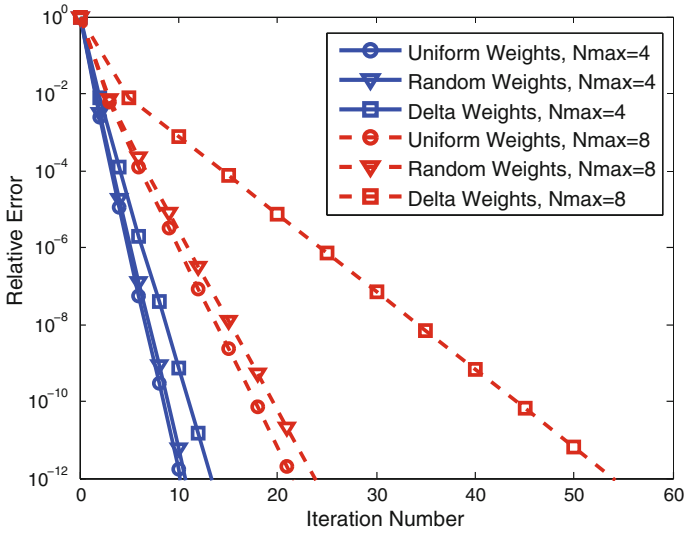


Fig. 12 The convergence behavior of ILMR for various division of centerless local sets and different local weights

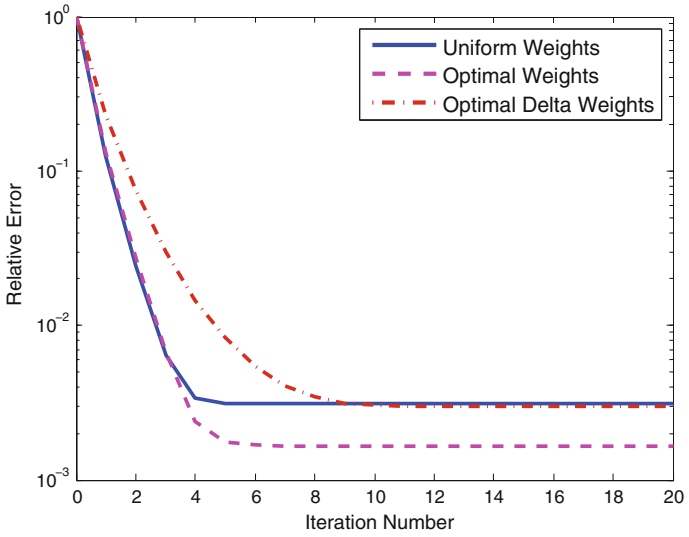


Fig. 13 The convergence curves of reconstruction with uniform weights, the optimal weights, and optimal Dirac delta weights when independent zero-mean Gaussian noise is added to each vertex

based ILMR behaves better than decimation-based IPR by combining the signals on different vertices properly.

In the second experiment, independent zero-mean Gaussian noise is added to each vertex with different variance. The original signal is normalized with unit norm. All

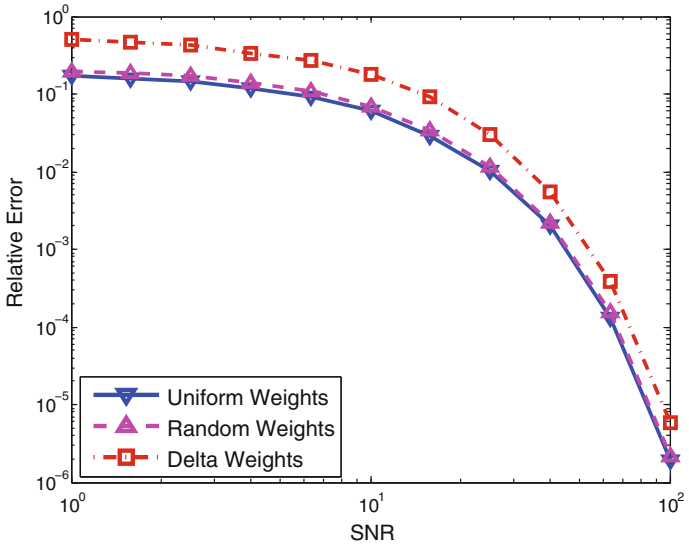


Fig. 14 Relative errors of ILMR under difference SNRs with various choices of local weights. The noise associated with each vertex is *i.i.d.* Gaussian

of the vertices are randomly divided into three groups with the standard deviations of the noise chosen as σ equals 1×10^{-4}, 2×10^{-4}, and 5×10^{-4}, respectively. The graph is partitioned into 358 centerless local sets with $N_{\max}$ equals 8. Three kinds of local weights are tested including case 1, 4, and 5. The averaged convergence curves are illustrated in Fig. 13 for 100 randomly generated original graph signals. The steady-state relative error with the optimal weight is smaller than those with uniform weight and the optimal Dirac delta weight. The experimental result verifies the analysis in Sect. 5. It implies that a better selection of local weights can reduce the reconstruction error if the noise variances on vertices are different.

In the last experiment, the performance of the proposed algorithm against *i.i.d.* Gaussian noise are tested for three kinds of local weights including case 1, 2, and 3. In this case the optimal local weights is equivalent to uniform weights. The graph is partitioned into 358 centerless local sets with $N_{\max}$ equals 8. The relative reconstruction errors of three tests are illustrated in Fig. 14. Each point is the average of 100 trials. The experimental result shows that for *i.i.d.* Gaussian noise, reconstruction with uniform weight or random weight performs beyond that with Dirac delta weight, which is actually the traditional sampling scheme of decimation. It shows that compared with decimation, the proposed generalized sampling scheme is more robust against noise, as analyzed in Sect. 5.

8 Conclusion

In this chapter, the problem of graph signal sampling and reconstruction in bandlimited space is studied. We first introduce the concepts of local set and centerless local set and then generalize sampling from decimation to measurement. Based on frame theory, two operators named local propagation and local-measurement-based propagation are presented and proved to be contraction mapping. Consequently, several series of signals are proved to be frames and their frame bounds are estimated. Above theory provides solid foundation for developing efficient reconstruction algorithms. Following the traditional decimation scheme, two local-set-based iterative methods called IWR and IPR are proposed to reconstruct the missing data from the observed samples. Strict proofs of convergence and error bounds of IWR and IPR are presented. Aiming at the new measurement scheme, a reconstruction algorithm ILMR is proposed to perfectly reconstruct original bandlimited signals iteratively. The convergence of ILMR is proved and its performance in noisy scenarios is analyzed. The optimal local weights are given to minimize the effect of noise, and a greedy algorithm for local sets partition is proposed. After comprehensive discussion on the proposed algorithms, we explore the correspondence between time-domain irregular sampling and graph signal sampling, which sheds light on the analysis in the graph vertex domain. Experiments, which verify the theoretical analysis, show that both IWR and IPR converge significantly faster than the reference algorithm ILSR. In addition, the local measurement sampling scheme together with reconstruction algorithm is more robust against additive noise.

9 Additional Reading Material

There has been some theoretical analysis on the sampling and reconstruction of bandlimited graph signals [10, 26–28]. Some existing works focus on the theoretical conditions for the exact reconstruction of bandlimited signals. The relationships between the sampling sets of unique reconstruction and the cutoff frequency of bandlimited signal space are established for normalized Laplacian [10] and unnormalized Laplacian [27, 28], respectively. A necessary and sufficient condition of exact reconstruction is established in [11]. In order to reconstruct bandlimited graph signals from sampled data, several methods have been proposed. In [3] a least square approach is proposed to solve this problem. Furthermore, an iterative reconstruction method is proposed and a tradeoff between smoothness and data-fitting is introduced for real world applications [12].

The idea of local measurements can be traced back to time-domain nonuniform sampling [18], or irregular sampling [16, 17], which has a close relationship with graph signal sampling and reconstruction. For the signals in time-domain [16, 21], shift-invariant space [29], or on manifolds [23, 24], based on the theoretical results of signal reconstruction from samples, there have been extended works on reconstruct-

ing signals from local averages. Time-domain local averages are taken from small intervals around the samples with proper averaging functions. Theoretical results show that bandlimited original signals can be accurately recovered if the cutoff frequency is smaller than a quantity which is inversely proportional to the length of intervals [21]. However, there are few such works on graph-signal-related problems. As far as we know, the only work related to local aggregation for graph signals is applying the graph-shift operator sequentially [30], which is different from our problem.

The problem of signal reconstruction is closely related to the frame theory, which is also involved in other areas of graph signal processing, e.g., wavelet and vertex-frequency analysis on graphs [31]. Based on windowed graph Fourier transform and vertex-frequency analysis, windowed graph Fourier frames are studied in [32]. A spectrum-adapted tight vertex-frequency frame is proposed in [33] via translation on the graph. These works focus on vertex-frequency frames whose elements make up over-representation dictionaries, while in the reconstruction problem the frames are always composed by elements centering at the vertices in the sampling sets.

Acknowledgements This work was funded by the National Natural Science Foundation of China (NSFC 61571263 and NSFC 61531166005), the National Key Research and Development Program of China (Project No. 2016YFE0201900 and 2017YFC0403600), and Tsinghua University Initiative Scientific Research Program (Grant 2014Z01005).

References

1. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
2. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs. IEEE Trans. Signal Process. **61**(7), 1644–1656 (2013)
3. S.K. Narang, A. Gadde, A. Ortega, Signal processing techniques for interpolation in graph structured data, in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2013), pp. 5445–5449
4. S. Chen, A. Sandryhaila, J.M.F. Moura, J. Kovačević, Adaptive graph filtering: multiresolution classification on graphs, in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2013), pp. 427–430
5. S. Chen, A. Sandryhaila et al., Signal inpainting on graphs via total variation minimization, in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2014), pp. 8267–8271
6. X. Dong, D. Thanou, P. Frossard, P. Vandergheynst, Learning graphs from signal observations under smoothness prior (2014), arXiv:1406.7842
7. X. Wang, P. Liu, Y. Gu, Local-set-based graph signal reconstruction. IEEE Trans. Signal Process. **63**(9), 2432–2444 (2015)
8. X. Wang, J. Chen, Y. Gu, Local measurement and reconstruction for noisy bandlimited graph signals. Signal Process. **129**, 119–129 (2016)
9. F.R.K. Chung, *Spectral Graph Theory* (American Mathematical Society, Providence, 1997)
10. I. Pesenson, Sampling in Paley–Wiener spaces on combinatorial graphs. Trans. Am. Math. Soc. **360**(10), 5603–5627 (2008)

11. A. Anis, A. Gadde, A. Ortega, Towards a sampling theorem for signals on arbitrary graphs, in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2014), pp. 3892–3896
12. S.K. Narang, A. Gadde, E. Sanou, A. Ortega, Localized iterative methods for interpolation in graph structured data, in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2013), pp. 491–494
13. O. Christensen, *An Introduction to Frames and Riesz Bases* (Springer, Berlin, 2003)
14. M. Vetterli, J. Kovačević, V.K. Goyal, *Foundations of Signal Processing* (Cambridge University Press, Cambridge, 2014)
15. M.W. Lipsey, D.B. Wilson, *Practical Meta-Analysis* (Sage Publications, Thousand Oaks, 2001)
16. H.G. Feichtinger, K. Gröchenig, Theory and practice of irregular sampling, *Wavelets: Mathematics and Applications* (1994), pp. 305–363
17. K. Gröchenig, A discrete theory of irregular sampling. Linear Algebr. Appl. **193**, 129–150 (1993)
18. F. Marvasti, *Nonuniform Sampling: Theory and Practice* (Springer, Berlin, 2001)
19. K.D. Sauer, J.P. Allebach, Iterative reconstruction of bandlimited images from nonuniformly spaced samples. IEEE Trans. Circuits Syst. **34**(12), 1497–1506 (1987)
20. F. Marvasti, M. Analoui, M. Gamshadzahi, Recovery of signals from nonuniform samples using iterative methods. IEEE Trans. Signal Process. **39**(4), 872–878 (1991)
21. K. Gröchenig, Reconstruction algorithms in irregular sampling. Math. Comput. **59**(199), 181–194 (1992)
22. J.J. Benedetto, Irregular sampling and frames, *Wavelets: A Tutorial in Theory and Applications*, vol. 2 (1992), pp. 445–507
23. I. Pesenson, Poincaré-type inequalities and reconstruction of Paley–Wiener functions on manifolds. J. Geom. Anal. **14**(1), 101–121 (2004)
24. H. Feichtinger, I. Pesenson, Recovery of band-limited functions on manifolds by an iterative algorithm. Contemp. Math. **345**, 137–152 (2004)
25. D. Gleich, The MatlabBGL Matlab Library [Online], http://www.cs.purdue.edu/homes/dgleich/packages/matlab_bgl/index.html
26. I. Pesenson, Variational splines and Paley–Wiener spaces on combinatorial graphs. Constr. Approx. **29**, 1–21 (2009)
27. I.Z. Pesenson, M.Z. Pesenson, Sampling, filtering and sparse approximations on combinatorial graphs. J. Fourier Anal. Appl. **16**(6), 921–942 (2010)
28. H. Führ, I.Z. Pesenson, Poincaré and Plancherel–Polya inequalities in harmonic analysis on weighted combinatorial graphs. SIAM J. Discret. Math. **27**(4), 2007–2028 (2013)
29. A. Aldroubi, Non-uniform weighted average sampling and reconstruction in shift-invariant and wavelet spaces. Appl. Comput. Harmon. Anal. **13**(2), 151–161 (2002)
30. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Sampling of graph signals with successive local aggregations. IEEE Trans. Signal Process. (2015)
31. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
32. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs, EPFL-ARTICLE-187669, Elsevier, 2013
33. D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames, EPFL-ARTICLE-190280, Institute of Electrical and Electronics Engineers, 2013

Time-Varying Graph Signals Reconstruction



Xianghui Mao and Yuantao Gu

Abstract Signal processing on graphs is an emerging research field dealing with signals living on an irregular domain that is captured by a graph, and has been applied to sensor networks, machine learning, climate analysis, et cetera. Existing works on sampling and reconstruction of graph signals mainly studied static bandlimited signals. However, many real-world graph signals are time-varying, and they evolve smoothly, so instead of the signals themselves being bandlimited or smooth on graph, it is more reasonable that their temporal differences are smooth on graph. In this chapter, two new batch reconstruction methods of time-varying graph signals are proposed by exploiting the smoothness of the temporal difference signals, and the uniqueness as well as the reconstruction error bound of the solutions to the corresponding optimization problems are theoretically analyzed. Furthermore, driven by practical applications faced with real-time requirements, huge size of data, lack of computing center, or communication difficulties between two non-neighboring vertices, an online distributed method is proposed by applying local properties of the temporal difference operator and the graph Laplacian matrix. We also highlight the spatio-temporal signals prevalently existing in sociology, climatology, and environmental studies form a special type of time-varying graph signals, and can be reconstructed by the proposed methods. Experiments on a variety of real-world datasets demonstrate the excellent performance of the proposed methods.

X. Mao · Y. Gu (✉)
Beijing National Research Center for Information Science and Technology (BNRist) and the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
e-mail: gyt@tsinghua.edu.cn

X. Mao
e-mail: maoxh92@sina.com

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*, Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_8

1 Introduction

1.1 Background, Motivation, and Organization

Graph signal processing is a new research direction that has received extensive attention in recent years [1–3]. It mainly analyzes signals living on an irregular domain that is captured by a graph, and has found wide applications in sensor networks [4], machine learning [5], image processing [6], biomedical fields [7], climate analysis [8], et cetera. Existing research topics of graph signal processing include graph filtering [9, 10], graph signal compression [11, 12], graph signal coarsening [13, 14], stationary graph signal processing [15–17], graph signal sampling and reconstruction [18–20], et cetera.

Time-varying graph signal reconstruction is an important topic with practical applications where missing data problem is prevalent. For example, sea surface temperature can provide important information in the study of the earth's climate dynamics. However, the spacial distribution of the sea surface temperature data collected from ships has varied in history due to economic and political changes, such as the opening of new canals and world wars, which significantly raises demands for reconstructing the global sea surface temperature [21]. For another example, in low-cost commodity sensor network, such as air temperature sensor network, lost data are common [22] due to sensor malfunctions or communication failures, which also calls for the reconstruction of time-varying signals.

In this chapter, we study the reconstruction problem of time-varying graph signals. Suppose that a time-varying graph signal is sampled at M consecutive time instances with equal interval, and at each time instance only some of the vertices are observed, i.e., partial values of the graph signal are sampled. The problem is to recover the missing values residing on the unobserved vertices according to the sampled data.

In general, for real-world datasets the static graph signals tend to be smooth with respect to the underlying graph. The reconstruction of smooth graph signals can be formulated as a convex optimization problem [23–25], the objective function of which promotes smoothness of the graph signal and penalizes reconstruction error on the known samples. A kerneled distance penalty term is introduced into the optimization problem in [26], which unifies and subsumes the Tikhonov-regularized graph signal reconstruction schemes in the previous works.

As for time-varying graph signals, a direct assumption is that the signal at each time instant is bandlimited or smooth on graph. Based on this assumption, a distributed reconstruction algorithm of time-varying graph signal is proposed in [27], which performs well in tracking slowly evolving time-varying graph signals. In [28], the authors reconstruct one or multiple smooth graph signals by solving a variation minimization problem. However, these methods do not take full advantage of the temporal correlation of the graph signals.

In order to properly model the structural nature of time-varying graph signals, we introduce the differential smooth property which captures the smoothness of temporal difference signals with respect to the underlying graph. Based on

differential smoothness, we design both batch and online time-varying graph signal reconstruction algorithms. Reconstruction prerequisite on sampling set and reconstruction error bounds of the proposed methods are theoretically analysed. We also highlight the application of the proposed methods in reconstructing spatio-temporal signals. Experiments on different real-world datasets verify the correctness of differential smooth property and the overwhelming performance of the proposed reconstruction methods.

The rest of this chapter is organized as follows. In the later part of this section, some basic conceptions regarding graph signal reconstruction and smooth graph signals are briefly reviewed. In Sect. 2, differential smoothness is introduced. In Sect. 3, two batch reconstruction methods based on differential smoothness are proposed, and their applying prerequisite and reconstruction error bounds are theoretically provided. In Sect. 4, an online reconstruction method is proposed to timely reconstruct the continually arriving time-varying graph signal. Performance analysis on the online reconstruction method is also included in the same section. In Sect. 5, the proposed batch and online reconstruction methods are tested on different real-world datasets and compared with state-of-art reconstruction methods. The chapter is concluded in Sect. 6.

1.2 Basic Concepts

1.2.1 Static Graph Signals Reconstruction

A static graph signal $\mathbf{x} \in \mathbb{R}^N$ is defined as a map on the graph that assigns signal value x_i to the ith vertex. When the aim is to recover the graph signal with only a part of the vertices sampled, smoothness of graph signals is widely applied as the priori information.

The smoothness of graph signals is a qualitative characteristic that expresses how much signal samples vary with respect to the underlying graph [1, 28]. A typical metric measuring the variation of graph signal is of quadratic form as shown below,

$$S(\mathbf{x}) := \|\mathbf{L}^{\frac{1}{2}}\mathbf{x}\|_2^2 = \mathbf{x}^{\mathsf{T}}\mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i} W_{ij}\left(x_j - x_i\right)^2, \tag{1}$$

which is a summation of neighborhood variation over all the vertices. The smaller the value of $S(\mathbf{x})$ is, the smoother the graph signal $\mathbf{x}$ is.

1.2.2 Graph Spectra

Consider an undirected weighted graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V}$ is the set of vertices with $|\mathcal{V}| = N$, $\mathcal{E}$ is the edge set, and the symmetric matrix $\mathbf{W}$ is the weighted adja-

cency matrix. The (i, j)th entry of $\mathbf{W}$, $W_{ij} \in \mathbb{R}^+$, denotes the edge weight between the ith and jth vertices and is a quantitative expression of their underlying relation, such as similarity, dependency, or communication pattern [28]. The graph Laplacian is defined as $\mathbf{L} = \text{diag}(d_1, \cdots, d_N) - \mathbf{W}$, where $d_i = \sum_{j=1}^{N} W_{ij}$ is the degree of the ith vertex.

Since $\mathbf{L}$ is symmetric and semi-positive definite, its singular value decomposition can be denoted as $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\mathsf{T}$. With spectra of $\mathbf{L}$ indicating the frequency, the graph Fourier transform of graph signal $\mathbf{x}$ is defined as

$$\hat{\mathbf{x}} = \mathbf{U}^\mathsf{T}\mathbf{x}.$$

As a generalization of discrete bandlimited graph to the field of graph signal processing, the concept of bandlimited graph signals first appears in [29]. Specifically, a graph signal $\mathbf{x}$ is named as ω-bandlimited graph signal if $\mathbf{x} \in PW_\omega(\mathcal{G})$, where ω denotes the highest graph frequency component of $\mathbf{x}$, and $PW_\omega(\mathcal{G})$ is the ω-bandlimited subspace of graph $\mathcal{G}$, defined as $\text{span}(\mathbf{u}_1, \cdots, \mathbf{u}_{m_\omega})$, with $\mathbf{u}_1, \cdots, \mathbf{u}_{m_\omega}$ denoting the first few eigenvectors of graph Laplacian $\mathbf{L}$, corresponding to the eigenvalues less or equal to ω.

Typically, bandlimited graph signals are always smooth, i.e., yielding smaller $S(\mathbf{x})$. The smoothness of ω-bandlimited graph signal $\mathbf{x}$ can be simply tested by

$$S(\mathbf{x}) = \mathbf{x}\mathbf{L}^\mathsf{T}\mathbf{x} \leq \omega\|\mathbf{x}\|^2.$$

1.2.3 Time-varying Graph Signals Reconstruction

A time-varying graph signal refers to a sequence of static graph signals $\{\mathbf{x}_0, \cdots, \mathbf{x}_T\}$ where $\mathbf{x}_t$ denotes the graph signal at the tth time instant. Sampling of time-varying graph signals denotes that at each time instant only part of the vertices are sampled. According to when the recovery process is conducted, the recovery of time-varying graph signals can be divided into three types: tracking, batch recovery, and online recovery. Tracking of time-varying graph signals [27] refers to the problem that reconstructing $\mathbf{x}_t$ after the observation to $\mathbf{x}_{t-1}$ is completed, that is, only the observations before the tth time instant is available for reconstructing $\mathbf{x}_t$. It is an online process and keeps forecasting the signal at the next time instant. Batch recovery of time-varying graph signals [30] refers to the case that reconstruction is conducted after the sampling at all time instants is finished and all the sampled data can be utilized for reconstruction. Online recovery [30] deals with the case that recovering $\mathbf{x}_t$ immediately after the observation to $\mathbf{x}_t$ is available.

1.2.4 Spatio-temporal Signals Form a Type of Time-varying Graph Signals

Spatio-temporal signals are time-varying graph signals residing on a graph of the observation sites with the edges labelling the geographical adjacency of the observation sites [31]. A typical method to generate such geographical graph is k-nearest neighbor method, that is, an edge is placed between a pair of vertices if and only if one of the two vertices is among the k nearest neighbors of the other.

Concretely speaking, we first compute the distances between any pair of observation sites and achieve the distance matrix $\mathbf{G}$ with the (i, j)th element $g_{i,j}$ indicating the distance between sites i and j. To avoid the connection between far away observation sites, a mask matrix $\mathbf{M}$ is generated based on $\mathbf{G}$. For each site i, we find the k smallest elements on the ith row of $\mathbf{G}$, denoted as $\{g_{i,j}\}_{j\in K_i}$ and set $\{m_{i,j}\}_{j\in K_i}$ as 1 with all other elements in the ith row of $\mathbf{M}$ as 0. Then the mask matrix is symmetrised by taking $\mathbf{M}$ as $\mathbf{M}^{\mathsf{T}}|\mathbf{M}$, with $|$ denotes the element-wise "or" operation. There exists an edge between sites i and j if and only if $m_{i,j}$ equals 1. Till now the underlying graph topology is achieved. To further suppress the influence of mal-connections and reflect the geographical adjacency quantitatively, the weight of each edge is set to be inversely proportional to the square of the distance between the connected two vertices, i.e., the (i, j)th element of the graph adjacency matrix $\mathbf{W}$, $w_{i,j} = m_{i,j}/g_{i,j}^2$.

However, since graph topology is not necessarily based on the spatial positions of the vertices, time-varying graph signals are not all spatio-temporal signals.

2 Differential Smoothness

If all the signals $\mathbf{x}_t, t \in \{1, \cdots, M\}$, are smooth on graph, then $\mathbf{X}$ could be recovered column by column using reconstruction methods of smooth graph signals. However, in many practical applications, $\mathbf{x}_t$ is not smooth enough on graph, which may lead to poor reconstruction quality by the aforementioned methods. Since a time-varying graph signal exhibits correlations both on graph and along the time direction, the reconstruction quality will be significantly improved by adequately exploiting the correlations of the signal from these two aspects.

We find that for most time-varying graph signals obtained from real-world datasets, the difference signal $\mathbf{x}_t - \mathbf{x}_{t-1}$ exhibits smoothness on graph, even if signals $\mathbf{x}_t, t \in \{1, \cdots, M\}$ are not smooth on graph. Take the three real-world datasets that we will use in the experiments as examples. The smoothness of these signals is compared in Table 1, where SST, SLP, and PM2.5 denote, respectively, the sea surface temperature dataset [32], the global sea-level pressure dataset [33], and the daily mean PM2.5 concentration of California [34]. More information on these datasets could be found in Sect. 5. Let $\mathbf{L}$ denote the Laplacian matrix of this graph. Then we introduce the temporal differential operator $\mathbf{D}$ as follows,

Table 1 Smoothness comparison

Datasets	SST	SLP	PM2.5
$\|\mathbf{XD}\|_F^2$	3.8×10^4	1.4×10^5	3.8×10^5
$S(\mathbf{X})$	1.6×10^3	230	210
$S(\mathbf{XD})$	71	91	170

$$\mathbf{D} = \begin{bmatrix} -1 & & & \\ 1 & -1 & & \\ & 1 & \ddots & \\ & & \ddots & -1 \\ & & & 1 \end{bmatrix}_{M \times (M-1).} \tag{2}$$

We have the temporal difference of $\mathbf{X}$ equal to $\mathbf{XD} = [\mathbf{x}_1 - \mathbf{x}_0, \mathbf{x}_2 - \mathbf{x}_1, \cdots, \mathbf{x}_M - \mathbf{x}_{M-1}]$. Then another way of saying that $\mathbf{X}$ is differentially smooth is that the mathematical expression (3) holds true for some positive parameter δ.

$$S(\mathbf{XD}) := \sum_{t=1}^{M} S(\mathbf{x}_t - \mathbf{x}_{t-1}) \overset{(a)}{=} \operatorname{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right) \leq \delta, \tag{3}$$

where $\operatorname{tr}(\cdot)$ denotes taking the trace of a matrix, and (a) follows from the definition of the smoothness metric $S(\cdot)$ in (1) and (2). To be noted, $S(\mathbf{XD})$ measures the variation of the time difference of $\mathbf{X}$ with respect to the graph topology.

It could be readily read that the temporal difference signals exhibit better smoothness property compared with the original signals. To be noted, since the maximum singular value of $\mathbf{D}$ is larger than one, the observed decrease in the quantity $S(\mathbf{XD})$ is due to the inherent data structure and not due to that the operator $\mathbf{D}$ reduces the magnitude of $\mathbf{X}$.

3 Batch Reconstruction Based on Differential Smoothness

The sampling of a time-varying graph signal $\mathbf{X}$ can be represented as

$$\mathbf{Y} = \mathbf{J} \circ \mathbf{X} + \mathbf{V}, \tag{4}$$

where $\mathbf{V}$ is the additive Gaussian white noise, $\circ$ is the Hadamard (element-wise) product, and $\mathbf{J} \in \{0, 1\}^{N \times M}$ is the sampling operator defined as

$$J_{u,t} = \begin{cases} 1 & u \in \mathscr{S}_t; \\ 0 & u \notin \mathscr{S}_t, \end{cases} \tag{5}$$

where $\mathscr{S}_t$ denotes the set of vertices sampled at time instant t.

In this section, we focus on the case where there is a computing center to collect the sampled data $\mathbf{Y}$ of all the vertices, and the graph Laplacian matrix $\mathbf{L}$ is also known. Two batch reconstruction methods for time-varying graph signals based on the proposed differential smoothness are proposed and analyzed in the following subsections.

3.1 Vanilla Batch Reconstruction Algorithm

Suppose that the time-varying graph signal to be reconstructed, $\mathbf{X}$, is differential smooth, the reconstruction problem can be formulated as

$$\min_{\mathbf{X}} \frac{1}{2}\|\mathbf{J}\circ\mathbf{X}-\mathbf{Y}\|_F^2 + \frac{\lambda}{2}\mathrm{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right) = f_u(\mathbf{X}), \tag{6}$$

where λ is the regularization parameter, balancing the coincidence with measurements and the fidelity of the differential smoothness. Problem (6) can be rewritten in an equivalent form,

$$\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{Q}\left[\mathbf{z}-\mathrm{vec}(\mathbf{Y})\right]\|_2^2 + \frac{\lambda}{2}\mathbf{z}^\mathsf{T}\left[\left(\mathbf{D}\mathbf{D}^\mathsf{T}\right)\otimes\mathbf{L}\right]\mathbf{z} = \tilde{f_u}(\mathbf{z}), \tag{7}$$

where $\mathbf{Q} = \mathrm{diag}(\mathrm{vec}(\mathbf{J})) \in \mathbb{R}^{MN\times MN}$, and $\mathbf{z} = \mathrm{vec}(\mathbf{X})$.

For better understanding problem (6), we study the uniqueness of solution to problem (6) in Theorem 1.

Theorem 1 *[30] When the following two conditions are simultaneously satisfied by the sampling operator* $\mathbf{J}$*, the optimal solution to problem* (6) *is unique.*

1. *For any* $n \in \{1,\cdots,N\}$*, there exists* $m \in \{1,\cdots,M\}$*, such that* $J_{n,m} = 1$.
2. *There is a fiducial time* $m_0 \in \{1,\cdots,M\}$*, such that for any* $m \in \{1,\cdots,M\}$, $m \neq m_0$*, there exists a vertex* $n_m \in \{1,\cdots,N\}$ *satisfying that* $J_{n_m,m_0} = J_{n_m,m} = 1$.

Remark 1 Theorem 1 gives properties that the sampling operator should satisfy to ensure the uniqueness of the solution to problem (6), so it suggests how to design a sampling operator. The first condition in the theorem means that every vertex should be sampled at least once. It makes sense, in that if there is a vertex never sampled, then adding a constant to the signal on that vertex will not change the value of the cost function.

According to the second condition, there should be a time m_0, such that for any other time $m \neq m_0$, there is a vertex sampled both at m_0 and m. The time m_0 serves as a fiducial time, so that the temporal difference becomes valid.

It should be noted that the two conditions in Theorem 1 are sufficient but not necessary.

According to the form of function $\tilde{f_u}$, problem (6) is to minimize a quadratic function, hence there is a closed form solution to it. When the conditions in Theorem 1 is satisfied, the unique optimal solution $\mathbf{X}^*$ could be obtained via

$$\mathrm{vec}(\mathbf{X}^*) = \left[\mathbf{Q} + \lambda\left(\mathbf{D}\mathbf{D}^{\mathrm{T}}\right) \otimes \mathbf{L}\right]^{-1} \mathbf{Q}\mathrm{vec}(\mathbf{Y}). \tag{8}$$

However, (8) involves calculating the inverse of a matrix of size $MN \times MN$, which is of high computational complexity, especially when we deal with a long-term tracking problem over a large-scale vertex set. Hence, we propose to solve problem (6) by conjugate gradient method.

In each iteration, the algorithm mainly consists of two steps: the determination of stepsize and the update of the next search direction. Denoting the search direction of the kth step as $\Delta\mathbf{X}^k$, the optimal stepsize τ of the kth step is decided by the line minimization rule

$$\min_{\tau} \quad f_u(\mathbf{X}^k + \tau\Delta\mathbf{X}^k).$$

By taking derivative, we have

$$0 = \frac{\partial f_u(\mathbf{X}^k + \tau\Delta\mathbf{X}^k)}{\partial\tau} = \langle\Delta\mathbf{X}^k, \nabla f_u(\mathbf{X}^k + \tau\Delta\mathbf{X}^k)\rangle, \tag{9}$$

where the gradient of function $f_u(\mathbf{X})$ is

$$\nabla f_u(\mathbf{X}) = \mathbf{J} \circ \mathbf{X} - \mathbf{Y} + \lambda\mathbf{L}\mathbf{X}\mathbf{D}\mathbf{D}^{\mathrm{T}}. \tag{10}$$

The optimal stepsize is the solution of equation (9)

$$\tau = -\frac{\langle\Delta\mathbf{X}^k, \nabla f_u\left(\mathbf{X}^k\right)\rangle}{\langle\Delta\mathbf{X}^k, \nabla f_u\left(\Delta\mathbf{X}^k\right) + \mathbf{Y}\rangle}.$$

According to the optimal stepsize τ, the iterative procedure is denoted as

$$\mathbf{X}^{k+1} = \mathbf{X}^k + \tau\Delta\mathbf{X}^k.$$

The search direction of the $(k+1)$th step $\Delta\mathbf{X}^{k+1}$ is the linear combination of the gradient at the $(k+1)$th step and the search direction at the kth step

$$\Delta\mathbf{X}^{k+1} = -\nabla f_u\left(\mathbf{X}^{k+1}\right) + \gamma\Delta\mathbf{X}^k,$$

where $\gamma = \dfrac{||\nabla f_u\left(\mathbf{X}^{k+1}\right)||_F^2}{||\nabla f_u\left(\mathbf{X}^k\right)||_F^2}$.

The proposed method for solving problem (6) is listed in Table 2. According to [35], this method can find the global optimal solution of problem (6) after at most MN iterations. When MN is large, after fewer iterations a sufficiently accurate solution can be obtained.

Table 2 Batch reconstruction method with noise

Input	$\mathbf{Y}$:	sampled data, $\mathbf{J}$: sampling operator, $\mathbf{L}$: Laplacian matrix,
	λ:	regularization parameter, K: maximum number of iterations, δ : tolerance
Output	$\mathbf{X}^k$:	reconstructed signal

Initialization: $\mathbf{X}^0 = 0;\quad \Delta\mathbf{X}^0 = -\nabla f_u\left(\mathbf{X}^0\right);$

Iteration:

(1) **Stepsize decision:**

$$\tau = -\frac{\langle \Delta\mathbf{X}^k, \nabla f_u\left(\mathbf{X}^k\right)\rangle}{\langle \Delta\mathbf{X}^k, \nabla f_u\left(\Delta\mathbf{X}^k\right) + \mathbf{Y}\rangle};$$

(2) **Search direction updating:**

$$\mathbf{X}^{k+1} = \mathbf{X}^k + \tau\Delta\mathbf{X}^k;$$
$$\gamma = ||\nabla f_u\left(\mathbf{X}^{k+1}\right)||_F^2 / ||\nabla f_u\left(\mathbf{X}^k\right)||_F^2;$$
$$\Delta\mathbf{X}^{k+1} = -\nabla f_u\left(\mathbf{X}^{k+1}\right) + \gamma\Delta\mathbf{X}^k;$$
$$k = k + 1;$$

(3) **Repeat** steps 1 and 2 until $k = K$ or $\|\Delta\mathbf{X}^k\|_F \leq \delta$.

3.2 Batch Reconstruction Inducing Low-rank Property

Recall Sect. 1.2.4, spatio-temporal signals form a special type of time-varying graph signals. With the spatial structure of the set of observation sites fixed, the spatio-temporal signals are usually correlated in a global sense [28, 36]. Such global correlation can be demonstrated as the temporal sequence $\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_{M-1}$ generated from limited patterns, or in other words, the spatio-temporal signal $\mathbf{X}$ being approximately low-rank [37, 38].

Then jointly applying the differential smoothness and low-rank property, the spatio-temporal signal recovery problem can be formulated as an optimization problem

$$\begin{aligned} \min_{\mathbf{X}}\ & \operatorname{rank}(\mathbf{X}) + \lambda \operatorname{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right) \\ \text{s.t.}\ & \|\mathbf{J}\circ\mathbf{X} - \mathbf{Y}\|_F^2 \leq \epsilon, \end{aligned} \tag{11}$$

where λ is the regularization parameter. Because matrix rank minimization is NP-hard, to solve it efficiently, we provide a convex relaxation to (11)

$$\min_{\mathbf{X}}\ \frac{1}{2}\|\mathbf{J}\circ\mathbf{X} - \mathbf{Y}\|_F^2 + \frac{\alpha}{2}\operatorname{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right) + \mu\|\mathbf{X}\|_*, \tag{12}$$

where α and μ are regularization parameters. In (12), $\operatorname{rank}(\mathbf{X})$ is replaced by nuclear norm $\|\mathbf{X}\|_*$, defined as the sum of the singular values of $\mathbf{X}$, which still promotes low rank [39, 40]. The convexity of (12) follows from the fact that $\|\mathbf{X}\|_*$ is the convex envelope of $\operatorname{rank}(\mathbf{X})$ over set $\{\mathbf{X} \in \mathbb{R}^{N\times M} : \|\mathbf{X}\|_2 \leq 1\}$ [40].

Noticing that the first two terms of (12) are differentiable, and the third term is proximable, we choose to solve (12) by applying ADMM [41, 42], which takes advantage of both the decomposability and the superior convergence properties of the method of multipliers. Firstly, we introduce an equivalent splitting version of (12) as follows

$$\min_{\mathbf{X},\mathbf{Z}} \frac{1}{2}\|\mathbf{J}\circ\mathbf{X}-\mathbf{Y}\|_F^2+\frac{\alpha}{2}\mathrm{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right)+\mu\|\mathbf{Z}\|_* \quad \text{s.t.} \quad \mathbf{X}=\mathbf{Z}. \tag{13}$$

Such splitting step followed by an augmented Lagrangian method to handle the linear equality constraint is what constitutes ADMM. The augmented Lagrangian of (13) is

$$\mathscr{L}_\rho(\mathbf{X},\mathbf{Z},\mathbf{P})=\frac{1}{2}\|\mathbf{J}\circ\mathbf{X}-\mathbf{Y}\|_F^2+\frac{\alpha}{2}\mathrm{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right)+\mu\|\mathbf{Z}\|_*+\langle\mathbf{P},\mathbf{X}-\mathbf{Z}\rangle+\frac{\rho}{2}\|\mathbf{X}-\mathbf{Z}\|_F^2, \tag{14}$$

where $\mathbf{P}$ is the Lagrange multiplier, ρ is the penalty parameter, and $\langle\cdot,\cdot\rangle$ denotes the inner product of matrices. As proved in [41, 43], ADMM finds a saddle point to (14) with the following iterative scheme

$$\mathbf{X}^{k+1}=\arg\min_{\mathbf{X}}\ \mathscr{L}_\rho\left(\mathbf{X},\mathbf{Z}^k,\mathbf{P}^k\right), \tag{15}$$

$$\mathbf{Z}^{k+1}=\arg\min_{\mathbf{Z}}\ \mathscr{L}_\rho\left(\mathbf{X}^{k+1},\mathbf{Z},\mathbf{P}^k\right), \tag{16}$$

$$\mathbf{P}^{k+1}=\mathbf{P}^k+\rho\left(\mathbf{X}^{k+1}-\mathbf{Z}^{k+1}\right). \tag{17}$$

Subproblem (15) is equivalent to

$$\mathbf{X}^{k+1}=\arg\min_{\mathbf{X}}\ \frac{1}{2}\|\mathbf{J}\circ\mathbf{X}-\mathbf{Y}\|_F^2+\frac{\alpha}{2}\mathrm{tr}\left(\mathbf{D}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{L}\mathbf{X}\mathbf{D}\right)+\frac{\rho}{2}\|\mathbf{X}-\mathbf{Z}^k+\frac{\mathbf{P}^k}{\rho}\|_F^2. \tag{18}$$

Denoting the above objective function as $f(\mathbf{X})$, then its gradient is calculated as

$$\nabla f(\mathbf{X})=\mathbf{J}\circ\mathbf{X}-\mathbf{Y}+\alpha\mathbf{L}\mathbf{X}\mathbf{D}\mathbf{D}^\mathsf{T}+\rho(\mathbf{X}-\mathbf{Z}^k)+\mathbf{P}^k. \tag{19}$$

By solving the linear equation $\nabla f(\mathbf{X})=\mathbf{0}$, we achieve the closed-form solution to (18) as in (20).

$$\mathrm{vec}(\mathbf{X}^{k+1})=\left(\tilde{\mathbf{J}}+\alpha\tilde{\mathbf{L}}\tilde{\mathbf{D}}+\rho\tilde{\mathbf{I}}\right)^{-1}\mathrm{vec}\left(\rho\mathbf{Z}^k+\mathbf{Y}-\mathbf{P}^k\right), \tag{20}$$

where vec(·) denotes the vectorization operator stacking the columns of a matrix into a long vector, and the matrices with tilde are all square matrices of $MN \times MN$ dimensional. Specifically speaking, $\tilde{\mathbf{I}}$ denotes the MN dimensional identity matrix, $\tilde{\mathbf{J}} = \text{diag}(\text{vec}(\mathbf{J}))$, $\tilde{\mathbf{L}} = \mathbf{I}_M \otimes \mathbf{L}$, and $\tilde{\mathbf{D}} = \left(\mathbf{D}\mathbf{D}^{\mathrm{T}}\right) \otimes \mathbf{I}_N$, with $\mathbf{I}_n$ denoting the n dimensional identity matrix and $\otimes$ denoting the Kronecker product. To be noted, $\tilde{\mathbf{J}}$, $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{D}}$ are all positive semi-definite matrices.

However, the calculation of (20) involves the inversion of an $MN \times MN$ dimensional matrix, which is computationally expensive. To reduce the computation complexity, conjugate gradient method can be applied to solve (18) iteratively. The detailed algorithm for solving problem (18) can be achieved by simply modifying Table 2 with $\mathbf{Y}$ substituted by $\mathbf{Y} + \rho\mathbf{Z}^k - \mathbf{P}^k$.

Subproblem (16) is equivalent to

$$\mathbf{Z}^{k+1} = \arg\min_{\mathbf{Z}} \frac{1}{2}\|\mathbf{Z} - \mathbf{X}^{k+1} - \frac{\mathbf{P}^k}{\rho}\|_F^2 + \frac{\mu}{\rho}\|\mathbf{Z}\|_*, \tag{21}$$

which has a closed-form solution

$$\mathbf{Z}^{k+1} = \text{SVT}_{\frac{\mu}{\rho}}\left(\mathbf{X}^{k+1} + \frac{\mathbf{P}^k}{\rho}\right), \tag{22}$$

where SVT is the singular value thresholding operator defined as

$$\text{SVT}_\tau(\mathbf{X}) = \mathbf{U}\mathbf{\Lambda}_\tau(\mathbf{\Sigma})\mathbf{V}^{\mathrm{T}}, \tag{23}$$

where $\mathbf{U}, \mathbf{V}, \mathbf{\Sigma}$ are from the singular value decomposition of $\mathbf{X}$, i.e., $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$, and

$$\mathbf{\Lambda}_\tau(x) := \text{sign}(x)\max(|x| - \tau, 0) \tag{24}$$

is the soft thresholding operator defined for $\tau \in \mathbb{R}^+$.

The detailed algorithm for solving problem (12) is listed in Table 3. The stopping criterion could be either a maximum number of iterations, or the change of $\mathbf{X}^k$ less than a threshold. According to [43] and noticing that the objective function of (13) is closed, proper, and convex, when $\mathbf{X}^k$ is updated with (20), we have that the sequence $\{\mathbf{X}^k, \mathbf{Z}^k, \mathbf{P}^k\}_{k\geq 0}$ generated by LRDS converges to a Karush–Kuhn–Tucker (KKT) point of (14), denoted as $\{\mathbf{X}^*, \mathbf{Z}^*, \mathbf{P}^*\}$.

Let $\mathbf{X}$ denote the ground truth spatio-temporal signal to be recovered. A reconstruction error bound of the proposed LRDS is provided in Proposition 1.

Proposition 1 ([31]) *If the following holds true for any* $\mathbf{M} \in \mathbb{R}^{N\times M}$ *with a constant* $\gamma \in [0, 1)$,

$$\|\mathbf{M} - \frac{1}{4}\mathbf{J} \circ \mathbf{M} - \frac{\alpha}{4}\mathbf{L}\mathbf{M}\mathbf{D}\mathbf{D}^{\mathrm{T}}\|_F \leq \gamma\|\mathbf{M}\|_F, \tag{25}$$

then the recovery error of the proposed method can be upper bounded as shown in (26),

Table 3 The procedure of LRDS

Input: $\mathbf{Y}$, $\mathbf{J}$, $\mathbf{L}$, α, μ, ρ, stopping criterion.
Output: $\mathbf{X}^k$: recovered signal
Initialization: $\mathbf{X}^0 = \mathbf{Y},\ \mathbf{Z}^0 = \mathbf{Y},\ \mathbf{P}^0 = \mathbf{0},\ k = 0$
Repeat:
Update $\mathbf{X}^{k+1}$ by Table 2 with $\mathbf{Y}$ substituted by $\mathbf{Y} + \rho\mathbf{Z}^k - \mathbf{P}^k$ or $\mathrm{vec}(\mathbf{X}^{k+1}) = \left(\tilde{\mathbf{J}} + \alpha\tilde{\mathbf{L}}\tilde{\mathbf{D}} + \rho\tilde{\mathbf{I}}\right)^{-1} \mathrm{vec}\left(\rho\mathbf{Z}^k + \mathbf{Y} - \mathbf{P}^k\right)$;
Update $\mathbf{Z}^{k+1}$ by $\mathbf{Z}^{k+1} = \mathrm{SVT}_{\frac{\mu}{\rho}}\left(\mathbf{X}^{k+1} + \frac{\mathbf{P}^k}{\rho}\right)$;
Update $\mathbf{P}^{k+1}$ by $\mathbf{P}^{k+1} = \mathbf{P}^k + \rho\left(\mathbf{X}^{k+1} - \mathbf{Z}^{k+1}\right)$;
$k = k + 1$;
Until: Stopping criterion satisfied.

$$\|\mathbf{X}^* - \mathbf{X}\| \leq \frac{\|\mathbf{V}\|_F + \mu\sqrt{2\mathrm{rank}(\mathbf{X})} + \alpha\|\mathbf{LXDD}^{\mathsf{T}}\|_F}{4(1-\gamma)}, \tag{26}$$

where $\mathbf{V}$ *is the observation noise in (4).*

For better understanding of the condition in Proposition 1, a sufficient condition for (25) is provided in Proposition 2.

Proposition 2 ([31]) *Assume that the generated graph is connected and the regularization parameter* α *in (12) satisfies* $\alpha \in (0, \frac{3}{4\|\mathbf{L}\|_2}]$. *Let* $S := \{(n, m) : J_{n,m} = 1\}$ *denote the sampling set. Once there exists a split of* $S = S_1 \cup S_2$, $S_1 \cap S_2 = \phi$, *such that the following two criteria hold true:*

1. *for each time instant* $m \in \{1, \cdots, M\}$, *there exists at least one sampling index of the form* $(\cdot, m)$ *belonging to* S_1;
2. *for each vertex* $n \in \{1, \cdots, N\}$, *there exists at least one sampling index of the form* $(n, \cdot)$ *belonging to* S_2,

then we claim that there exists a constant $\gamma \in [0, 1)$ *such that (25) is satisfied by any matrix* $\mathbf{M} \in \mathbb{R}^{N\times M}$.

Remark 2 Proposition 2 provides a sufficient but not necessary condition for (25). As shown in Proposition 2, such sufficient condition is tightly related to the sampling strategy $\mathbf{J}$. The only constraint on the graph topology is that it is connected. The proposed condition on $\mathbf{J}$ can be easily satisfied. Specifically speaking, to sample a spatio-temporal signals with N observation sites over M time instants, there exists a sampling strategy taking only $M + N$ samples that satisfies the condition on $\mathbf{J}$ stated in Proposition 2.

4 Online Reconstruction Based on Differential Smoothness

4.1 Online Centralized Reconstruction

The batch reconstruction of time-varying graph signals requires to obtain the sampled values within a long time before reconstructing them all together, which leads to high reconstruction delay and high computational complexity. Considering that among the prevalent practical applications, real-time analysis is of vital importance, so we focus on online reconstruction method in this section. Furthermore, we deal with the fully distributed scenario, where no computing center exists, and all communications are limited in the one-hop neighborhood. In such scenario, we propose a distributed reconstruction method to solve the online reconstruction problem.

The aim of online reconstruction of time-varying graph signals is to recover the current unsampled values according to the sampled data of current and previous time. In order to alleviate the storage burden, we only preserve the current sampled data and the previous reconstructed signal.

The online reconstruction of time-varying graph signals can be formulated as an unconstrained optimization problem

$$\min_{\mathbf{x}_t} \frac{1}{2}\|\mathbf{j}_t \circ \mathbf{x}_t - \mathbf{y}_t\|_2^2 + \frac{\lambda}{2}\left(\mathbf{x}_t - \hat{\mathbf{x}}_{t-1}\right)^{\mathrm{T}} \mathbf{L}\left(\mathbf{x}_t - \hat{\mathbf{x}}_{t-1}\right). \tag{27}$$

Denote the above objective function as $f_o(\mathbf{x}_t)$, where $\mathbf{x}_t$ is the current signal to be reconstructed, $\hat{\mathbf{x}}_{t-1}$ is the reconstructed signal of the previous time, $\mathbf{j}_t$ is the current sampling operator, $\mathbf{y}_t$ is the current sampled data, and λ is the regularization parameter.

The gradient of function $f_o(\mathbf{x}_t)$ is

$$\nabla f_o(\mathbf{x}_t) = \mathbf{j}_t \circ \mathbf{x}_t - \mathbf{y}_t + \lambda \mathbf{L}(\mathbf{x}_t - \hat{\mathbf{x}}_{t-1}). \tag{28}$$

Letting the above gradient equal zero, we get the closed form solution of problem (27)

$$\hat{\mathbf{x}}_t = (\lambda \mathbf{L} + \mathbf{J}_t)^{-1}\left(\lambda \mathbf{L}\hat{\mathbf{x}}_{t-1} + \mathbf{y}_t\right), \tag{29}$$

where $\mathbf{J}_t = \mathrm{diag}\,(\mathbf{j}_t)$. With the reconstructed signal at the $(t-1)$th time instant close to the ground truth $\mathbf{x}_{t-1}$, we show in Proposition 3 that the estimation error of the tth time instant is well bounded.

Proposition 3 ([30]) *Assume that the difference signal $\mathbf{g}_t := \mathbf{x}_t - \mathbf{x}_{t-1}$ is ϵ-smooth with respect to graph $\mathcal{G}$, i.e. $S(\mathbf{g}_t) \leq \epsilon$, and there is at least one vertex sampled at the tth time instant. With the estimation error of the $(t-1)$th time instant satisfying $\|\hat{\mathbf{x}}_{t-1} - \mathbf{x}_{t-1}\|_2 \leq \delta_{t-1}$, and the additive noise at the tth time instant $\|\mathbf{v}_t\|_2 \leq \sigma_t$, we have*

$$\|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2 \leq \frac{2\sqrt{2}\theta_t}{\lambda_{\min}} := \delta_t, \tag{30}$$

$$\theta_t = \sqrt{\sigma_t^2 + \lambda(\sqrt{\epsilon} + s_{\max}\delta_{t-1})^2}, \tag{31}$$

where λ_{min} *denotes the smallest singular value of matrix* $\mathbf{J}_t + \sqrt{\lambda}\mathbf{L}^{1/2}$, *and* $s_{\max}$ *denotes the largest singular value of* $\mathbf{L}^{1/2}$.

4.2 Online Distributed Reconstruction

If the graph has too many vertices, the matrix inversion in (29) will be hard to calculate. In such case, problem (27) can be solved by a gradient descent method

$$\mathbf{x}_t^{k+1} = \mathbf{x}_t^k - \mu \nabla f_o(\mathbf{x}_t^k), \tag{32}$$

where $\mathbf{x}_t^k$ is the iterative value of vector $\mathbf{x}_t$ at the kth step, and μ is the stepsize. Because problem (27) is convex, $\mathbf{x}_t^k$ will converge to the global optimal solution $\hat{\mathbf{x}}_t$ after sufficient iterations.

In this part, we propose a distributed reconstruction method based on a distributed calculation of the gradient in (32).

Denoting $\mathbf{d}_t^k = \mathbf{x}_t^k - \hat{\mathbf{x}}_{t-1}$ and plugging (28) into (32), we have

$$\mathbf{x}_t^{k+1} = \mathbf{x}_t^k - \mu\left(\mathbf{j}_t \circ \mathbf{x}_t^k - \mathbf{y}_t\right) - \mu\lambda\mathbf{L}\mathbf{d}_t^k. \tag{33}$$

Due to the local property of the graph Laplacian $\mathbf{L}$, the update in (33) can be implemented locally. The iterative formula of sampled vertex s at time instant t is

$$x_t^{k+1}(s) = (1-\mu)x_t^k(s) + \mu y_t(s) - \mu\lambda\sum_{j\in N(s)} W_{sj}\left[d_t^k(s) - d_t^k(j)\right], \tag{34}$$

where W_{sj} is the element of the weighted matrix $\mathbf{W}$, and $N(s)$ is the set of the neighbors of vertex s. The update of $x_t^{k+1}(s)$ is composed of two parts. One is the new information from the sampled value of vertex s, and the other is the value change of its neighboring vertices. The iterative formula of unsampled vertex u at time instant t is

$$x_t^{k+1}(u) = x_t^k(u) - \mu\lambda \sum_{j\in N(u)} W_{uj}\left[d_t^k(u) - d_t^k(j)\right]. \tag{35}$$

The update of $x_t^{k+1}(u)$ is from the value change on its neighboring vertices. The overall online distributed reconstruction method is presented in Table 4.

Table 4 Online distributed method at time instant t

Input	$\mathbf{y}_t$:	sampled data at time instant t, $\hat{\mathbf{x}}_{t-1}$: the recovered signal at time instant $t-1$
	$\mathbf{j}_t$:	sampling operator at time instant t, $\mathbf{W}$: Adjacency matrix, λ : regularization parameter
	μ :	stepsize, K : maximum number of iterations
Output	$\hat{\mathbf{x}}_t$:	reconstructed signal

Initialization: $\mathbf{x}_t^0 = \hat{\mathbf{x}}_{t-1}$;

Iteration:

(1) **Distributed gradient descent:**

For each vertex $s \in \{1, \cdots, N\}$ do:

$d_t^k(s) = x_t^k(s) - \hat{x}_{t-1}(s)$;

if $J_{s,t} = 1$

$x_t^{k+1}(s) = (1-\mu)x_t^k(s) + \mu y_t(s)$

$\quad -\mu\lambda \sum_{j\in N(s)} W_{sj}\left(d_t^k(s) - d_t^k(j)\right)$;

else

$x_t^{k+1}(s) = x_t^k(s) - \mu\lambda \sum_{j\in N(s)} W_{sj}\left(d_t^k(s) - d_t^k(j)\right)$;

$k = k+1$;

(2) **Repeat** step 1 until $k = K$.

5 Numerical Experiments

5.1 Datasets and Basic Experimental Settings

In the experiments we test the proposed algorithms on three real-world datasets which are the sea surface temperature [32], the global sea-level pressure [33], and the daily mean PM2.5 concentration of California [34], respectively. The detailed information about these datasets are listed as below.

1. The sea surface temperature data [32] is collected monthly from 1870 to 2014 with a spatial resolution of 1° latitude $\times$1° longitude global grid. We randomly select 100 locations on the Pacific for simulation, with the observing duration of 500 months. The selected data ranges from 0.02 to 30.72 °C, and the mean temperature is 19.14 °C.
2. The global sea-level pressure data [33] is collected from 1948 to 2010. The spatial resolution is 2.5° latitude $\times$2.5° longitude global grid, and the temporal resolution is five days. We randomly select 500 locations worldwide over a time period of 600. The selected data ranges from 96.22 to 110.06 kPa, and the average is 101.22 kPa.
3. The California daily mean PM2.5 concentration data [34] are collected from 93 observation sites over 200 days starting from January 1, 2015, and the size of the data is 93 $\times$ 200. Not all of these sites collected valid data everyday, and the percentages of valid data collected each day roughly range from 90 to 45. The valid data range from 0.1 to 102.7 μg/m^3.

In each experiment, simple random sampling scheme is adopted to select the sampled data. For example, when the sampling rate is 40%, it means that 40% of the vertices are randomly sampled at each time instant, and the goal is to reconstruct the whole signal according to the sampled data. In our proposed methods, the maximum number of iterations K is 10^4, the tolerance δ is 10^{-6}, and the stepsize μ of the online distributed method is taken as 10^{-2}. The parameters of all the methods are scanned to generate best performance. [1]

5.2 *Batch Reconstruction Algorithms*

In the following experiments, the proposed two batch reconstruction methods, including the vanilla Batch Reconstruction of Time-Varying Graph Signals (BR-TVGS) introduced in Sect. 3.1 and Low-Rank Differential Smoothness based batch reconstruction (LRDS) introduced in Sect. 3.2 are compared with Natural Neighbor Interpolation (NNI) [44], Fixed Point Continuation with Approximate SVD (FPCA) [39], Graph signal Matrix Completion via total variation Regularization (GMCR) [28] and Spatio-Temporal constrained Low Rank Matrix Approximation (ST-LRMA) [45]. Among them, NNI utilizes the spatial and temporal smooth property of the time-varying graph signal to be reconstructed; FPCA is a low-rank matrix reconstruction method; GMCR assumes the spatial smoothness and low-rank property; ST-LRMA is based on low-rank property, spatial smoothness and local smoothness of the signal to be reconstructed. Both GMCR and ST-LRMA apply the spatial smooth property by solving optimization problems that induces smoothness with respect to the underlying graph topology.

In order to provide intuitive understanding to the algorithms' performance, we use mean absolute error (MAE) to evaluate the recovery result, denoted as $\text{MAE} = \|\mathbf{x}^* - \mathbf{x}\|_1 / N_x$, where $\mathbf{x}$ is the ground true signal, $\mathbf{x}^*$ is the recovered signal, and N_x is the length of the signals.

5.2.1 The Sea Surface Temperature Dataset

We test the performance of all the methods under different sampling rates, and the results are displayed in Fig. 1. One can read from the results that the recovery error of all the methods decreases with the increasing of sampling rate, and the proposed LRDS is always superior to the others. Except for LRDS, BR-TVGS performs the best. This implies that the low-rank inducing term introduced in LRDS do improve the performance of BR-TVGS. Besides, although merely applying the differential smoothness, BR-TVGS shows competitive reconstruction performance.

[1] The MATLAB codes for the proposed methods and all experiments are available at http://gu.ee.tsinghua.edu.cn/codes/Timevarying_GS_Reconstruction.zip.

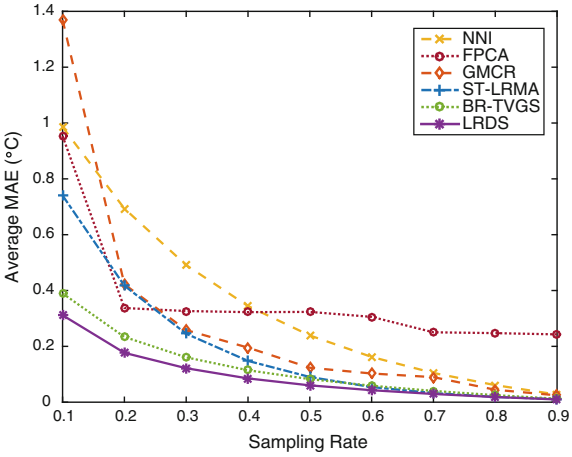


Fig. 1 The sea surface temperature dataset: the average MAEs under different sampling rates

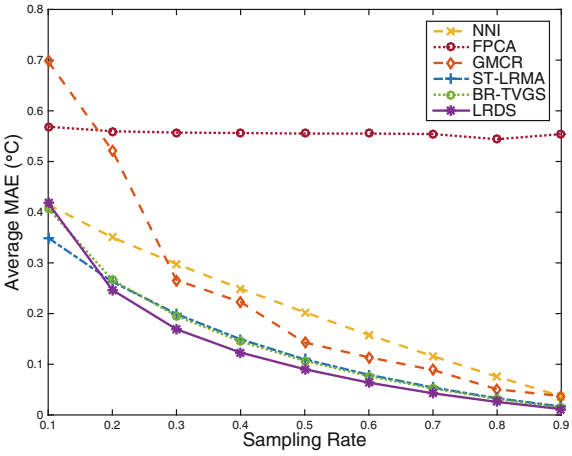


Fig. 2 The global sea-level pressure dataset: the average MAEs under different sampling rates

5.2.2 The Global Sea-Level Pressure Dataset

The average MAEs for all the methods under different sampling rates are displayed in Fig. 2. We can draw a similar conclusion as the previous experiment that the proposed method LRDS is superior to the others. Except for LRDS, ST-LRMA and the proposed BR-TVGS show similar performance and outperform all the other methods.

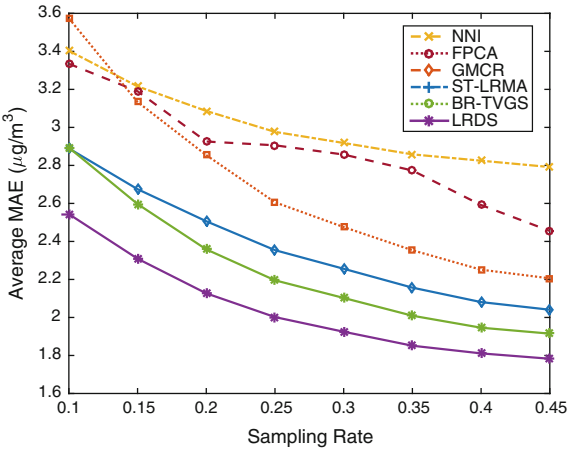


Fig. 3 Mean reconstruction error under different sampling rates on daily mean PM2.5 concentration data, where the mean is taken over both time, all the vertices and 50 independent trials

5.2.3 The Daily Mean PM2.5 Concentration Dataset

The average MAEs for all the methods under different sampling rates are displayed in Fig. 3. One can read that both the proposed batch reconstruction methods overwhelm all the other methods, with LRDS delivering the best reconstruction performance. This implies that differential-smoothness fits this dataset well.

5.3 Online Reconstruction Alogrithm

In this section, the online distributed method proposed in Sect. 4.2 is compared with natural neighbor interpolation [44], low-rank matrix completion [39], the smooth graph signal reconstruction method [23] which is called graph regularization, the Tikhonov-regularization method [46] also referred to as graph-time Tikhonov and the proposed vanilla batch reconstruction method.

To evaluate the reconstruction performance of online reconstruction method, the reconstruction result is evaluated by the root-mean-square error (RMSE)

$$\text{RMSE} = \frac{\|\hat{\mathbf{x}} - \mathbf{x}^*\|_2}{\sqrt{N_x}}, \tag{36}$$

where $\mathbf{x}$ is the ground truth, $\hat{\mathbf{x}}^*$ is the recovered signal, and N_x is the length of the signals. When the signal is in matrix form, it is vectorized to fit (36).

Due to the presence of ℓ_2 norm, compared with MAE, RMSE is more sensitive to outliers. This is because that RMSE^2 is proportional to the square sum of absolute differences, which can be seen as putting more weight on the larger differences.

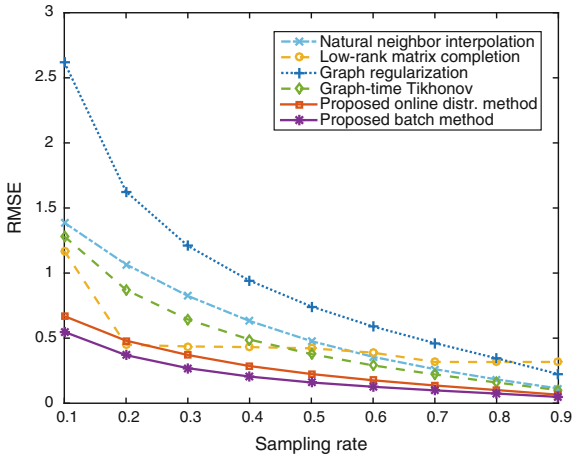


Fig. 4 The sea surface temperature dataset: The reconstruction error under different sampling rate. The RMSE of each method is an average of 50 tests

Whereas MAE puts equal weights to the differences. To test the robustness of the proposed online reconstruction method to outliers, we utilize RMSE as the measuring criterion.

5.3.1 The Sea Surface Temperature Dataset

The performance of all the methods under different sampling rate is displayed in Fig. 4. One can read from the results that the performance of the proposed batch method is better than all the comparison methods, and the performance of the proposed online distributed method is slightly worse than the proposed batch method.

5.3.2 The Global Sea-Level Pressure Dataset

The performance of all the methods under different sampling rate is shown in Fig. 5. As can be seen, the performance of the proposed methods is better than all the comparison methods. There are two main reasons for the improvement of the proposed methods. Firstly, we construct a graph to describe the relation among the temperatures on the sea surface, which exploits the spatial correlation adequately. Secondly, we utilize the smoothness of the temporal difference signal as a prior rather than the smoothness of the signal itself, and the former is more reasonable for this real-world time-varying graph signal.

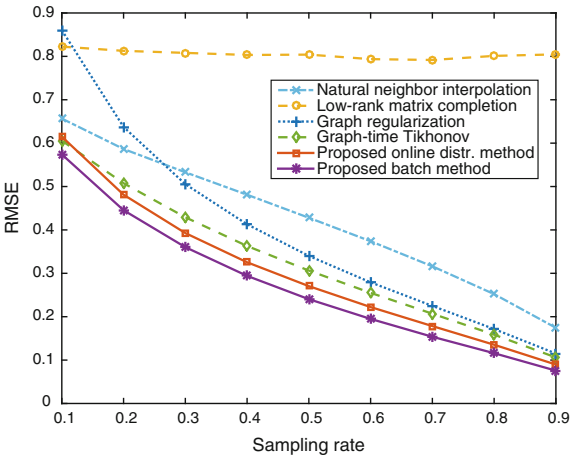


Fig. 5 The global sea-level pressure dataset: The reconstruction error under different sampling rate. The RMSE of each method is an average of 50 tests

6 Conclusion

In this chapter, the reconstruction of time-varying graph signals is studied. The existing literature of graph signal reconstruction usually assumes that the signal is bandlimited or at least approximately smooth on graph. However, we find that for most real-world time-varying graph signals, their temporal differences usually exhibit better smoothness on graph than the original signals do. Accordingly, two batch reconstruction methods for time-varying graph signals are proposed by exploiting the smoothness of the temporal difference signals on graph and the low-rank property of the original signals. Further, we propose an online distributed method based on the local properties of the temporal difference operator and the graph Laplacian matrix, which allows the signal reconstruction on each vertex to only rely on its neighbors and its own value at the previous time instance. We also highlight that spatio-temporal signals are special cases of time-varying graph signals and can be reconstructed from partial observations by the proposed methods. Performance analyses on sampling prerequisite and reconstruction error bounds are included. Experiments on different real-world datasets verify the efficiency of the proposed methods.

7 Additional Reading Material

There has been plenty of theoretical analysis on sampling and reconstruction of bandlimited graph signals [23, 47–50]. The concept of Paley–Wiener space is proposed in [47], and it is proved that a graph signal in Paley–Wiener space can be uniquely determined by its uniqueness set. That is to say, a bandlimited graph signal can be exactly reconstructed when sampling in accordance with its uniqueness set. In order to reconstruct bandlimited graph signals from decimation, several iterative

reconstruction algorithms of bandlimited graph signals are proposed [23, 48], and the generalizations to local aggregations are also studied [49, 50].

Nowadays time-varying graph signal processing has attracted increasing attention [51–54]. In order to process massive datasets with irregular structure, filtering and Fourier transform on product graphs are studied in [51], which is also suitable for time-varying signals. In [52], product graphs are also used to describe time-varying data, and visualization of time-varying graph signals is studied based on graph wavelet theory and stacked graph metaphor. In [53, 54], a more in-depth study of frequency analysis of temporal graph signals is conducted, and joint graph and temporal filters are designed. Recently, a new concept of joint stationarity is studied in [46, 55], which generalizes the classical definition of time stationarity to time-varying graph signals, assuming that signals are stationary with respect to both the time direction and the graph topology. Based on joint stationarity, the reconstruction of time-varying graph signals can be formulated as a Tikhonov-regularization problem that constrains the solution to be smooth with respect to both time and graph [46].

By constructing a graph to connect the observation sites, spatio-temporal signals can be viewed as time-varying graph signals. Nowadays many studies approximate spatio-temporal signals with low-rank matrices [37, 45, 56–58], and achieve satisfying results. A low-rank matrix estimation-based spatio-temporal image reconstruction method is investigated for dynamic photoacoustic computed tomography in [56], which assumes that the matrix collecting the sequence of vectorized images as its columns is approximately low-rank. The same assumption is also made in [37], which studies the recovery of arterial spin labeling MRI data from the noisy and corrupted observations. In [45, 57], data collection in wireless sensor networks is studied. The former [57] assumes that the sensor network data is approximately low-rank and has short-term stability in the temporal dimension. The latter [45] further supposes that the sensor network data is correlated in the spatial dimension, that is, the closely located sensors usually have similar measurements. Recovery of traffic matrices is studied in [58], which makes the same assumptions as [45], including low rank, temporal smoothness, and spatial smoothness. By minimizing the graph total variation of the signals at all time instants, a graph signal matrix completion algorithm is proposed in [28].

Acknowledgements This work was funded by the National Natural Science Foundation of China (NSFC 61571263 and NSFC 61531166005), the National Key Research and Development Program of China (Project No. 2016YFE0201900 and 2017YFC0403600), and Tsinghua University Initiative Scientific Research Program (Grant 2014Z01005).

References

1. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
2. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs. IEEE Trans. Signal Process **61**(7), 1644–1656 (2013)

3. N. Thanou, *Graph Signal Processing* (2016)
4. X. Shi, H. Feng, M. Zhai, T. Yang, Infinite impulse response graph filters in wireless sensor networks. IEEE Signal Process. Lett. **22**(8), 1113–1117 (2015)
5. A. Gadde, A. Ortega, A probabilistic interpretation of sampling theory of graph signals, in *IEEE International Conference on Acoustics, Speech and Signal Processing* (2015)
6. S.K. Narang, Y.H. Chao, A. Ortega, Graph-wavelet filterbanks for edge-aware image processing, in *Proceedings of the IEEE Statistics Signal Processing Workshop (SSP'12)* (2012), pp. 141–144
7. W.H. Kim, N. Adluru, M.K. Chung, S. Charchut, J.J. Gadelkarim, L. Altshuler, T. Moody, A. Kumar, V. Singh, A.D. Leow, Multi-resolutional brain network filtering and analysis via wavelets on non-euclidean space. Med. Image Comput. Comput. Assist. Interv. **16**(Pt3), 643–651 (2013)
8. S. Chen, A. Sandryhaila, J.M.F. Moura, J. Kovacevic, Signal denoising on graphs via graph filtering, in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (IEEE, 2014), pp. 872–876
9. S. Chen, A. Sandryhaila, J.M.F. Moura, J. Kovacevic, Adaptive graph filtering: multiresolution classification on graphs, in *2013 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2013), pp. 427–430
10. E. Isufi, A. Loukas, A. Simonetto, G. Leus, *Distributed Time-varying Graph Filtering*, arXiv:1602.04436
11. X. Zhu, M. Rabbat, Approximating signals supported on graphs, in *Proceedings of the 37th IEEE International Conference Acoustics, Speech, and Signal Processing (ICASSP)* (2012), pp. 3921–3924
12. D. Thanou, P.A. Chou, P. Frossard, Graph-based compression of dynamic 3d point cloud sequences. IEEE Trans. Image Process. **25**(4), 1765–1778 (2016)
13. P. Liu, X. Wang, Y. Gu, Coarsening graph signal with spectral invariance, in *Proceedings of the 39th IEEE International Conference Acoustics, Speech, Signal Processing (ICASSP)* (2014), pp. 1075–1079
14. P. Liu, X. Wang, Y. Gu, Graph signal coarsening: dimensionality reduction in irregular domain, in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2014), pp. 798–802
15. B. Girault, Stationary graph signals using an isometric graph translation, in *2015 23rd European Signal Processing Conference (EUSIPCO)* (2015), pp. 1516–1520
16. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Stationary graph processes and spectral estimation (2016), arXiv:1603.04667
17. N. Perraudin, P. Vandergheynst, Stationary signal processing on graphs (2016), arXiv:1601.02522
18. I. Pesenson, Variational splines and paley-wiener spaces on combinatorial graphs. Constr. Approx. **29**(1), 1–21 (2009)
19. I.Z. Pesenson, M.Z. Pesenson, Sampling, filtering and sparse approximations on combinatorial graphs. J. Fourier Anal. Appl. **16**(6), 921–942 (2010)
20. S. Chen, R. Varma, A. Sandryhaila, J. Kovačević, Discrete signal processing on graphs: sampling theory. IEEE Trans. Signal Process. **63**(24), 6510–6523 (2015)
21. T.M. Smith, R.W. Reynolds, Improved extended reconstruction of sst (1854–1997). J. Clim. **17**(12), 2466–2477 (2004)
22. L. Kong, M. Xia, X.Y. Liu, M. Wu, X. Liu, Data loss and reconstruction in sensor networks, in *INFOCOM, 2013 Proceedings IEEE* (2013), pp. 1654–1662
23. S.K. Narang, A. Gadde, E. Sanou, A. Ortega, Localized iterative methods for interpolation in graph structured data, in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2013), pp. 491–494
24. M. Belkin, I. Matveeva, P. Niyogi, Regularization and semi-supervised learning on large graphs, in *International Conference on Computational Learning Theory* (Springer, Berlin, 2004), pp. 624–638

25. S. Bougleux, A. Elmoataz, M. Melkemi, Discrete regularization on weighted graphs for image and mesh filtering, in *International Conference on Scale Space and Variational Methods in Computer Vision* (Springer, Berlin, 2007), pp. 128–139
26. D. Romero, M. Ma, G.B. Giannakis, Kernel-based reconstruction of graph signals (2016), arXiv:1605.07174
27. X. Wang, M. Wang, Y. Gu, A distributed tracking algorithm for reconstruction of graph signals. IEEE J. Sel. Top. Signal Process. **9**(4), 728–740 (2015)
28. S. Chen, A. Sandryhaila, J.M.F. Moura, J. Kovacevic, Signal recovery on graphs: variation minimization. IEEE Trans. Signal Process. **63**(17), 4609–4624 (2015)
29. I. Pesenson, Sampling in paley-wiener spaces on combinatorial graphs. Trans. Am. Math. Soc. **360**(10), 5603–5627 (2008)
30. K. Qiu, X. Mao, X. Shen, X. Wang, T. Li, Y. Gu, Time-varying graph signal reconstruction. IEEE J. Sel. Top. Signal Process. **11**(6), 870–883 (2017)
31. X. Mao, K. Qiu, T. Li, Y. Gu, Spatio-temporal signal recovery based on low rank and differential smoothness, in *Submitted to IEEE Transactions on Signal Processing*
32. Sea surface temperature (sst) v2 (2015), http://www.esrl.noaa.gov/psd/data/gridded/data.noaa.oisst.v2.html
33. Sea-level pressure, 1948–2010 (2016), http://research.jisao.washington.edu/data_sets/reanalysis/
34. Air quality data (2016), https://www.epa.gov/outdoor-air-quality-data
35. D.P. Bertsekas, *Nonlinear Programming* (Athena Scientific, Belmont, 1999), pp. 130–145
36. M.T. Bahadori, Q.R. Yu, Y. Liu, Fast multivariate spatio-temporal analysis via low rank tensor learning, in *Advances in Neural Information Processing Systems* (2014), pp. 3491–3499
37. R. Fang, J. Huang, W.M. Luh, A spatio-temporal low-rank total variation approach for denoising arterial spin labeling MRI data, in *IEEE 12th International Symposium on Biomedical Imaging (ISBI)* (IEEE, 2015), pp. 498–502
38. M.E. El-Telbany, M.A. Maged, Exploiting sparsity in wireless sensor networks for energy saving: a comparative study. Int. J. Appl. Eng. Res. **12**(4), 452–460 (2017)
39. S. Ma, D. Goldfarb, L. Chen, Fixed point and bregman iterative methods for matrix rank minimization. Math. Progr. **128**(1–2), 321–353 (2011)
40. M. Fazel, Matrix rank minimization with applications. Ph.D. dissertation, Ph.D. thesis, Stanford University (2002)
41. S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends® Mach. Learn. **3**(1), 1–122 (2011)
42. D. Gabay, B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Comput. Math. Appl. **2**(1), 17–40 (1976)
43. J. Eckstein, W. Yao, Understanding the convergence of the alternating direction method of multipliers: theoretical and computational perspectives. Pac. J. Optim. **11**(4), 619–644 (2015)
44. S. Robert, A brief description of natural neighbour interpolation. Interpret. Multivar. Data **21**, 21–36 (1981)
45. X. Piao, Y. Hu, Y. Sun, B. Yin, J. Gao, Correlated spatio-temporal data collection in wireless sensor networks based on low rank matrix approximation and optimized node sampling. Sensors **14**(12), 23 137–23 158 (2014)
46. N. Perraudin, A. Loukas, F. Grassi, P. Vandergheynst, Towards stationary time-vertex signal processing (2016), arXiv:1606.06962
47. I. Pesenson, Sampling in paley-wiener spaces on combinatorial graphs. Trans. Am. Math. Soc. **360**(10), 5603–5627 (2008)
48. X. Wang, P. Liu, Y. Gu, Local-set-based graph signal reconstruction. IEEE Trans. Signal Process. **63**(9), 2432–2444 (2015)
49. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Sampling of graph signals with successive local aggregations. IEEE Trans. Signal Process. **64**(7), 1832–1843 (2015)
50. X. Wang, J. Chen, Y. Gu, Local measurement and reconstruction for noisy bandlimited graph signals. Signal Process. **129**(5), 119–129 (2016)

51. A. Sandryhaila, J.M.F. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process. Mag. **31**(5), 80–90 (2014)
52. P. Valdivia, F. Dias, F. Petronetto, C.T. Silva, L.G. Nonato, Wavelet-based visualization of time-varying data on graphs, in *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2015), pp. 1–8
53. A. Loukas, D. Foucard, Frequency analysis of temporal graph signals (2016), arXiv preprint arXiv:1602.04434
54. E. Isufi, A. Loukas, A. Simonetto, G. Leus, Autoregressive moving average graph filtering. IEEE Trans. Signal Process. **65**(2), 274–288 (2017)
55. A. Loukas, N. Perraudin, Stationary time-vertex signal processing (2016), arXiv preprint arXiv:1611.00255
56. K. Wang, J. Xia, C. Li, L.V. Wang, M.A. Anastasio, Low-rank matrix estimation-based spatio-temporal image reconstruction for dynamic photoacoustic computed tomography, in *SPIE BiOS. International Society for Optics and Photonics* (2014), pp. 89 432I–89 432I
57. J. Cheng, Q. Ye, H. Jiang, D. Wang, C. Wang, STCDG: an efficient data gathering algorithm based on matrix completion for wireless sensor networks. IEEE Trans. Wirel. Commun. **12**(2), 850–861 (2013)
58. Y. Zhang, M. Roughan, W. Willinger, L. Qiu, Spatio-temporal compressive sensing and internet traffic matrices, in *ACM SIGCOMM 2009 Conference on Data Communication* (2009), pp. 267–278

Uncertainty Principle on Graphs



Bastien Pasdeloup, Vincent Gripon, Réda Alami and Michael G. Rabbat

Abstract Graph Signal Processing (GSP) is a mathematical framework that aims at extending classical Fourier harmonic analysis to irregular domains described using graphs. Within this framework, authors have proposed to define operators (e.g. translations, convolutions) and processes (e.g. filtering, sampling). A very important and fundamental result in classical harmonic analysis is the uncertainty principle, which states that a signal cannot be localized both in time and in frequency domains. In this chapter, we explore the uncertainty principle in the context of GSP. More precisely, we present notions of graph and spectral spreads, and show that the existence of signals that are both localized in the graph domain and in the spectrum domain depends on the graph.

1 Introduction

In classical signal processing, signals under study are generally defined on very regular domains, such as a path graph for temporal signals like audio, a two-dimensional lattice for spatial signals like images, and a three-dimensional lattice for spatio-temporal signals like video. Frequency analysis of such objects is performed thanks to the Fourier transform operator, which projects the signal under study into a basis of sines, providing a convenient dual representation for it. In the case of more complex

B. Pasdeloup (✉)
EPFL, Route Cantonale, 1015 Lausanne, Switzerland
e-mail: bastien.pasdeloup@epfl.ch

V. Gripon
IMT Atlantique, Technopole Brest-Iroise, Plouzané, France
e-mail: vincent.gripon@imt-atlantique.net

R. Alami
Orange Labs, Lannion, France
e-mail: reda.alami@imt-atlantique.net

M. G. Rabbat
McGill University, Montréal, QC, Canada
e-mail: michael.rabbat@mcgill.ca

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_9

signals, such as images or videos, the idea remains the same, except that the sines are now two or three-dimensional.

When the underlying domain on which signals are studied becomes irregular—e.g., a sensor network, a social network, an affinity graph, etc.—, defining an adapted Fourier transform is not as natural. However, providing a frequency representation for signals evolving over such domains is of real interest, and has applications such as anomaly detection [11], brain activity comprehension [12], or topology analysis [4].

In order to study signals over such complex topologies, *graph signal processing* (GSP) has emerged as a field proposing to extend classical signal processing tools to irregular domains modeled as graphs. The field developed from the observation that the basis of eigenvectors associated with the Laplacian matrix of a ring graph are exactly sines, with increasing frequency as the associated eigenvalues increase [19]. The Fourier transform operator is therefore directly given by this particular matrix, which is strongly tied with the graph modeling the support of information. Interestingly, this correspondence between the eigenvectors of the Laplacian matrix and the Fourier modes also holds for arbitrary graphs. As a matter of fact, the eigenvectors associated with the lowest eigenvalues vary smoothly on the graph, while more important variations appear as the associated eigenvalues increase. Using this so-called Graph Fourier Transform (GFT), numerous tools have been successfully developed, allowing operations such as filtering [19], wavelet decomposition of signals on graphs [9], translation, modulation, etc. (see [14, 19] for overviews of such tools).

Existence of a Fourier transform for signals on graphs also raises the question of uncertainty. In classical signal processing, it is established that a signal cannot be simultaneously localized both in time and in the frequency domain. In this chapter, we present a corresponding uncertainty principle for signals on graphs. We show that the extent to which a signal can be localized in both the graph vertex domain and in the graph spectral domain is tied to the graph topology, and that different graphs have different locality properties.

2 Definitions

In the rest of this chapter, we adopt the following notation. Sets are denoted in calligraphic letters (e.g., $\mathcal{V}$, $\mathcal{E}$). Constants and scalar variables are written in italic font, respectively using capital and lower-case letters (e.g., constant N, and scalar variables i, j). Matrices and vectors are denoted in bold, respectively with capital and lower-case letters, with entries having subscripted indices (e.g., matrix $\mathbf{A}$ and vector $\mathbf{v}$ with entries $\mathbf{A}_{ij}$, $\mathbf{v}_i$).

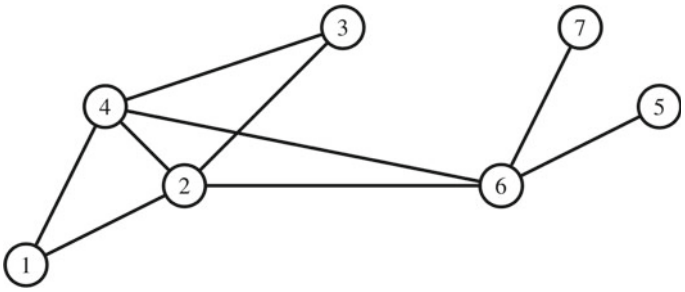


Fig. 1 Example of a graph containing 7 vertices $1, 2, \ldots, 7$. Vertices are depicted as circles and edges as lines connecting them

2.1 Graphs and Matrices

Let us consider a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ is the finite set of vertices and $\mathcal{E}$ is the set of edges. Graphs are mathematical models that are useful to describe relations between objects (vertices). In the context of graph signal processing, it often makes sense to consider edges to be pairs of distinct, unordered vertices, meaning that an edge conveys just enough information to denote whether two vertices are connected or not. As such, there are at most $\binom{N}{2}$ edges in a graph containing N vertices. An example of a graph is depicted in Fig. 1.

In order to ease readability, let us consider vertices to be indexed from 1 to N: $\mathcal{V} = \{1, 2, \ldots, N\}$.

Any graph can be conveniently described by its adjacency matrix. The adjacency matrix $\mathbf{A}$ of a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ with N vertices is an N-by-N matrix with entries defined as follows:

$$\forall i, j \in \mathcal{V} : \mathbf{A}_{ij} = \begin{cases} 1 \text{ if } \{i, j\} \in \mathcal{E} \\ 0 \text{ otherwise.} \end{cases} \tag{1}$$

In the remainder of this chapter, we always consider vertices to be integers between 1 and N, such that we make no distinction between a vertex and its index.

An example graph is shown in Fig. 1, and its adjacency matrix is:

$$\mathbf{A} = \begin{bmatrix} 0\,1\,0\,1\,0\,0\,0 \\ 1\,0\,1\,1\,0\,1\,0 \\ 0\,1\,0\,1\,0\,0\,0 \\ 1\,1\,1\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,0 \\ 0\,1\,0\,1\,1\,0\,1 \\ 0\,0\,0\,0\,0\,1\,0 \end{bmatrix}. \tag{2}$$

Note that the adjacency matrix of a graph is symmetric. This is because we chose edges as *pairs* (unordered sets) of vertices. Extended definitions of graphs have

been proposed in the literature, including digraphs where edges are *couples* (ordered pairs) of vertices. Digraphs are particularly useful when representing oriented relations between vertices. As a consequence, adjacency matrices of digraphs are not necessarily symmetric. For the rest of the chapter we focus on symmetric graphs.

Other extensions include weighted graphs. With such graphs, the notion of edges is refined to take into account intensities. In this context, we introduce the weight matrix $\mathbf{W}$ which is such that:

$$\forall i, j \in \mathscr{V} : (\mathbf{W}_{ij} > 0) \Rightarrow (\mathbf{A}_{ij} = 1). \tag{3}$$

In other words, the weight matrix of a graph has the same support as the adjacency matrix of the graph. More precisely, it determines the weight of each edge in the graph. If the graph is unweighted, we conveniently adopt the convention that the weight matrix is exactly identical to the adjacency matrix of the graph.

2.2 Basic Definitions on Graphs

In this section, we introduce some definitions from graph theory that will be used in the following sections. More complete literature on graphs can be found for instance in [8].

Let us first introduce paths and walks on a graph.

Definition 1 A *walk* on a graph is a (possibly infinite) sequence of vertices, such that any two consecutive vertices form an edge in the graph. Getting back to the example of Fig. 1, the sequence (1, 2, 6, 2, 3, 4) is a walk.

Definition 2 A *path* on a graph is a walk in which each consecutive (unordered) pair of vertices appears at most once. As such, paths are necessarily finite because there is only a finite number of possible pairs of vertices. The starting and ending vertices of a path are called its *extremities*. An example of a path for the graph in Fig. 1 is (1, 2, 4, 6).

Paths and walks are often confused in the literature due to their very similar definitions. Walks are shorter to define, but paths allow to define cycles.

Definition 3 A *cycle* on a graph is a path with identical extremities. Not all graphs admit cycles. An example of a cycle for the graph in Fig. 1 is (1, 2, 4, 1).

Definition 4 The *length* of a path is the number of vertices in the sequence minus 1. For example, the length of the path (1, 2, 4, 6) is 3. This is also the number of edges traversed in the path.

Definition 5 The *weight* of a path is the sum of the weights of edges formed by consecutive vertices in the path.

Weights and length are two separate notions for weighted graphs. Thanks to our previously mentioned convention, in the case of unweighted graphs they are identical.

Definition 6 A *connected* graph is a graph for which every pair of vertices are extremities of at least one path.

Definition 7 The *geodesic distance* $d_{\mathscr{G}}$ on a graph $\mathscr{G}$ is a function that associates a pair of vertices with the minimum weight of a path having these vertices as its extremities.

Based on these definitions, one can define classical families of graphs:

Definition 8 A *tree* is a connected graph that contains no cycle.

Definition 9 A *bipartite* graph is a graph that contains no cycle with odd length.

Definition 10 A *ring* graph is an unweighted graph with N vertices in which all edges appear in a single cycle of length N.

Definition 11 A *complete* graph is an unweighted graph containing all possible edges.

Definition 12 A *star* graph is an unweighted graph with N vertices and $N-1$ edges for which all edges have one extremity in common.

2.3 Meaning of Edge Weights

It is very important to understand that the convention we use here to put 0s in the matrices when there is no link is not without consequence. Indeed, we suppose a weight 0 is equivalent to the absence of an edge between the corresponding vertices. As such, weights should not represent quantities for which this is a contradiction.

For example, consider a graph in which vertices model cities and connection weights represent road distances between these cities. The absence of a connection between two cities could correspond to the absence of a direct road connecting them, in which case the distance should not be 0, but to the contrary $+\infty$. Such weights do not make sense with respect to our convention.

Now imagine a graph in which vertices are terminals on the Internet and weights represent the number of packets that directly travel between pairs of terminals. In such a graph, a connection weight 0 corresponds to the absence of a direct connection, or to a completely useless one. Such weights make sense with respect to our convention.

There are fine underlying theoretical reasons to explain why we choose this convention of 0s and 1s in the adjacency matrix, and it is mainly related to the fact that we suppose working with the regular linear algebra. Distance graphs, that we discussed before, are better processed using the tropical algebra in which $+\infty$ is a neutral element for the addition [7].

The weighted graphs we introduce in this document typically model the similarity of their corresponding vertices, and not their distances.

2.4 The Graph Laplacian and its Properties

Consider a connected graph $\mathcal{G}$ with N vertices together with its weight matrix $\mathbf{W}$. We call *strength*[1] of a vertex i the quantity:

$$s(i) = \sum_{j \in \mathcal{V}} \mathbf{W}_{ij} \ . \tag{4}$$

Definition 13 A graph is said to be *regular* if all of its vertices have the same strength.

The strengths of all vertices can be merged into a single diagonal matrix $\mathbf{S}$ called the *strength matrix*, such that:

$$\forall i, j \in \mathcal{V} : \mathbf{S}_{ij} = \begin{cases} s(i) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The strength matrix of a graph is especially useful to perform various kinds of normalizations on this graph. For instance, a common way to transform a graph into a Markov chain consists in considering the matrix $\mathbf{S}^{-1}\mathbf{W}$, of which sums of rows always equal 1.

The strength matrix of a graph can also be used to define the Laplacian of a graph:

Definition 14 The *Laplacian* of a graph $\mathcal{G}$ with weight matrix $\mathbf{W}$ and strength matrix $\mathbf{S}$ is:

$$\mathbf{L} = \mathbf{S} - \mathbf{W} \ . \tag{6}$$

As an example, the Laplacian of the example graph in Fig. 1 is:

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 4 & -1 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ -1 & -1 & -1 & 4 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & -1 & 0 & -1 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} \tag{7}$$

Being a symmetric, real-valued matrix, the Laplacian of a graph with N vertices can be written as:

$$\mathbf{L} = \mathbf{F}\mathbf{\Lambda}\mathbf{F}^{\top} \ , \tag{8}$$

[1] This quantity is often referred to as *degree* in the literature of GSP. In graph theory the degree refers to the *number* of neighbors of a given vertex whereas its strength takes into account the weights of corresponding edges. These two quantities are identical when considering unweighted graphs.

where $\mathbf{F}$ is such that $\mathbf{F}\mathbf{F}^{-1} = \mathbf{F}^{-1}\mathbf{F} = \mathbf{I}_N$, $\mathbf{I}_N$ being the identity matrix of dimension N, and $\mathbf{\Lambda}$ is a diagonal matrix of which diagonal elements are $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. In other words, $\mathbf{F}$ is a matrix of eigenvectors and $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues, arranged in ascending order.

The Laplacian of a graph offers multiple interesting properties, as pointed out in the next propositions.

Proposition 1 *Let* $\mathbf{L}$ *be the Laplacian matrix of a graph* $\mathcal{G}$*. The vector* $\mathbf{1}$ *with all entries equal to* 1 *is an eigenvector of* $\mathbf{L}$ *associated with the eigenvalue* 0*.*

Proof The proof is straightforward:

$$\forall i \in \mathcal{V} : (\mathbf{L1})_i = (\mathbf{S1})_i - (\mathbf{W1})_i = s(i) - \sum_{j \in \mathcal{V}} \mathbf{W}_{ij} = 0 \, . \tag{9}$$

Proposition 2 *The eigenvalues of the Laplacian of a graph are all nonnegative.*

Proof Suppose, for the sake of a contradiction, that some negative eigenvalue exists. Let us denote it λ and let $\mathbf{f}$ be an associated nonzero eigenvector. Let us look at one of the entries i of $\mathbf{f}$ such that $|\mathbf{f}_i|$ is maximum. We obtain:

$$s(i)\mathbf{f}_i - \sum_{j \in \mathcal{V}} \mathbf{W}_{ij}\mathbf{f}_j = \lambda \mathbf{f}_i \, , \tag{10}$$

which can be rewritten as:

$$\mathbf{f}_i - \sum_{j \in \mathcal{V}} \frac{\mathbf{W}_{ij}}{s(i)}\mathbf{f}_j = \frac{\lambda}{s(i)}\mathbf{f}_i \, . \tag{11}$$

Without loss of generality, we can suppose that $\mathbf{f}_i$ is positive. As such, the right term of this equality is negative. We conclude that:

$$\sum_{j \in \mathcal{V}}^{N} \frac{\mathbf{W}_{ij}}{s(i)}\mathbf{f}_j > \mathbf{f}_i \, , \tag{12}$$

which is a contradiction since the left part of this inequality is a weighted average of values of $\mathbf{f}_j$ that are by definition all less than or equal to $\mathbf{f}_i$.

From the two previous propositions we conclude that $\lambda_1 = 0$, in all cases.

Proposition 3 *The second eigenvalue of the Laplacian of a graph* $\mathcal{G}$ *is* 0 *if and only if the graph is not connected.*

Proof It is immediate to see that the second eigenvalue is 0 if the graph is not connected. Indeed, fix some vertex $i \in \mathcal{V}$ and consider the set $\mathcal{V}_i \subset \mathcal{V}$ of vertices connected to i and the complement set $\overline{\mathcal{V}_i}$. Then the two linearly independent vectors

obtained by putting 1s on coordinates in $\mathscr{V}_i$ and 0s in those of $\overline{\mathscr{V}_i}$, and conversely, are both eigenvectors associated with the eigenvalue 0.

Conversely, consider a non-constant eigenvector associated with eigenvalue 0. Denote it by $\mathbf{f}$, and let i be an index such that $|\mathbf{f}_i|$ is maximum. Since we have:

$$\mathbf{f}_i = \sum_{j \in \mathscr{V}}^{N} \frac{\mathbf{W}_{ij}}{s(i)} \mathbf{f}_j \tag{13}$$

is a weighted sum of the value of the neighbors of vertex i, we conclude that all its neighbors have the same value as i. By repeating this process, we conclude that any vertex connected to i has the same value in $\mathbf{f}$ as i. If the graph were connected, the obtained vector would be constant.

Proposition 4 *The quadratic form of the Laplacian* $\mathbf{L}$ *of a graph is such that:*

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \sum_{\{i,j\} \in \mathscr{E}} \mathbf{W}_{ij} \left(\mathbf{x}_i - \mathbf{x}_j\right)^2 . \tag{14}$$

Proof We use simple mathematics here:

$$\begin{aligned}
\mathbf{x}^\top \mathbf{L}\mathbf{x} &= \mathbf{x}^\top (\mathbf{S} - \mathbf{W})\mathbf{x} \\
&= \sum_{i \in \mathscr{V}} s(i)\mathbf{x}_i^2 - \sum_{\substack{i,j \in \mathscr{V} \\ i<j}} 2\mathbf{W}_{ij}\mathbf{x}_i\mathbf{x}_j \\
&= \sum_{\substack{i,j \in \mathscr{V} \\ i<j}} \mathbf{W}_{ij} \left(\mathbf{x}_i^2 + \mathbf{x}_j^2\right) - \sum_{\substack{i,j \in \mathscr{V} \\ i<j}} 2\mathbf{W}_{ij}\mathbf{x}_i\mathbf{x}_j \\
&= \sum_{\substack{i,j \in \mathscr{V} \\ i<j}} \mathbf{W}_{ij} \left(\mathbf{x}_i - \mathbf{x}_j\right)^2 .
\end{aligned} \tag{15}$$

Some authors prefer to use the normalized Laplacian $\mathbf{Ł}$ instead of the Laplacian $\mathbf{L}$, where:

$$\mathbf{Ł} = \mathbf{S}^{-\frac{1}{2}} \mathbf{L} \mathbf{S}^{-\frac{1}{2}} . \tag{16}$$

Quadratic forms involving the normalized Laplacian satisfy a similar relationship:

$$\mathbf{x}^\top \mathbf{Ł}\mathbf{x} = \sum_{\{i,j\} \in \mathscr{E}} \mathbf{W}_{ij} \left(\frac{\mathbf{x}_i}{\sqrt{s(i)}} - \frac{\mathbf{x}_j}{\sqrt{s(j)}} \right)^2 . \tag{17}$$

The normalized Laplacian also satisfies Proposition 3, along with three others:

Proposition 5 *Consider the normalized Laplacian* $\mathbf{Ł}$ *of a connected graph* $\mathscr{G}$*, then the spectrum of* $\mathbf{Ł}$ *is between* 0 *and* 2.

Proof Introduce the Rayleigh coefficient:

$$r(\text{Ł}, \mathbf{x}) = \frac{\mathbf{x}^\top \text{Ł}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} \,. \tag{18}$$

It is thus sufficient to show that this coefficient is between 0 and 2 [13]. We obtain:

$$\begin{aligned} \frac{\mathbf{x}^\top \text{Ł}\mathbf{x}}{\mathbf{x}^\top \mathbf{x}} &= \frac{\mathbf{x}^\top \mathbf{S}^{-\frac{1}{2}} \mathbf{L} \mathbf{S}^{-\frac{1}{2}} \mathbf{x}}{\mathbf{x}^\top \mathbf{x}} \\ &= \frac{\mathbf{x}^\top \mathbf{L}\mathbf{x}}{\left(\mathbf{S}^{\frac{1}{2}}\mathbf{x}\right)^\top \mathbf{S}^{\frac{1}{2}}\mathbf{x}} \\ &= \frac{\sum_{\substack{i,j \in \mathscr{V} \\ i<j}} \mathbf{W}_{ij} \left(\mathbf{x}_i - \mathbf{x}_j\right)^2}{\sum_{i \in \mathscr{V}} s(i) \mathbf{x}_i^2} \,. \end{aligned} \tag{19}$$

This quantity is clearly nonnegative. This concludes the proof since, for any real numbers a and b, $(a-b)^2 \leq 2(a^2+b^2)$.

Proposition 6 *The largest eigenvalue of the normalized Laplacian* Ł *of a connected unweighted graph* $\mathscr{G}$ *is* 2 *if and only if the graph is bipartite.*

Proof The proof is omitted here. See [3] for details.

Proposition 7 *The first eigenvalue of the normalized Laplacian is always* 0*, and it has associated eigenvector* $\mathbf{f}$ *where:*

$$\forall i \in \mathscr{V} : \mathbf{f}_i = \sqrt{\frac{s(i)}{\sum_{j \in \mathscr{V}} s(j)}} \,. \tag{20}$$

Proof Observe that

$$\mathbf{f} = \frac{\mathbf{S}^{\frac{1}{2}}\mathbf{1}}{\sqrt{\sum_{j \in \mathscr{V}} s(j)}} \,. \tag{21}$$

We have:

$$\begin{aligned} \left[\sqrt{\sum_{j \in \mathscr{V}} s(j)}\right] \text{Ł}\mathbf{f} &= \mathbf{S}^{\frac{1}{2}}\mathbf{1} - \mathbf{S}^{-\frac{1}{2}}\mathbf{W}\mathbf{1} \\ &= 0 \,. \end{aligned} \tag{22}$$

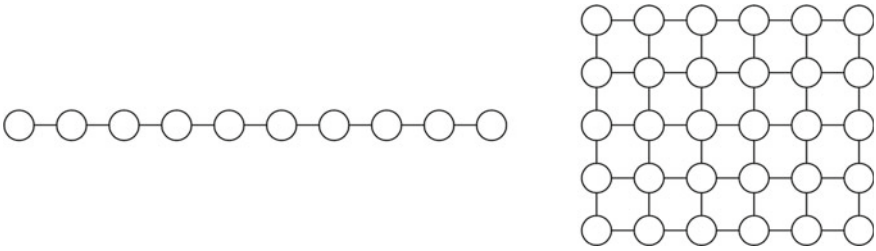


Fig. 2 Example of a line graph (left) and a grid graph (right) that are natural topologies to represent a sound or an image

3 Graph Signals and Fourier Transform

3.1 Graphs and Signals

As mentioned in the introduction, a graph is a convenient tool to model the topology of a signal. Consider a snippet of audio for instance, which is a continuous, smooth function of time. Typically, such signal is represented in computer memories using a regular sampling of time. It is thus a collection of values corresponding to the amplitude of the sound measured at distinct, regularly-spaced time steps. With no additional priors about the considered sound, it is reasonable to model smoothness by saying that two consecutive measurements are likely to be similar—at least, more than two measurements separated by more time. A natural representation of the topology of a sound is therefore obtained using a line graph, as depicted in Fig. 2 (left). Considering an image instead, and following the same reasoning, a typical graph to model its topology would be a grid, as depicted in Fig. 2 (right).

As such, we have a first correspondence between signals and graphs. If a signal is a vector $\mathbf{x} \in \mathbb{R}^N$, then its topology should be modeled by a graph containing N vertices, one per coordinate of the vector.

Of course there is no real interest in introducing graphs to represent the topology of images or sounds. These are just particular examples of topologies. In practice we are interested in other families of graphs, some of which we introduce in the following definitions.

Definition 15 An *Erdős–Rényi* graph with parameters N and P is an unweighted graph with N vertices that is obtained by drawing independently at random each edge with a Bernoulli random variable with parameter P.

As such, an Erdős–Rényi graph with $P = 1$ is a complete graph.

Definition 16 A *random geometric* graph with parameters N and R is an unweighted graph with N vertices that is obtained by drawing uniformly at random 2D coordinates between 0 and 1 for the N vertices. Then vertices which coordinates are less than R apart, considering ℓ_2 norm, are connected through an edge.

Again, choosing $R \geq \sqrt{2}$ leads to a complete graph.

Erdős–Rényi graphs have interesting asymptotic properties, such as possibly being at the same time sparsely connected together with each pairs of vertices at very small distances. Random geometric graphs are often used to describe sensor networks, as they are built using underlying 2D coordinates. Although they may have irregular, complex structure, random geometric graphs often have properties which are very similar to those of a two-dimensional grid.

3.2 Sharpness

Given a signal $\mathbf{x} \in \mathbb{R}^N$, there are $2^{\binom{N}{2}}$ possible unweighted graphs to represent its topology, and an infinite number of weighted graphs. Understanding the relations between graphs and signals requires to quantify these relationships.

Since the graphs we introduce in this chapter typically model similarities between their vertices, it is natural to expect connected vertices to contain similar values. This can be measured with sharpness:

Definition 17 Consider a graph $\mathcal{G}$ with N vertices and normalized Laplacian $\mathbf{Ł}$, and a signal $\mathbf{x} \in \mathbb{R}^N$. The *sharpness*[2] of $\mathbf{x}$ on $\mathcal{G}$ is the quantity:

$$h(\mathbf{x}) = \mathbf{x}^\top \mathbf{Ł}\mathbf{x} \,. \tag{23}$$

Note that sharpness grows quadratically with the norm of a signal:

$$\forall \mathbf{x} \in \mathbb{R}^N, \forall \alpha \in \mathbb{R} : \alpha^2 h(\mathbf{x}) = h(\alpha \mathbf{x}) \,, \tag{24}$$

such that we consider in the following the normalized sharpness:

$$\hbar(\mathbf{x}) = \frac{\mathbf{x}^\top \mathbf{Ł}\mathbf{x}}{\|\mathbf{x}\|_2^2} = h\left(\frac{\mathbf{x}}{\|\mathbf{x}\|_2}\right) \,, \tag{25}$$

for nonzero signals, with the convention that $\hbar(\mathbf{0}) = 0$.

More generally, signals with very similar values at well-connected vertices will present a smaller normalized sharpness than signals with significant differences between those vertices. As such, when the normalized sharpness is close to zero, it makes sense to say that the signal is aligned with the graph whereas when sharpness is large it is not. In other words, signals aligned with the graph are signals that are close (in terms of angle) to the span of the first eigenvectors of $\mathbf{Ł}$ (those with smallest eigenvalues).

[2]This quantity is often denoted *smoothness* in the GSP literature. However, the lower this value, the smoother the signal on the graph.

3.3 *Diffusion Sequence*

Definition 18 Consider a signal $\mathbf{x}$. We call *diffusion sequence* of $\mathbf{x}$ the sequence:

$$\left((\mathbf{I}_N - \mathbf{Ł})^t \mathbf{x}\right)_{t \in \mathbb{N}^*} . \tag{26}$$

In other words, the diffusion sequence takes a signal $\mathbf{x}$, and iteratively mixes it using the matrix $\mathbf{I}_N - \mathbf{Ł}$.

Proposition 8 *For any connected graph, the diffusion sequence of a signal* $\mathbf{x}$ *converges to the first eigenvector of its normalized Laplacian (the one associated with eigenvalue* 0*) if the graph is not bipartite.*

Proof Denote $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$ the eigenvalues of the normalized Laplacian $\mathbf{Ł}$ of a connected graph $\mathcal{G}$, and $\mathbf{f}_1, \ldots, \mathbf{f}_N$ the corresponding unit norm eigenvectors. Then for any unit norm signal $\mathbf{x}$, we obtain:

$$(\mathbf{I}_N - \mathbf{Ł})^t \mathbf{x} = (1 - \lambda_1)^t \left(\mathbf{x}^\top \mathbf{f}_1\right) \mathbf{f}_1 + \cdots + (1 - \lambda_N)^t \left(\mathbf{x}^\top \mathbf{f}_N\right) \mathbf{f}_N . \tag{27}$$

Because of Proposition 6, we have $\forall i \geq 2 : |1 - \lambda_i| < 1$, and thus:

$$\lim_t (\mathbf{I}_N - \mathbf{Ł})^t \mathbf{x} = \left(\mathbf{x}^\top \mathbf{f}_1\right) \mathbf{f}_1 . \tag{28}$$

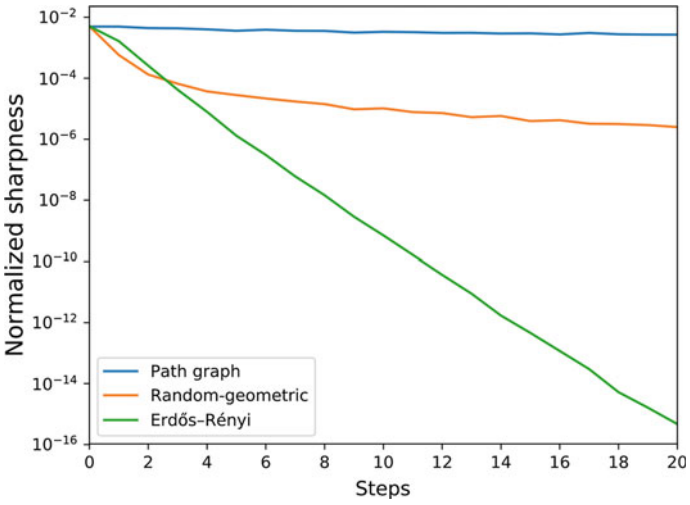


Fig. 3 Normalized sharpness of uniformly drawn random signals, as a function of the number of diffusion steps and for three graphs: path, Erdős–Rényi and random-geometric. All graphs contain exactly 200 vertices. Erdős–Rényi graphs were generated with $P = 0.07$ and random-geometric graphs with $R = 0.2$. Drawn curves were obtained by averaging over 100 graphs

In the case of a bipartite graph, it is interesting to see that in the above expression both $1 - \lambda_1 = 1$ and $1 - \lambda_N = -1$ do not vanish, leading to an almost alternative sequence.

In some sense, as we go through the diffusion sequence of a signal, it is increasingly more bound to the graph, with the extreme case of reaching a sharpness of 0 (using normalized Laplacian) after an infinite number of steps when the graph is not bipartite. In Fig. 3, we draw the average normalized sharpness of randomly uniformly generated signals on a path graph, an Erdős–Rényi graph and a random-geometric graph as a function of the number of steps. The path graph is a bipartite graph, which explains why its curve does not seem to converge to 0. But interestingly, we see here that some graphs have faster convergence to 0 than others, which is due to the spectrum of their normalized Laplacian, more concentrated around 1.

4 Graph Fourier Transform (GFT)

4.1 Analogy with Discrete Fourier Transform and Definitions

In the case of a ring graph, vertices can be indexed so that the adjacency matrix becomes circulant [6]. For a ring graph with 7 vertices, it would look like:

$$\mathbf{A} = \begin{bmatrix} 0&1&0&0&0&0&1 \\ 1&0&1&0&0&0&0 \\ 0&1&0&1&0&0&0 \\ 0&0&1&0&1&0&0 \\ 0&0&0&1&0&1&0 \\ 0&0&0&0&1&0&1 \\ 1&0&0&0&0&1&0 \end{bmatrix}$$

Of course, it is also the case for its Laplacian matrix, which is:

$$\mathbf{L} = \begin{bmatrix} 2&-1&0&0&0&0&-1 \\ -1&2&-1&0&0&0&0 \\ 0&-1&2&-1&0&0&0 \\ 0&0&-1&2&-1&0&0 \\ 0&0&0&-1&2&-1&0 \\ 0&0&0&0&-1&2&-1 \\ -1&0&0&0&0&-1&2 \end{bmatrix},$$

and the normalized Laplacian is also circulant.

As a consequence, it is possible to obtain a mathematical closed form of the eigenvectors, which in this case are exactly discrete Fourier modes. In other words,

consider a signal $\mathbf{x} \in \mathbb{R}^N$, and multiply it by the transpose matrix of eigenvectors $\mathbf{F}$ of the Laplacian of the ring graph. You obtain the discrete Fourier transform of $\mathbf{x}$.

Following this analogy, the *graph Fourier transform* (GFT) can be defined.

Definition 19 The *Graph Fourier Transform (GFT)* of a signal $\mathbf{x}$ on a graph $\mathscr{G}$ is the operation:

$$\widehat{\mathbf{x}} = \mathbf{F}^\top \mathbf{x} \,. \tag{29}$$

The *Inverse Graph Fourier Transform (IGFT)* of a signal $\widehat{\mathbf{x}}$ on a graph $\mathscr{G}$ is the operation:

$$\mathbf{x} = \mathbf{F}\widehat{\mathbf{x}} \,. \tag{30}$$

4.2 Examples

In order to illustrate some of the previously introduced properties, we consider here a random-geometric graph $\mathscr{G}$, depicted in Fig. 4. A uniformly random signal $\mathbf{x}$ has been initialized on its vertices. Its spectrum $\widehat{\mathbf{x}} = \mathbf{F}^\top \mathbf{x}$ (using the eigenvectors of the normalized Laplacian) is also shown.

The graph Fourier transform provides two representations of a same signal on the graph, linearly related through the matrix of eigenvectors of the Laplacian matrix. When represented on the graph, the signal is a scalar value observed at each vertex, which generally represents an observed phenomenon to analyze. Its spectral representation gives insights of the properties of this signal, such as its sharpness on the graph or its bandwidth for instance.

Also, these eigenvectors being vectors of dimension N, they can be seen as signals on the graph, revealing some interesting properties of the associated eigenvalues. Figure 5 depicts some of these.

As illustrated, the eigenvectors associated with the lowest eigenvalues of the Laplacian matrix vary smoothly on the graph, while those associated with larger ones tend to model more localized, sharp variations. This illustrates the analogy between the eigenvalues of this matrix and some notion of frequencies in classical signal processing.

When considering a diffusion process, eigenvalues in the Laplacian spectrum that are close to 0 are located next to eigenvalue 1 in the spectrum of the diffusion matrix. As a consequence, they vanish in a slower way compared to Laplacian higher eigenvalues. Since the eigenvectors associated to these eigenvalues are highly localized patterns on the graph, a diffusion phenomenon on a graph can be understood as a process that smoothens signal values across the graph, similarly to heat diffusion in the classical settings. Figure 6 depicts a few diffusion steps of the signal in Fig. 4.

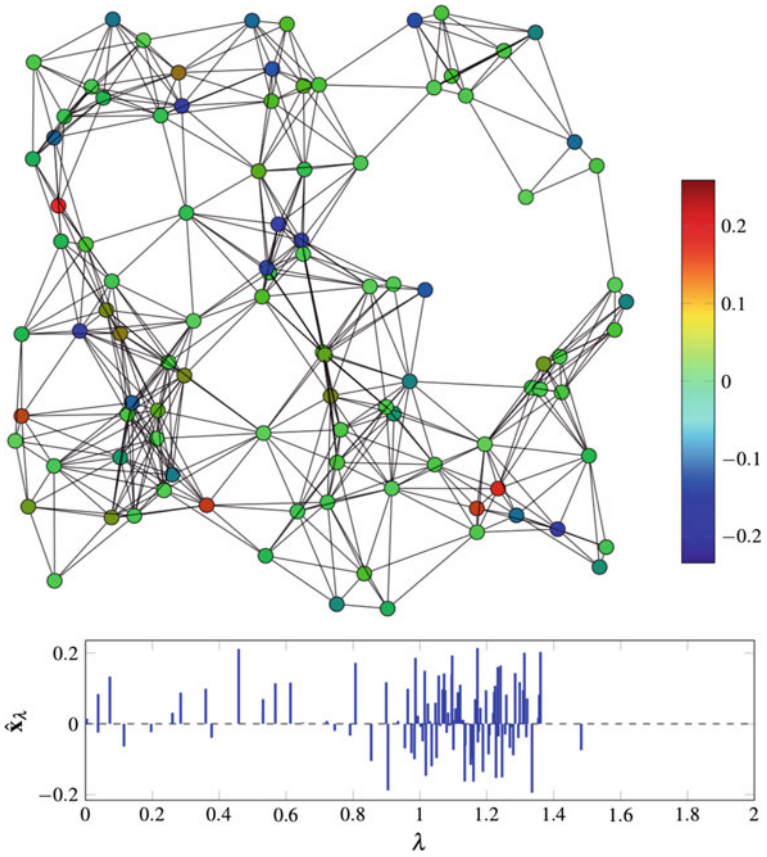


Fig. 4 Example of a random geometric graph on which a uniformly random signal has been initialized (top). The spectrum of this signal in the basis of the normalized Laplacian is also given (bottom). We use the notation $\widehat{\mathbf{x}}_\lambda$ to describe the signal amplitude at eigenvalue λ

5 Graph Uncertainty Principle

5.1 Classical Uncertainty Principle

In classical signal processing, the uncertainty principle states that a signal cannot be perfectly localized both in time and frequency [10]. More precisely, define the variance of a measurable function f as:

$$v(f) = \inf_{a \in \mathbb{R}} \int (x-a)^2 df \ . \tag{31}$$

Introduce the Fourier transform $\widehat{f}$ of f as:

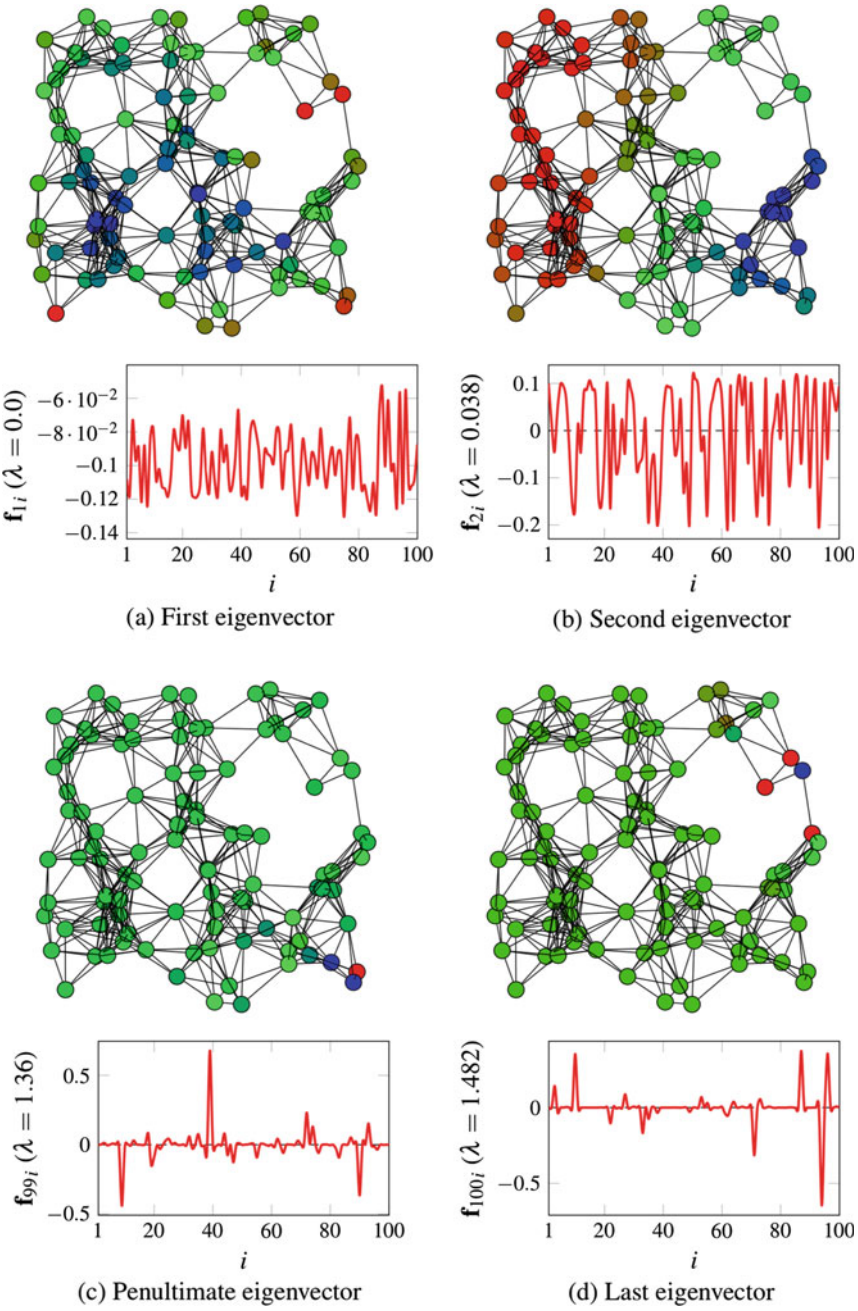
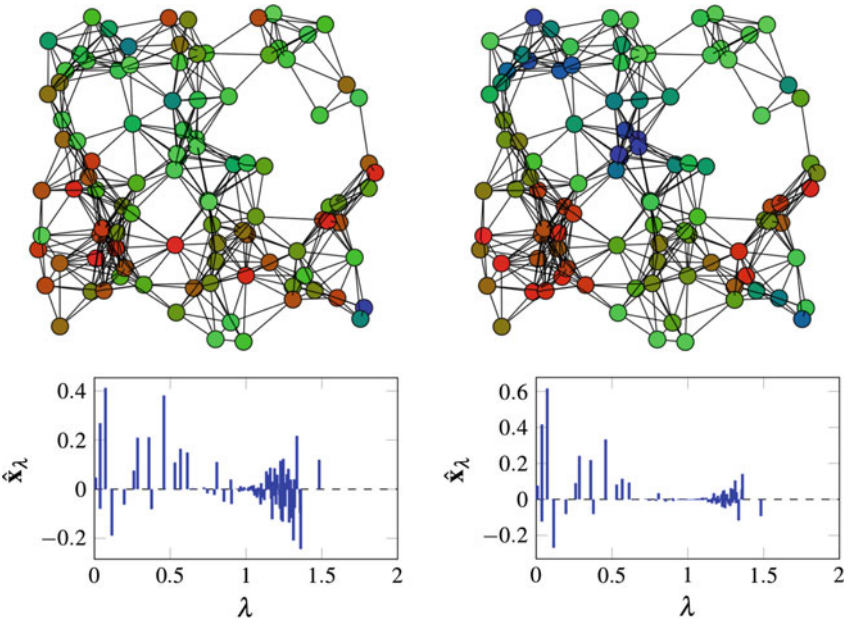
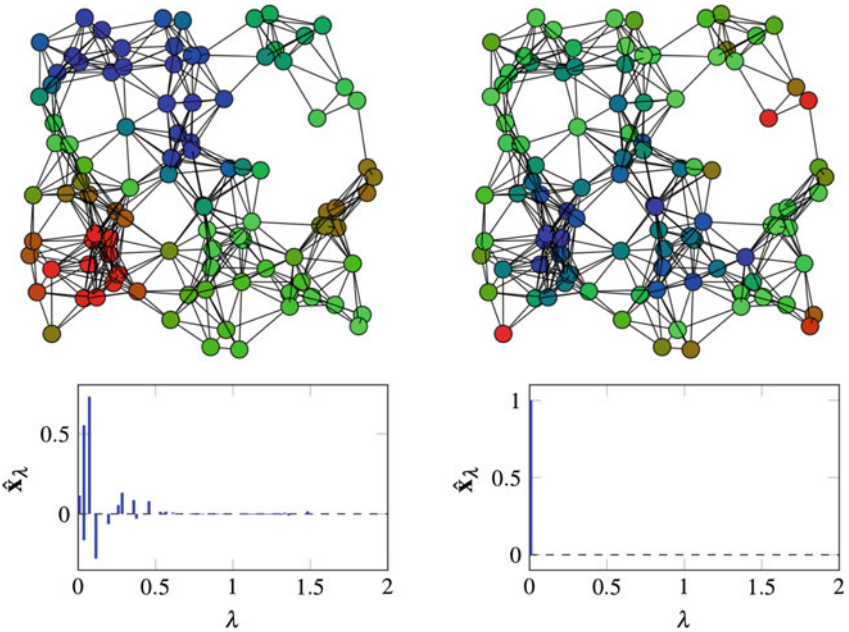


Fig. 5 Representation of some of the eigenvectors of the normalized Laplacian (bottom) associated with the graph in Fig. 4. Eigenvectors associated with lower eigenvalues correspond to low frequencies, and vary smoothly when used as signals on the graph (top). On the contrary, those associated with large eigenvalues feature some strong local variations when represented in the graph domain



(a) $t = 1$ (b) $t = 2$



(c) $t = 50$ (d) $t = 1{,}000$

◀ **Fig. 6** Examples of diffusions of the signal in Fig. 4, for various values of t, represented both on the graph (top) and in the spectral basis of the normalized Laplacian (bottom). As t increases, spectral components of the signal associated with eigenvalues close to 1 vanish faster than others. Eventually, as t grows to infinity, only the signal spectrum associated with eigenvalue 0 remains. As a consequence, the signal converges to the first eigenvalue of the normalized Laplacian, or its opposite depending on the sign of the initial signal component associated with eigenvalue 0. This can be observed in the present example by comparing the graph signal in **d** with the one in Fig. 5a

$$\widehat{f} : \lambda \mapsto \int f(x) e^{-2\pi i x \lambda} dx \ . \tag{32}$$

Then the uncertainty principle states that:

$$\forall f \in L^2(\mathbb{R}) : \int f = 1 \Rightarrow v(f) v(\widehat{f}) \geq \frac{1}{2} \ . \tag{33}$$

In other words, the variances of a function f and of its Fourier transform cannot both simultaneously be small (since their product is at least $\frac{1}{2}$). Note that the variance of a function tends to zero precisely when the function tends to a (possibly shifted) Dirac delta; i.e., when it is well localized.

An important consequence of the uncertainty principle is related to sampling: a function which is very localized in time necessarily is spread in the frequency domain; i.e., it has a wide bandwidth and thus must be sampled at a higher rate. Put differently, if one samples a very brief portion of a time function f, then one can only hope to reconstruct signals whose spectra are widely spread. Conversely, if one samples a very narrow frequency band, one can only expect to reconstruct signals that are spread in time.

This is a very strong fundamental result. Consequently, an important literature has been developed in the past few years about finding counterparts of this result in the context of graph signal processing. In particular, it is of interest to know when one may hope to sample a graph signal at a few vertices (for instance) and hope to faithfully recover the signal value at other unobserved vertices.

5.2 Graph Spread and Spectral Spread

Throughout this section, we consider the normalized Laplacian and not the Laplacian to define the Graph Fourier Transform of signals. The notions we use are based on those introduced in [1, 2]. We only consider connected graphs.

It is important to note that the uncertainty principle on graphs discussed in this chapter—chosen due to its tied relationship with Heisenberg's uncertainty principle—is only one of multiple proposed definitions in the literature. Different uncertainty principles have been proposed in [17, 20, 21]. Additional results on the

presented uncertainty principle can be found for instance in [5, 15, 16, 18]. Here, we choose to focus on the fundamentals.

The spread of a signal in time is a measure of how different its values are after large delays. As a consequence, measuring the spread of a signal on a graph should take into account the variations of the values together with their relative distance on the graph. Following this lead, we introduce the graph spread.

Definition 20 The graph spread $\delta_{\mathscr{G}}(\mathbf{x})$ of a unit norm signal $\mathbf{x}$ on a connected graph $\mathscr{G}$ is defined by:

$$\delta_{\mathscr{G}}(\mathbf{x}) = \inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} d_{\mathscr{G}}(\{i, j\}) \mathbf{x}_j^2 \ . \tag{34}$$

Note that the graph spread is trivially nonnegative. Also, it can be 0 as stated in the following proposition.

Proposition 9 *The graph spread of a one-hot signal is 0.*

Proof Consider i the vertex where the coordinate of the vector is nonzero.

Proposition 10 *The graph spread of a unit norm signal is upper bounded by the diameter of the graph:*

$$\max_{i,j \in \mathscr{V}} d_{\mathscr{G}}(\{i, j\}) \ . \tag{35}$$

Proof

$$\inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} d_{\mathscr{G}}(\{i, j\}) \mathbf{x}_j^2 \leq \max_{i,j \in \mathscr{V}} d_{\mathscr{G}}(\{i, j\}) \inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} \mathbf{x}_j^2 \ . \tag{36}$$

The spectral spread of a signal is a measure of how localized the spectrum is. Similarly to its definition in the classical case, it should take into account both the spread between frequencies and the corresponding values of the Fourier transform of the signal.

Definition 21 The *spectral spread* of a unit norm signal $\mathbf{x}$ on graph $\mathscr{G}$ is defined as:

$$\widehat{\delta}_{\mathscr{G}}(\mathbf{x}) = \inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} \|\lambda_j - \lambda_i\|_2 \widehat{\mathbf{x}}_j^2 \ . \tag{37}$$

A particular case in the expression of the spectral spread is for $i = 1$ where the expression boils down to that of the sharpness. Thus in general the spectral spread is lesser than the sharpness of the signal, and as a corollary it is upper bounded by 2. It is also trivially lower-bounded by 0.

Definition 22 The uncertainty domain of a connected graph $\mathscr{G}$ is the set:

$$\mathscr{U}_{\mathscr{G}} = \left\{ \left(\delta_{\mathscr{G}}(\mathbf{x}), \widehat{\delta}_{\mathscr{G}}(\mathbf{x}) \right), \|\mathbf{x}\|_2 = 1 \right\} \ . \tag{38}$$

Proposition 11 *The uncertainty domain of a connected graph $\mathcal{G}$ is compact.*

Proof By previous remarks it is bounded. It is also the image of a compact (the sphere of unit norm vectors) by continuous functions so it is closed.

Proposition 12 *Noticeable points of the uncertainty domain are the one-hot vectors, which have zero graph spread and nonzero spectral spread, and eigenvectors of the normalized Laplacian, which have zero spectral spread and nonzero graph spread.*

Proof Consider a one-hot vector. Obviously its graph spread is zero. Its spectral spread cannot be zero as the graph is connected, and thus each row of the normalized Laplacian matrix contains at least two nonzero coordinates.

Conversely, each eigenvector contains at least two nonzero coordinates, and thus has a nonzero graph spread.

Of particular interest is the lower frontier of this compact set that characterizes the extent to which signals can simultaneously achieve low graph and spectral spreads.

Definition 23 The *uncertainty curve* of a connected graph $\mathcal{G}$ is the function

$$u : g \mapsto \inf \left\{ \widehat{\delta}_{\mathcal{G}}(\mathbf{x}), \|\mathbf{x}\|_2 = 1 \wedge \delta_{\mathcal{G}}(\mathbf{x}) = g \right\} , \tag{39}$$

defined on the interval $[0, \delta_{\mathcal{G}}(\lambda_1)]$.

5.3 Example Graphs

5.3.1 Complete Graphs

Proposition 13 *The graph spread of a unit norm signal on a complete graph $\mathcal{G}$ can be written:*

$$\delta_{\mathcal{G}}(\mathbf{x}) = 1 - \max_{i \in \mathcal{V}} \mathbf{x}_i^2 . \tag{40}$$

Proof

$$\begin{aligned} \delta_{\mathcal{G}}(\mathbf{x}) &= \inf_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} d_{\mathcal{G}}(\{i, j\}) \mathbf{x}_j^2 \\ &= \inf_{i \in \mathcal{V}} \sum_{\substack{j \in \mathcal{V} \\ i \neq j}} \mathbf{x}_j^2 \end{aligned} \tag{41}$$

Proposition 14 *The normalized Laplacian of a complete graph has two eigenvalues: 0 and $\frac{N}{N-1}$.*

Proof Denote N the number of vertices of the complete graph. First note that $\mathbf{1}$ is an eigenvector associated with eigenvalue 0. Now consider any vector $\mathbf{x}$ orthogonal to $\mathbf{1}$, that is to say its coordinates sum to 0. we obtain:

$$\begin{aligned}
\text{Ł}\mathbf{x} &= \begin{bmatrix} 1 & -\frac{1}{N-1} & \cdots & -\frac{1}{N-1} \\ -\frac{1}{N-1} & 1 & \cdots & -\frac{1}{N-1} \\ \cdots & \cdots & \cdots & \cdots \\ -\frac{1}{N-1} & -\frac{1}{N-1} & \cdots & 1 \end{bmatrix} \mathbf{x} \\
&= \begin{bmatrix} -\frac{1}{N-1} & -\frac{1}{N-1} & \cdots & -\frac{1}{N-1} \\ -\frac{1}{N-1} & -\frac{1}{N-1} & \cdots & -\frac{1}{N-1} \\ \cdots & \cdots & \cdots & \cdots \\ -\frac{1}{N-1} & -\frac{1}{N-1} & \cdots & -\frac{1}{N-1} \end{bmatrix} \mathbf{x} + \frac{N}{N-1}\mathbf{x} \\
&= \frac{N}{N-1}\mathbf{x}. \qquad (42)
\end{aligned}$$

Proposition 15 *Consider a unit norm signal* $\mathbf{x}$ *and denote* $\alpha_1 = \frac{\mathbf{x}^\top \mathbf{1}}{\sqrt{N}}$. *On the complete graph we have:*

$$\widehat{\delta}_{\mathscr{G}}(\mathbf{x}) = \frac{N}{N-1} \min\{\alpha_1^2, 1-\alpha_1^2\} . \qquad (43)$$

Proof Using Proposition 14 we obtain that:

$$\begin{aligned}
\widehat{\delta}_{\mathscr{G}}(\mathbf{x}) &= \inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} \|\lambda_j - \lambda_i\|_2 \widehat{\mathbf{x}}_j^2 \\
&= \frac{N}{N-1} \min \left\{ \alpha_1^2, \sum_{j=2}^{N} \widehat{\mathbf{x}}_j^2 \right\}, \qquad (44)
\end{aligned}$$

We conclude using the orthonormality of $\mathbf{F}$.

Proposition 16 *For sufficiently large number of vertices* N, *the uncertainty domain of complete graphs can be made arbitrarily close to* $(0, 0)$.

Proof Consider a one-hot vector. Its graph spread is 0 and its spectral spread is $\frac{1}{N-1}$.

Proposition 16 suggests there is no equivalent of the classical uncertainty principle for signals on complete graphs. This is not a surprising result as the complete graph is a degenerate topology in which all elements are identically close to all others.

5.3.2 Star Graphs

Proposition 17 *Normalized Laplacians of star graphs admit only three eigenvalues: 0, 1 and 2.*

Proof Proposition 7 gives us that the first eigenvector of a star graph is a vector containing values $\sqrt{\frac{1}{2N-2}}$ everywhere but at one coordinate where it is $\sqrt{\frac{1}{2}}$. Without loss of generality, let us suppose the coordinate where it is $\sqrt{\frac{1}{2}}$ is 1.

It is trivial to verify that vectors containing a 1 at coordinate $i \geq 2$ and a -1 at coordinate $i+1$ are linearly independent and eigenvectors associated with eigenvalue 1.

Finally, being bipartite, Proposition 6 gives us that 2 is an eigenvalue. We easily check that the corresponding eigenvector is the one that contains $-\sqrt{\frac{1}{2}}$ at coordinate 1 and $\sqrt{\frac{1}{2N-2}}$ everywhere else.

Proposition 18 *For sufficiently large number of vertices N, the uncertainty domain of star graphs can be made arbitrarily close to* $(0, 0)$.

Proof Following the notations introduced in the proof of Proposition 17, we consider a one-hot vector where the 1 is not at coordinate 1 but at some other coordinate j. Its graph spread is thus 0. Also, its Fourier transform is a vector containing $\sqrt{\frac{1}{2N-2}}$ at coordinates 1 and N.

Finally, choosing $j = i$ we obtain that the spectral spread is not greater than $2\sqrt{\frac{1}{2N-2}}$.

Similar to complete graphs, Proposition 18 is not surprising as star graphs also correspond to degenerate topologies.

5.3.3 Ring Graphs

Proposition 19 *Eigenvectors of the normalized Laplacian of ring graphs associated with a nonzero eigenvalue can be chosen as uniformly sampled cosines and sines describing at least one period.*

Proof Denote:

$$\mathbf{x}^k = \begin{pmatrix} \cos(0) \\ \cos\left(\frac{k2\pi}{N}\right) \\ \cos\left(\frac{k4\pi}{N}\right) \\ \cdots \\ \cos\left(\frac{(N-1)k2\pi}{N}\right) \end{pmatrix}. \tag{45}$$

Then:

$$\begin{aligned} \left(\text{Ł}\mathbf{x}^k\right)_i &= 2\cos\left(\frac{i2\pi k}{N}\right) - \cos\left(\frac{(i+1)2\pi k}{N}\right) - \cos\left(\frac{(i-1)2\pi k}{N}\right) \\ &= 2\left(1 - \frac{\cos(2\pi k)}{N}\right)\mathbf{x}_i^k. \end{aligned} \tag{46}$$

A very similar proof can be derived using sines instead of cosines.

Proposition 20 *There exist $N_0 \in \mathbb{Z}$ and $M > 0$ such that for any number of vertices $N \geq N_0$, the uncertainty domain of a ring graph is at least at a distance M from the set $\{(0, 0)\}$.*

Proof Fix $\alpha > 0$ and consider a unit-norm signal $\mathbf{x}$ with graph spread at most α:

$$\begin{aligned} \alpha &\geq \inf_{i \in \mathscr{V}} \sum_{j \in \mathscr{V}} d_{\mathscr{G}}(\{i, j\}) \mathbf{x}_v^2 \\ &\geq \inf_{i \in \mathscr{V}} \sum_{\substack{j \in \mathscr{V} \\ i \neq j}} \mathbf{x}_j^2 \\ &= \inf_{i \in \mathscr{V}} \{1 - \mathbf{x}_i^2\} \end{aligned} \tag{47}$$

Denote i^* a value of i that reaches the minimum of the last quantity:

$$\mathbf{x}_{i^*}^2 \geq 1 - \alpha \ . \tag{48}$$

Now using Proposition 19 we obtain that the largest magnitude value in an eigenvector of a ring graph is of the order of $\frac{1}{\sqrt{N}}$. Thus, each value in $\widehat{\mathbf{x}}$ is at most of the order of $\frac{\sqrt{1-\alpha}+\sqrt{\alpha}}{\sqrt{N}}$. Because $\|\widehat{\mathbf{x}}\|_2 = 1$, we obtain that of the order of N of the coordinates of $\widehat{\mathbf{x}}$ are close to this maximum.

We conclude by observing the span of eigenvalues yielded in Proposition 19 and the definition of $\widehat{\delta}_{\mathscr{G}}(\mathbf{x})$.

So, contrary to the previous examples of the complete graph and the star graph, in the case of a ring graph does exhibit a non-trivial uncertainty principle.

6 Conclusion

We introduced graphs and signals on graphs. We showed there are strong relations that tie signals to the graphs they are defined on. We considered several examples of graphs, either deterministic or randomized.

As when it comes to an uncertainty principle, we showed examples where such a principle holds and examples where it does not.

Better understanding the connections between graphs and this principle could lead to important developments in the field. In particular being able to characterize for which graphs the principle does not hold could allow very the rise of very efficient sampling strategies, able to capture very precise events both in the vertex and in the spectrum domains.

References

1. A. Agaskar, Y. Lu, Uncertainty principles for signals defined on graphs: bounds and characterizations, in *IEEE International Conference on Acoustics, Speech and Signal Processing* (2012)
2. A. Agaskar, Y. Lu, A spectral graph uncertainty principle. IEEE Trans. Inf. Theory **59**, 4338 (2013)
3. F. Bauer, J. Jost, Bipartite and neighborhood graphs and the spectrum of the normalized graph Laplacian (2009), arXiv:0910.3118
4. E. Bayram, P. Frossard, E. Vural, A. Alatan, Analysis of airborne LiDAR point clouds with spectral graph filtering. IEEE Geosci. Remote Sens. Lett. **15**, 1284 (2018)
5. P. Di Lorenzo, S. Barbarossa, P. Banelli, S. Sardellitti, Adaptive least mean squares estimation of graph signals. IEEE Trans. Signal Inf. Process. Over Netw. **2**, 555 (2016)
6. R.M. Gray, *Toeplitz and Circulant Matrices: A Review*. Foundations and Trends in Communications and Information Theory (Now Publishers, Boston, 2006)
7. V. Gripon, Tropical graph signal processing, in *IEEE 51st Asilomar Conference on Signals, Systems and Computers* (2017)
8. J.L. Gross, J. Yellen, *Graph Theory and Its Applications* (Chapman and Hall/CRC, Boca Raton, 2005)
9. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**, 129 (2011)
10. W. Heisenberg, Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. Zeitschrift für Physik **33** (1925)
11. S. Hosseinalipour, J. Wang, H. Dai, W. Wang, Detection of infections using graph signal processing in heterogeneous networks, in *IEEE Global Communications Conference* (2017)
12. W. Huang, T.A.W. Bolton, J.D. Medaglia, D.S. Bassett, A. Ribeiro, D. Van De Ville, A graph signal processing perspective on functional brain imaging, in *Proceedings of the IEEE* (2018)
13. P. Lancaster, M. Tismenetsky, *The Theory of Matrices: With Applications* (Elsevier, Amsterdam, 1985)
14. A. Ortega, P. Frossard, J. Kovačević, J.M.F. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, in *Proceedings of the IEEE* (2018)
15. B. Pasdeloup, R. Alami, V. Gripon, M.G. Rabbat, Toward an uncertainty principle for weighted graphs, in *IEEE 23rd European Signal Processing Conference* (2015)
16. B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, Towards a characterization of the uncertainty curve for graphs, in *IEEE International Conference on Acoustics, Speech and Signal Processing* (2016)
17. N. Perraudin, B. Ricaud, D. Shuman, P. Vandergheynst, Global and local uncertainty principles for signals on graphs. APSIPA Trans. Signal Inf. Process. **7** (2018)
18. M.G. Rabbat, V. Gripon, Towards a spectral characterization of signals supported on small-world networks, in *IEEE International Conference on Acoustics, Speech and Signal Processing* (2014)
19. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**, 83 (2013)
20. O. Teke, P.P. Vaidyanathan, Uncertainty principles and sparse eigenvectors of graphs. IEEE Trans. Signal Process. **65**, 5406 (2017)
21. M. Tsitsvero, S. Barbarossa, P. Di Lorenzo, Signals on graphs: uncertainty principle and sampling. IEEE Trans. Signal Process. **64**, 4845 (2016)

A Filtering Framework for Time-Varying Graph Signals



Addison W. Bohannon, Brian M. Sadler and Radu V. Balan

Abstract Time-varying graph signal processing generalizes scalar graph signals to multivariate time-series data with an underlying graph structure. Important applications include network neuroscience, social network analysis, and sensor processing. In this chapter, we present a framework for modeling the underlying graphs of these multivariate signals along with a filter design methodology based on invariance to the graph-shift operator. Importantly, these approaches apply to directed and undirected graphs. We present three classes of filters for time-varying graph signals, providing example application of each in design of ideal bandpass filters.

1 Introduction

The field of graph signal processing has seen rapid growth since its introduction just five years ago in the works of Sandryhaila and Moura [1, 2] and Shuman et al. [3]. Much of the graph signal processing work can be understood as migrating the theoretical underpinnings of classical signal processing of time-series data to that of scalar signals defined on the nodes of a graph. Not all of this work is within the scope of this chapter. For a comprehensive review of the developments, applications, and open problems in graph signal processing more generally, see Ortega et al. [4].

A. W. Bohannon (✉)
US Army Research Laboratory, Aberdeen Proving Ground, College Park, MD, USA
e-mail: addison.w.bohannon.civ@mail.mil

A. W. Bohannon
Center for Scientific Computation and Mathematical Modeling, University of Maryland, College Park, MD, USA

B. M. Sadler
US Army Research Laboratory, Adelphi, MD, USA
e-mail: brian.m.sadler6.civ@mail.mil

R. V. Balan
Department of Mathematics and Center for Scientific Computation and Mathematical Modeling, University of Maryland, College Park, MD, USA
e-mail: rvbalan@cscamm.umd.edu

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*, Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_10

In the present chapter, we consider the modeling and filtering of time-varying graph signals on possibly directed graphs. Time-varying graph signals can be understood as multivariate stochastic processes for which the multivariate components have a topology induced by a graph structure. Unlike conventional graph signal processing with scalar signals on graphs, time-varying graph signals also incorporate a temporal dimension. Here, we consider how to model the underlying directed (or undirected) graph along with the design of filters for such signals. Our filtering construction is motivated by covariance to the underlying graph shift operator. Target applications for such modeling and filtering are network neuroscience, social network analysis, and sensor array processing. Each of these applications has a physical topology induced by an underlying graph, while the signals are revealed in time-series. For example, the members of a social network comprise the nodes of a graph with edges connecting friends or followers, and the recorded signal could be network activity (e.g. sending messages or updating status) indexed by time.

One notable application of graph signal processing has been its use in deep learning architectures. Deep learning encompasses a diverse field of statistical models and learning algorithms which have achieved state-of-the-art results in image, video, speech, and language applications [5]. The seminal work by Mallat [6, 7] and Bruna and Mallat [8] argued that the power of deep learning to achieve invariances necessary for generalization could be understood as the result of composing conventional wavelet filter banks with non-linear functions. This led to the first work to generalize deep learning to graph signals by Bruna et al. [9]. In this work, the convolution operators in deep learning architectures are replaced with spectrally-defined graph filters consistent with the graph Fourier transform of Shuman et al. [3]. This work inspired a thread of follow-on research [10–12] which is reviewed in detail in Bronstein et al. [13].

At its foundation, the field of graph signal processing has had a fundamental divide. The theory proposed by Sandryhaila and Moura [1, 2] advocates an algebraic construction of graph signal processing from the weighted adjacency matrix. Importantly, this theory accommodates directed graphs. However, the theory proposed by Shuman et al. [3] advocates a construction of graph signal processing from the definition of the graph Fourier transform by means of the graph Laplacian. Importantly, the graph Laplacian for an undirected graph is symmetric and positive semi-definite. This yields an orthonormal basis in the definition of the graph Fourier transform along with imparting a physical intuition for the eigenvectors. Underlying this debate is the lack of a canonical definition for a graph-shift and the consequences of choosing one over another [14]. Later work has advocated that the graph-shift be an isometry [15, 16], a stochastic matrix [17], or have a uniquely-defined orthonormal basis [18]. To add to the confusion, Deri and Moura point-out the inherent ambiguity in choosing eigenvectors from the invariant subspaces of matrices with semi-simple and degenerate eigenvalues [19].

Regarding extensions of graph signal processing to time-varying signals on graphs, there are two primary thrusts. One follows from a desire to track random processes on graphs. Auto-regressive moving average models for tracking time-varying signals on graphs are proposed in [20, 21], and reconstruction techniques

for sub-sampled time-varying graph signals are proposed in [22–25]. The other collection of works relates more closely to the proposed approach of this chapter. These works consider multi-dimensional signals on graphs as factor graphs. Sandryhaila and Moura first proposed this approach in [26] for a general factor graph framework and discussed the application of time-varying signals on a graph. A similar concept was proposed for a generalized analysis of multi-dimensional graph signals recently in [27]. In the interim, Loukas and Foucard and Grassi et al. have proposed a joint Fourier transform and associated analysis of time-varying signals on graphs in [28, 29] respectively.

Other works which address time-varying signals on graphs is that of Villafane-Delgado et al. which considers the tensor decomposition of a Laplacian tensor defined over all time [30]. In Yan et al. [17], a parameterized filter is defined which allows the filtering of continuous time processes on graphs. Smith et al. [31] propose methods by which to define correlation, coherence, and phase-lag index for time-varying signals.

The current chapter shares a common motivation with that of Bruna et al. [9] with regard to designing filters for a machine learning application. It is also similar in its stated goals to the optimal filtering framework of Segarra et al. [32], but for the focus on time-varying graph signals. The definition of shift-invariance used in the current work first appeared in Sandryhaila and Moura [1] without the context of time. The result of Theorem 2 is very similar to notions of graph stationarity proposed in Girault [33] and Marques et al. [34], but stationarity is understood to be a property of the graph signal and not a property of the filter as in the current chapter. Romero et al. [24] propose an extended graph formulation very similar to that introduced in Sect. 2 of the current chapter. The extended graph formulation of Romero et al. is used to define regularization terms for recovering time-varying signals on graphs. However, these works are not primarily concerned with analysis and filtering as is the focus of this chapter. The works of Loukas and Foucard [28] and Grassi et al. [29] are seen to be the most closely related in scope and purpose to the current chapter. These works directly address the analysis and filtering of time-varying signals on graphs. However, these works consider only undirected graphs and use the factor graph approach which does not allow edges across multiple time scales as in the current chapter.

In Sandryhaila and Moura [26], the authors propose a procedure for filtering time-varying graph signals by modeling the underlying graph as one of the various graph products between the circulant shift operator and the weighted adjacency matrix of the graph nodes. This formulation allows the authors to apply both the tools of discrete signal processing and of graph signal processing disjointly on the problem. Similar models have been proposed in Loukas and Foucard [28], Grassi et al. [29], and Kurokawa et al. [27].

Implicit in the factor graph model is a strong Markov assumption about how the nodes of a graph interact in time. The factor graph models only the interaction among nodes at a fixed time-scale. This is not overly limiting, but it misses a more full modeling of the possible interactions among nodes in time. For instance, two nodes could interact via an unmodeled process, such as a node that is not included in the graph. This interaction may manifest over two, three, or more time-steps.

Table 1 Summary of filter design results for $\mathsf{A} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$

Filter type	Complexity	Design parameters
Linear and time-invariant	$\mathscr{O}\left(n^2\right)$	$\left\{\hat{a}_{j,k} \in L^\infty\left([0,1]\right)\right\}_{j,k\in\mathscr{V}}$
Linear and shift-invariant	$\mathscr{O}\left(n\right)$	$\left\{\hat{a}_{k} \in L^\infty\left([0,1]\right)\right\}_{k\in\mathscr{V}}$
Function of graph-shift operator	$\mathscr{O}\left(1\right)$	$\phi : U \to \mathbb{C}$, holomorphic

Simultaneously, other nodes may interact at a single time-step. Additionally, the interaction between nodes may be cumulative or history-dependent. Neither of these cases fit into the factor graph model of time-series graph signals.

This chapter aims to address this gap by modeling stationary interactions between nodes in time. We propose a theoretical framework for linear filtering of time-varying graph signals on directed or undirected time-invariant graphs. This framework yields a family of filter design methods of decreasing design complexity (see Table 1). The proposed filter design methods will be motivated by a common example application of bandpass filtering.

Section 2 covers some preliminaries which establish the extended graph framework and the function spaces in which time-varying graph signals exist. Section 3 adapts linear and time-invariant systems theory to a graph geometry. Using this framework, filter design procedures are derived which yield quadratic complexity. Section 4.2 derives shift-invariant filters and design procedures. Shift-invariance is a graph signal processing concept proposed in Sandryhaila and Moura [1], and it is used here to impart the statistical properties of the graph onto filters. The design complexity of shift-invariant filters is shown to be linear. Section 5 defines shift-invariant filters which have constant learning complexity. Functional calculus is used to define filters which are functions of a given graph-shift operator. In Sect. 6, the previous classes of filters are discussed for the case in which the graph is undirected.

2 Preliminaries

Let $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ be a graph with nodes $\mathscr{V} = \{0, \ldots, n-1\}$ such that $n = |\mathscr{V}| < \infty$ and $\mathscr{E} \subseteq \mathscr{V} \times \mathscr{V}$. A weight function assigns a relationship between any two nodes with an edge connecting them $w : \mathscr{E} \to \mathbb{R}$. The function, w, defines the entries of the adjacency, or weight matrix, W ($[\mathsf{W}]_{j,k} = w(j,k)$). If w is symmetric (i.e. $w(j,k) = w(k,j)$), then the graph is undirected, otherwise it is directed. The degree of each node follows from the definition of W, $d_i = \sum_{j\in\mathscr{V}} w(i,j)$, and the diagonal degree matrix is $\mathsf{D} = diag(d_0, \ldots, d_{n-1})$. The Laplacian of $\mathscr{G}$ is $\mathsf{L} = \mathsf{D} - \mathsf{W}$. If the weight matrix is undirected, then the Laplacian is positive semi-definite. Other matrices of interest are the normalized Laplacian, $\mathsf{L}_n = \mathsf{D}^{-1/2}\mathsf{L}\mathsf{D}^{-1/2}$, and the random walk Laplacian, $\mathsf{L}_r = \mathsf{D}^{-1}\mathsf{L}$.

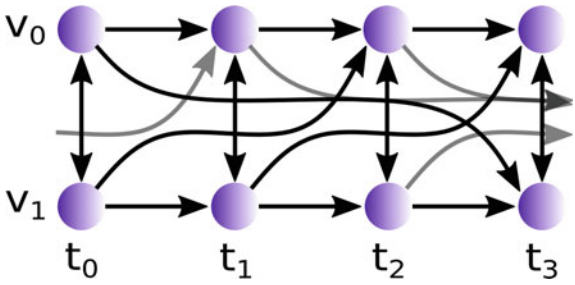


Fig. 1 An example extended graph $\tilde{\mathcal{G}}$. Each time $t \in \mathbb{Z}$ is associated with the node set $\mathcal{V} = \{v_0, v_1\}$ so that each node is indexed by time and space, (v, t). The extended graph admits edges between any two nodes regardless of proximity in time and space

We consider time-varying graph signals which can be represented as functions of time supported on the nodes of a fixed graph. This can be incorporated into the above graph notation by considering an extended graph $\tilde{\mathcal{G}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$ where $\tilde{\mathcal{V}} = \mathbb{Z} \times \mathcal{V}$ and $\tilde{\mathcal{E}} \subseteq \tilde{\mathcal{V}} \times \tilde{\mathcal{V}}$. An example extended graph is depicted in Fig. 1.

Thus, time-varying graph signals are functions on $\tilde{\mathcal{G}}$, equivalently vector-valued sequences indexed by time $\{\mathbf{x}[t] \in \mathbb{C}^n\}_{t\in\mathbb{Z}}$ or scalar functions on $\tilde{\mathcal{V}}$, $x : \mathbb{Z} \times \mathcal{V} \to \mathbb{C}$. We use both notations, the vector-valued sequence and scalar-valued function, as is convenient in context. Some knowledge of function spaces is assumed, namely the definitions and norms of $\ell^p(\mathbb{Z})$ and $L^p([0, 1])$ for $1 \leq p \leq \infty$. The Lebesgue measure is used throughout on L^p-spaces. Some of the analysis will take place in finite dimensions on a function space $\ell^2(\mathcal{V})$ which is isomorphic to $\mathbb{C}^n$ with the usual Euclidean norm, but the primary analysis will occur in two Hilbert spaces, $\ell^2(\mathbb{Z} \times \mathcal{V})$ and $L^2([0, 1] \times \mathcal{V})$, the time and frequency domain of time-varying graph signals.

Definition 1 The $\ell^2(\mathbb{Z} \times \mathcal{V})$-norm is defined as

$$\|x\|_{\ell^2(\mathbb{Z}\times\mathcal{V})} = \left(\sum_{t\in\mathbb{Z}}\sum_{v\in\mathcal{V}} |x(t, v)|^2\right)^{1/2} = \left(\sum_{t\in\mathbb{Z}} \|\mathbf{x}[t]\|^2_{\ell^2(\mathcal{V})}\right)^{1/2},$$

for a function $x : \mathbb{Z} \times \mathcal{V} \to \mathbb{C}$, and it defines the function space

$$\ell^2(\mathbb{Z} \times \mathcal{V}) = \left\{x : \mathbb{Z} \times \mathcal{V} \to \mathbb{C} \mid \|x\|_{\ell^2(\mathbb{Z}\times\mathcal{V})} < \infty\right\}.$$

Moreover, $\|\cdot\|_{\ell^2(\mathbb{Z}\times\mathcal{V})}$ is induced by an inner product

$$\langle x, y\rangle_{\ell^2(\mathbb{Z}\times\mathcal{V})} = \sum_{t\in\mathbb{Z}}\sum_{v\in\mathcal{V}} x(t, v)\bar{y}(t, v) = \sum_{t\in\mathbb{Z}} \langle \mathbf{x}[t], \mathbf{y}[t]\rangle_{\ell^2(\mathcal{V})}$$

where $x, y \in \ell^2(\mathbb{Z} \times \mathcal{V})$.[1]

Definition 2 The $L^2([0, 1] \times \mathcal{V})$-norm is defined as

[1] $\bar{y}$ denotes the complex conjugate of y.

$$\|x\|_{L^2([0,1]\times\mathscr{V})} = \left(\int_{[0,1]} \sum_{v\in\mathscr{V}} |x(\omega, v)|^2 \, d\omega\right)^{1/2} = \left(\int_{[0,1]} \|\mathbf{x}(\omega)\|^2_{\ell^2(\mathscr{V})} \, d\omega\right)^{1/2}$$

for $x : [0, 1] \times \mathscr{V} \to \mathbb{C}$ a measurable function and $d\omega$ is the Lebesgue measure. It defines the function space

$$L^2([0, 1] \times \mathscr{V}) = \left\{x : [0, 1] \times \mathscr{V} \to \mathbb{C} \mid \|x\|_{L^2([0,1]\times\mathscr{V})} < \infty\right\}.$$

Moreover, $\|\cdot\|_{L^2([0,1]\times\mathscr{V})}$ is induced by an inner product

$$\langle x, y\rangle_{L^2([0,1]\times\mathscr{V})} = \int_{[0,1]} \sum_{v\in\mathscr{V}} x(\omega, v)\bar{y}(\omega, v)d\omega = \int_{[0,1]} \langle \mathbf{x}[t], \mathbf{y}[t]\rangle_{\ell^2(\mathscr{V})} \, d\omega$$

where $x, y \in L^2([0, 1] \times \mathscr{V})$.

Fourier analysis plays a central role in signal processing of time-series signals, and it will play an important role in the design and analysis of filters on $\ell^2(\mathbb{Z} \times \mathscr{V})$. Fundamental results from Fourier analysis on $\ell^2(\mathbb{Z})$ and $L^2([0, 1])$ for $1 \le p \le \infty$, including Plancherel's theorem and the convolution theorem carry over to $\ell^2(\mathbb{Z} \times \mathscr{V})$ and $L^2([0, 1] \times \mathscr{V})$ with the following definitions of the Fourier and inverse Fourier transform (see e.g. [35] for reference).

Definition 3 The *Fourier transform* $\mathscr{F}$ of $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ is defined as

$$(\mathscr{F}x)(\omega, v) = \hat{x}(\omega, v) = \sum_{t\in\mathbb{Z}} \mathrm{e}^{2\pi i\omega t} x(t, v),$$

where $\omega \in [0, 1]$.

Definition 4 The *inverse Fourier transform* $\mathscr{F}^*$ of $\mathbf{x} \in L^2([0, 1] \times \mathscr{V})$ is defined as

$$\left(\mathscr{F}^*x\right)(t, v) = \int_{[0,1]} \mathrm{e}^{-2\pi i\omega t} x(\omega, v)d\omega,$$

where $t \in \mathbb{Z}$ and $d\omega$ is the Lebesgue measure.

In the discrete signal processing setting, filtering builds on linear system theory and finite-dimensional linear algebra. These concepts need to be generalized for signals in $\ell^2(\mathbb{Z} \times \mathscr{V})$.

Definition 5 The *operator norm* of $\mathsf{A} : \ell^2(\mathbb{Z} \times \mathscr{V}) \to \ell^2(\mathbb{Z} \times \mathscr{V})$ is defined as

$$\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} = \sup_{\mathbf{x}\in\ell^2(\mathbb{Z}\times\mathscr{V})} \frac{\|\mathsf{A}\mathbf{x}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})}}{\|\mathbf{x}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})}} = \sup_{\|\mathbf{x}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})}=1} \|\mathsf{A}\mathbf{x}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})}.$$

A is said to be *bounded* if $\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} < \infty$. The set of bounded linear transformations is denoted $\mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$.

Bounded operators are more amenable to analysis than unbounded operators in general, and for practical filtering applications, it is presumed that filtered graph signals should be bounded. Moreover, the set of bounded linear transformation $\mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$ with identity E and composition is a Banach algebra [36].

Definition 6 For $\mathsf{A}\in\mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$ and $\mathbf{x},\mathbf{y}\in\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)$, the *adjoint*, $\mathsf{A}^*\in\mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$ satisfies

$$\langle\mathsf{A}\mathbf{x},\mathbf{y}\rangle_{\ell^2(\mathbb{Z}\times\mathscr{V})}=\left\langle\mathbf{x},\mathsf{A}^*\mathbf{y}\right\rangle_{\ell^2(\mathbb{Z}\times\mathscr{V})}.$$

An operator $\mathsf{A}\in\mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$ is said to to be *self-adjoint* if $\mathsf{A}=\mathsf{A}^*$.

There is a natural generalization of finite-dimensional linear operators to infinite-dimensional multiplication operators. Let $\mathbf{x}\in\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)$, then for each $\mathsf{A}\in\mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$, there exists a unique kernel $\mathsf{K}:\mathbb{Z}\times\mathbb{Z}\to\mathscr{B}\left(\ell^2\left(\mathscr{V}\right)\right)$ such that

$$\left(\mathsf{A}\mathbf{x}\right)[t]=\lim_{N\to\infty}\left(\sum_{s=-N}^{N}\mathsf{K}(t,s)\mathbf{x}[s]\right). \tag{1}$$

Note that K is a finite-dimensional linear operator, and recall that $\ell^2\left(\mathscr{V}\right)$ is isomorphic to $\mathbb{C}^n$. It follows that K acts like a matrix on $\mathbb{C}^n$. This intuition then informs the matrix-vector representation of the action of A, a bi-infinite block matrix acting on a bi-infinite vector.

$$\mathsf{A}\mathbf{x}=\begin{bmatrix}\ddots & \vdots & \vdots & \vdots & \\ \cdots & \mathsf{K}(-1,-1) & \mathsf{K}(-1,0) & \mathsf{K}(-1,1) & \cdots \\ \cdots & \mathsf{K}(0,-1) & \mathsf{K}(0,0) & \mathsf{K}(0,1) & \cdots \\ \cdots & \mathsf{K}(1,-1) & \mathsf{K}(1,0) & \mathsf{K}(1,1) & \cdots \\ & \vdots & \vdots & \vdots & \ddots\end{bmatrix}\begin{bmatrix}\vdots \\ \mathbf{x}[-1] \\ \mathbf{x}[0] \\ \mathbf{x}[1] \\ \vdots\end{bmatrix} \tag{2}$$

In linear systems theory, matrix decompositions and the spectra of matrices play an important role. The canonical decomposition in finite dimensions is the Jordan spectral representation which follows (for a proof of the theorem, see e.g. [37]).

Theorem 1 (Jordan spectral representation) *Let* $\mathsf{A}\in\mathscr{B}\left(\ell^2\left(\mathscr{V}\right)\right)$. *Then, there exists* $m\leq n=|\mathscr{V}|$ *distinct eigenvalues* $\{\lambda_k\in\mathbb{C}\}_{1\leq k\leq m}$, *projections* $\left\{\mathsf{P}_k\in\mathscr{B}\left(\ell^2\left(\mathscr{V}\right)\right)\right\}_{1\leq k\leq m}$, *and nilpotents* $\left\{\mathsf{N}_k\in\mathscr{B}\left(\ell^2\left(\mathscr{V}\right)\right)\right\}_{1\leq k\leq m}$ *such that*

$$\mathsf{A}=\sum_{k=0}^{m-1}\lambda_k\mathsf{P}_k+\mathsf{N}_k \tag{3}$$

with the following properties:

1. $\mathsf{P}_j\mathsf{P}_k=\mathsf{P}_k\mathsf{P}_j=\delta_{jk}\mathsf{P}_k$

2. $\mathsf{P}_k \mathsf{N}_k \mathsf{P}_k = \mathsf{N}_k$
3. $(\mathsf{N}_k)^n = 0$
4. $\sum_{k=0}^{m-1} \mathsf{P}_k = \mathsf{E}$ *(the identity operator on* $\ell^2(\mathscr{V})$*).*

Properties 1–4 of Theorem 1 imply additionally that

$$\mathsf{P}_k \mathsf{N}_k = \mathsf{N}_k \mathsf{P}_k = \mathsf{N}_k \tag{4}$$

and

$$\mathsf{P}_j \mathsf{N}_k = \mathsf{N}_k \mathsf{P}_j = \delta_{jk} \mathsf{N}_k. \tag{5}$$

The Jordan spectral representation is perhaps less familiar than the equivalent Jordan normal form, but the projections and nilpotents will be important for the analysis of Sects. 4 and 5.

In the following, filter design will rely heavily on spectral theory which extends the notion of matrix decomposition to infinite-dimensional linear operators on Banach spaces. This theory deals with the spectrum of an operator in place of the eigenvalues of a matrix. Some pertinent definitions follow.

Definition 7 The *resolvent set* of $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ is

$$\rho(\mathsf{A}) = \left\{ z \in \mathbb{C} \mid (\mathsf{A} - z\mathsf{E})^{-1} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right) \right\}.$$

where E denotes the identity operator on $\ell^2(\mathbb{Z} \times \mathscr{V})$.

Definition 8 The *spectrum* of $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ is $\sigma(\mathsf{A}) = \mathbb{C} \setminus \rho(\mathsf{A})$.

Definition 9 The *resolvent* of $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ is the operator-valued function $\mathsf{R} : \rho(\mathsf{A}) \to \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$,

$$\mathsf{R}_{\mathsf{A}}(z) = (A - z\mathsf{E})^{-1}.$$

The relationship between the spectrum of an operator and an eigenvalue of a matrix can be seen from the above definitions. The spectrum is the subset of $\mathbb{C}$ such that $\mathsf{A} - z\mathsf{E}$ is not bijective, equivalent to an eigenvalue in finite dimensions. The spectrum of a bounded operator is closed, bounded, and never empty [36], and the resolvent set is everything except the spectrum. As opposed to the eigenvalues of a matrix, which are a finite set of elements in $\mathbb{C}$, the spectrum of a bounded operator in general has in addition to a point spectrum, a continuous and a residual spectrum [36]. As the eigenvalues of a matrix are a special case of the spectrum of a bounded operator, the spectrum of $\mathsf{A} \in \ell^2(V)$ will also be denoted $\sigma(\mathsf{A})$.

3 Linear and Time-Invariant Filters on $\ell^2(\mathbb{Z} \times \mathscr{V})$

Linear, time-invariant systems theory underlies much of signal processing. With respect to filtering operations, linear time-invariant filters are amenable to analysis and yield well-understood behavior. As discussed in Sect. 2, operators in $\mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ naturally generalize matrices. It remains only to extend the notion of time-invariance.

3.1 Definition of Time-Invariance in $\ell^2(\mathbb{Z} \times \mathscr{V})$

Generalizations of time-invariance exist for signal processing on spatial data (e.g. translation invariance in image processing), but this chapter concerns graph signals with both spatial and time dimensions. Therefore, time-invariance in $\ell^2(\mathbb{Z} \times \mathscr{V})$ should simply extend the definition of time-invariance from $\ell^2(\mathbb{Z})$ to $\ell^2(\mathbb{Z} \times \mathscr{V})$. Time-invariant filters by convention means that the entries of the matrix are not functions of time. Then, there is a certain invariance to when the filter is applied. If this definition is deconstructed, it depends first on a notion of time-evolution and second on an invariant action of the operator.

Let time-evolution of a discrete signal $x \in \ell^2(\mathbb{Z})$ be associated with the time-shift operator, $\mathsf{T} \in \mathscr{B}\left(\ell^2(\mathbb{Z})\right)$,

$$(\mathsf{T}\mathbf{x})[t] = \mathbf{x}[t-1]. \tag{6}$$

For finite-dimensional signals, T is a circulant matrix with ones on the sub-diagonal, and it is diagonalized by the discrete Fourier transform [1]. The same operation can be defined for time-varying graph signals for $\mathsf{T} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ and $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$, which acts like the shift operator except for being infinitely block circulant with the multiplicative identity $\mathsf{E} \in \ell^2(\mathscr{V})$ on the sub-diagonal,

$$\mathsf{T} = \begin{bmatrix} \ddots & & & \\ & \mathsf{E} & & \\ & & \mathsf{E} & \\ & & & \mathsf{E} \\ & & & & \ddots \end{bmatrix}. \tag{7}$$

Note that for for $\mathsf{T} \in \mathscr{B}\left(\ell^2(\mathbb{Z})\right)$, $\mathsf{E} = 1$. Thus, the time-shift operation advances not just a scalar signal but the scalar graph signal one step in time.

Consider the action of a linear filter $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z})\right)$ on $x \in \ell^2(\mathbb{Z})$. When we say time-invariant filtering, we mean

$$(\mathsf{A}\mathsf{T}x)[t] = (\mathsf{T}\mathsf{A}x)[t], \tag{8}$$

which is also understood as covariance (or equivariance) to time-shift. If $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ and $\mathsf{T} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$, then the same condition should define time-invariance for graph signals. Thus, for a filter $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$, it must commute with the cyclic group generated by the time shift operator T, $\mathscr{T} = \langle \mathsf{T} \rangle$.

Definition 10 $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ is called *time-invariant* if A is in the commutant of $\mathscr{T}$, $\left\{\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right) \mid \mathsf{A}\mathscr{T} = \mathscr{T}\mathsf{A}\right\}$.

3.2 Laurent Operators on $\ell^2(\mathbb{Z} \times \mathscr{V})$

In finite dimensions, Toeplitz matrices commute with the circulant shift matrix because both are diagonalized by the discrete Fourier transform. These matrices define scalar convolutions. The matrix representation of Eq. (2) makes possible the generalization of Toeplitz operators, and also convolution, to operators on $\ell^2(\mathbb{Z} \times \mathscr{V})$.

Definition 11 Let $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ with kernel $\mathsf{K} : \mathbb{Z} \times \mathbb{Z} \to \mathscr{B}\left(\ell^2(\mathscr{V})\right)$. Then, A is said to be *Laurent* if $\mathsf{K}(s,t) = \mathsf{K}(s+d, t+d)$ for all $d \in \mathbb{Z}$.

Denote $\mathsf{K}_s = \mathsf{K}(s,0)$. Laurent operators generalize convolution operators as can be seen from the action of a Laurent operator,

$$(\mathsf{A}\mathbf{x})[t] = \lim_{N\to\infty} \left(\sum_{t=-N}^{N} \mathsf{K}_{t-s}\mathbf{x}[s] \right), \tag{9}$$

which has a block Toeplitz matrix representation,

$$\mathsf{A}\mathbf{x} = \begin{bmatrix} \ddots & \ddots & \ddots & & & \\ \cdots & \mathsf{K}_1 & \mathsf{K}_0 & \mathsf{K}_{-1} & \cdots & \\ & \cdots & \mathsf{K}_1 & \mathsf{K}_0 & \mathsf{K}_{-1} & \cdots \\ & & \cdots & \mathsf{K}_1 & \mathsf{K}_0 & \mathsf{K}_{-1} \cdots \\ & & & & \ddots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \mathbf{x}[-1] \\ \mathbf{x}[0] \\ \mathbf{x}[1] \\ \vdots \end{bmatrix}. \tag{10}$$

Time-invariant filters are inherently Laurent, and vice-versa, as the following theorem shows.

Theorem 2 $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *is time-invariant if and only if* A *is Laurent.*

Proof $\Leftarrow$

First, let A be Laurent. It is sufficient to show that $\mathsf{A}\mathsf{T}^d = \mathsf{T}^d\mathsf{A}$ for some $d \in \mathbb{Z}$.

$$\begin{aligned}\left(\mathsf{A}\mathsf{T}^d\mathbf{x}\right)[t] &= \lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}_{t-s}\left(\mathsf{T}^d\mathbf{x}\right)[s]\\ &= \lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}_{t-s}\mathbf{x}[s-d]\\ &= \lim_{N\to\infty}\sum_{s'=-N}^{N}\mathsf{K}_{t-(s'+d)}\mathbf{x}[s']\\ &= \mathsf{T}^d\left(\lim_{N\to\infty}\sum_{s'=-N}^{N}\mathsf{K}_{t-s'}\mathbf{x}[s']\right)\\ &= \left(\mathsf{T}^d\mathsf{A}\mathbf{x}\right)[t]\end{aligned}$$

$\Rightarrow$

Now, let A be time-invariant. By Definition 10, A commutes with $\mathscr{T}$. Without loss of generality, choose $\mathsf{T}^d \in \mathscr{T}$ for some $d \in \mathbb{Z}$. It is necessary to show that $\mathsf{K}(t,s) = \mathsf{K}(t+d, s+d)$. Let $\mathbf{x} \in \ell^2(\mathbb{Z}\times\mathscr{V})$.

$$\left(\mathsf{A}\mathsf{T}^d\mathbf{x}\right)[t+d] = \left(\mathsf{T}^d\mathsf{A}\mathbf{x}\right)[t+d]$$

$$\lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}(t+d,s)\left(\mathsf{T}^d\mathbf{x}\right)[s] = \mathsf{T}^d\left(\lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}(t+d,s)\mathbf{x}[s]\right)$$

$$\lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}(t+d,s)\mathbf{x}[s-d] = \lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}(t,s)\mathbf{x}[s]$$

$$\lim_{N\to\infty}\sum_{s'=-N}^{N}\mathsf{K}(t+d,s'+d)\mathbf{x}[s'] = \lim_{N\to\infty}\sum_{s=-N}^{N}\mathsf{K}(t,s)\mathbf{x}[s]$$

By the uniqueness of K, A is Laurent. □

3.3 Design of Linear and Time-Invariant Filters on $\ell^2(\mathbb{Z}\times\mathscr{V})$

Next, we exploit Theorem 2 to develop a filter design procedure. Consider the following.

Theorem 3 *Let* $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)$ *be Laurent with* $\sum_{t\in\mathbb{Z}}\|\mathsf{K}_t\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} < \infty$. *Then,* $\sigma(\mathsf{A}) = \cup_{\omega\in[0,1]}\sigma\left(\hat{\mathsf{A}}(\omega)\right)$ *and*

$$(\mathscr{F}\mathsf{A}\mathbf{x})(\omega) = \hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega) \tag{11}$$

for $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ *where*

$$\hat{\mathsf{A}}(\omega) = \sum_{t\in\mathbb{Z}} \mathrm{e}^{2\pi i\omega t}\mathsf{K}_t. \tag{12}$$

Moreover,

$$\|\mathsf{A}\|_{\mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))} = \max_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right) \tag{13}$$

where $s_{\max}(\cdot)$ *is the maximum singular value.*[2]

Proof Let $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$.

$$\begin{aligned}
(\mathscr{F}\mathsf{A}\mathbf{x})(\omega) &= \sum_{t\in\mathbb{Z}} \mathrm{e}^{2\pi i\omega t}\left(\lim_{N\to\infty}\left(\sum_{s=-N}^{N}\mathsf{K}_{t-s}\mathbf{x}[s]\right)\right)\\
&= \lim_{N\to\infty}\sum_{t\in\mathbb{Z}}\sum_{s=-N}^{N}\mathrm{e}^{2\pi i\omega(t-s)}\mathsf{K}_{t-s}\mathrm{e}^{2\pi i\omega s}\mathbf{x}[s]\\
&= \left(\lim_{N\to\infty}\sum_{t'=-N}^{N}\mathrm{e}^{2\pi i\omega t'}\mathsf{K}_{t'}\right)\left(\sum_{s\in\mathbb{Z}}\mathrm{e}^{2\pi i\omega s}\mathbf{x}[s]\right)\\
&= \left(\lim_{N\to\infty}\sum_{t'=-N}^{N}\mathrm{e}^{2\pi i\omega t'}\mathsf{K}_{t'}\right)(\mathscr{F}\mathbf{x})(\omega)
\end{aligned}$$

The limit exists by absolute summability of the kernels. Moreover, $\hat{\mathsf{A}}$ is norm-continuous for $\omega \in [0, 1]$. This leads to the convention that $\hat{A}$ is diagonalized by the Fourier transform, $\mathsf{A} = \mathscr{F}^*\hat{A}\mathscr{F}$. The spectrum identity is established next. It follows that

$$\mathsf{A} - z\mathsf{E} = \mathscr{F}^*\left(\hat{\mathsf{A}}(\omega) - z\mathsf{E}\right)\mathscr{F}.$$

If $z \in \rho(\mathsf{A})$, then there exists a $\mathsf{B} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ such that $(\mathsf{A} - z\mathsf{E})\mathsf{B} = \mathsf{E}$.

$$\begin{aligned}
(\mathsf{A} - z\mathsf{E})\,\mathsf{B} &= \mathsf{E}\\
\mathscr{F}^*\left(\hat{\mathsf{A}}(\omega) - z\mathsf{E}\right)\mathscr{F}\mathsf{B} &= \mathsf{E}\\
\left(\hat{\mathsf{A}}(\omega) - z\mathsf{E}\right)\hat{\mathsf{B}}(\omega) &= \mathsf{E}
\end{aligned}$$

Let $S = \cup_{\omega\in[0,1]}\sigma\left(\hat{\mathsf{A}}(\omega)\right)$. For $z \notin S$, $\hat{\mathsf{A}}(\omega) - z\mathsf{E}$ is invertible on $\ell^2(\mathscr{V})$, and for $z \in S$, $\hat{\mathsf{A}}(\omega) - z\mathsf{E}$ is not invertible, $S \subseteq \sigma(\mathsf{A}) \subseteq S$. Thus, $\sigma(\mathsf{A}) = S$. Lastly, the

[2] $\sigma(\cdot)$ here denotes the spectrum of an infinite-dimensional linear operator and the spectrum (eigenvalues) of a finite-dimensional matrix.

norm identity is proved.

$$\begin{aligned}
\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} &= \left\|\hat{\mathsf{A}}\right\|_{\mathscr{B}\left(L^2([0,1]\times\mathscr{V})\right)} \\
&= \sup_{\|\hat{\mathbf{x}}\|_{L^2([0,1]\times\mathscr{V})}=1} \left\|\hat{\mathsf{A}}\hat{\mathbf{x}}\right\|_{\ell^2(\mathbb{Z}\times\mathscr{V})} \\
&\leq \sup_{\|\hat{\mathbf{x}}\|_{L^2([0,1]\times\mathscr{V})}=1} \operatorname*{ess\,sup}_{\omega\in[0,1]} \left\|\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)\right\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\
&\leq \operatorname*{ess\,sup}_{\omega\in[0,1]} \sup_{\|\hat{\mathbf{x}}\|_{L^2([0,1]\times\mathscr{V})}=1} \left\|\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)\right\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\
&= \operatorname*{ess\,sup}_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right)
\end{aligned}$$

So far, this has only established an inequality, but the equality can be shown to be attained for a particular $\mathbf{x} \in \ell^2(\mathbb{Z}\times\mathscr{V})$. The goal is to find a unit norm function which lumps its mass on the measurable set at which $s_{\max}\left(\hat{\mathsf{A}}\right)$ attains its essential supremum while also being in the invariant subspace associated with the maximum singular value on that set. Let $\Omega \subset [0,1]$ be the set on which $s_{\max}\left(\hat{\mathsf{A}}\right)$ achieves its essential supremum. Then, $\hat{\mathsf{A}}^*(\Omega)\hat{\mathsf{A}}(\Omega)$ has an eigenvector $\mathbf{u}(\omega)$ for $\omega \in \Omega$. $\mathbf{u}$ will certainly not be a continuous function of $\omega \in [0,1]$, but it is possible to find a sequence of continuous functions $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ which converge to $\mathbf{u}$ on Ω and are zero otherwise, for which the equality is achieved. Lastly, since $[0,1]$ is a closed set and $\hat{\mathsf{A}}$ is norm-continuous for $\omega \in [0,1]$, $s_{\max}\left(\hat{\mathsf{A}}(\omega)\right)$ is attained on $[0,1]$.

$$\operatorname*{ess\,sup}_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right) = \max_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right)$$

□

As a result of Theorem 3, linear and time-invariant operators on $\ell^2(\mathbb{Z}\times\mathscr{V})$ are Laurent, these operators A have a spectral form given by Eq. 12, where $\hat{\mathsf{A}} : [0,1] \to \ell^2(\mathscr{V})$ is a finite-dimensional operator-valued function of $\omega \in [0,1]$. The action of A on a signal $\mathbf{x} \in \ell^2(\mathbb{Z}\times\mathscr{V})$ is

$$(\mathsf{A}\mathbf{x})[t] = \left(\mathscr{F}^*\left[\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)\right]\right)[t] \tag{14}$$

where $\hat{\mathbf{x}} \in L^2([0,1]\times\mathscr{V})$. This action also admits a matrix-vector representation

$$(\mathsf{A}\mathbf{x})[t] = \left(\mathscr{F}^*\left\{\begin{bmatrix} \hat{a}_{0,0}(\omega) & \cdots & \hat{a}_{0,n-1}(\omega) \\ \vdots & \ddots & \vdots \\ \hat{a}_{n-1,0}(\omega) & \cdots & \hat{a}_{n-1,n-1}(\omega) \end{bmatrix}\begin{bmatrix} \hat{x}_0(\omega) \\ \vdots \\ \hat{x}_{n-1}(\omega) \end{bmatrix}\right\}\right)[t]. \tag{15}$$

Here, $\hat{\mathsf{A}}$ acts as a transfer function on a vector space $\ell^2(\mathscr{V})$.

The significance of Theorem 2 and Eq. (14) is that the design of linear and time-invariant filters can be carried out in the frequency domain where the action of the operator is a matrix-vector product. Equation (15) shows that there are $\mathscr{O}(n^2)$ parameters in the filter, of which each is a function $\{\hat{a}_{j,k} : [0,1] \to \mathbb{C}\}_{j,k\in\mathscr{V}}$. In fact, a stronger statement can be made about the functions $\hat{a}_{j,k}$, that they must be bounded.

Theorem 4 *Let $\hat{\mathsf{A}} : [0,1] \to \mathscr{B}(\ell^2(\mathscr{V}))$ be a a finite-dimensional operator-valued measurable function with entries $\hat{a}_{j,k} : [0,1] \to \mathbb{C}$ for $j,k \in \mathscr{V}$. Then, $\mathsf{A} = \mathscr{F}^*\hat{\mathsf{A}}\mathscr{F}$ defines a Laurent operator, and $\mathsf{A} \in \mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))$ if and only if $\hat{a}_{j,k} \in L^\infty([0,1])$ for $j,k \in \mathscr{V}$. Furthermore,*

$$\max_{j,k\in\mathscr{V}} \left\|\hat{a}_{j,k}\right\|_{L^\infty([0,1])} \leq \|\mathsf{A}\|_{\mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))} \leq \left(\sum_{j\in\mathscr{V}}\sum_{j\in\mathscr{V}} \left\|\hat{a}_{j,k}\right\|_{L^\infty([0,1])}\right)^{1/2}. \tag{16}$$

Proof By Theorem 2, $\mathsf{A} = \mathscr{F}^*\hat{\mathsf{A}}\mathscr{F}$ is Laurent. Eq. (16) will be proved next. From there, the remainder of the theorem follows easily. It is known that for $\mathsf{A} \in \ell^2(\mathscr{V})$, $\max_{j,k\in\mathscr{V}} \left|a_{j,k}\right| \leq s_{\max}(\mathsf{A}) \leq \|\mathsf{A}\|_F$ (see e.g. [38]). Now, consider $\mathsf{A} = \mathscr{F}^*\hat{\mathsf{A}}\mathscr{F}$.[3]

$$\begin{aligned}
\max_{j,k\in\mathscr{V}} \left\|\hat{a}\right\|^2_{\ell^\infty([0,1))} &= \operatorname*{ess\,sup}_{\omega\in[0,1]} \max_{j,k\in\mathscr{V}} \left|\hat{a}_{j,k}(\omega)\right|^2 \\
&\leq \operatorname*{ess\,sup}_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right) \\
&= \|\mathsf{A}\|^2_{\mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))} \\
&= \operatorname*{ess\,sup}_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right) \\
&\leq \operatorname*{ess\,sup}_{\omega\in[0,1]} \sum_{j\in\mathscr{V}}\sum_{k\in\mathscr{V}} \left|\hat{a}_{j,k}(\omega)\right|^2 \leq \sum_{j\in\mathscr{V}}\sum_{k\in\mathscr{V}} \left\|\hat{a}_{j,k}\right\|^2_{\ell^\infty([0,1))}
\end{aligned}$$

This establishes Eq. (16). Now, if $\hat{a}_{j,k} \in L^\infty([0,1])$ for $j,k \in \mathscr{V}$, then $\|\mathsf{A}\|_{\mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))}$ is bounded. □

3.4 Example

We aim to implement a bandpass filter on a signal $\mathbf{x} \in \ell^2(\mathbb{Z}\times\mathscr{V})$ using the framework of Sect. 3.3. A bandpass filter passes a signal at a specific range of frequencies

[3] Unlike in Theorem 3, $\hat{\mathsf{A}} : [0,1] \to \ell^2(\mathscr{V})$ is not necessarily norm continuous. $\mathscr{F}^* : L^\infty([0,1]) \to \ell^1(\mathbb{Z})$ is not bijective, and K may not be absolutely summable. Thus, $\|\mathsf{A}\|_{\mathscr{B}(\ell^2(\mathbb{Z}\times\mathscr{V}))} = \operatorname{ess\,sup}_{\omega\in[0,1]} s_{\max}\left(\hat{\mathsf{A}}(\omega)\right)$.

and attenuates the signal at frequencies outside of the specified range. Implementing a bandpass filter in this construction looks very similar to a bandpass filter on a scalar time-series signal $\ell^2(\mathbb{Z})$. On $\ell^2(\mathbb{Z} \times \mathscr{V})$, the transfer function should be approximately equal to the multiplicative identity E for the desired range of frequencies and close to the multiplicative zero element $\mathsf{0}$ for those frequencies outside. Let a measurable subset $\Omega \subset [0, 1]$ be the desired frequency range. Then, an ideal bandpass filter $\mathsf{A} = \mathscr{F}^* \mathsf{A} \mathscr{F}$ would be

$$\hat{\mathsf{A}}(\omega) = \begin{cases} \mathsf{E} & \omega \in \Omega \\ \mathsf{0} & o.w. \end{cases}. \tag{17}$$

This would correspond to choosing $\hat{a}_{j,k}(\omega) = \delta_{j,k} \chi_\Omega(\omega)$ for all $j, k \in \mathscr{V}$, where

$$\chi_\Omega(\omega) = \begin{cases} 1 & \omega \in \Omega \\ 0 & o.w. \end{cases}. \tag{18}$$

Since $\chi_\Omega \in L^\infty([0, 1])$, this defines a bounded operator $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ by Theorem 4. The action of A on $\mathbf{x}$ would be

$$(\mathsf{A}\mathbf{x})[t] = \int_\Omega \mathrm{e}^{-2\pi i \omega t} \hat{\mathbf{x}}(\omega) d\omega. \tag{19}$$

In the limit as $d\omega(\Omega) \to 0$ so that $\Omega = \{\omega_0\}$, this would implement an ideal bandpass,[4]

$$(\mathsf{A}\mathbf{x})[t] = \mathrm{e}^{-2\pi i \omega_0 t} \hat{\mathbf{x}}(\omega_0). \tag{20}$$

4 Linear and Shift-Invariant Filters on $\ell^2(\mathbb{Z} \times \mathscr{V})$

To this point, the graph geometry has not informed the design procedure. As argued in Sandryhaila and Moura [1], the weighted adjacency matrix describes the spatial evolution of a signal. It captures the flow rate between nodes or the conditional probabilities between random variables. These are relationships between distinctly spatial and non-temporal elements of the signal. A similar intuition follows from the random walk Laplacian as is claimed in [17], which propagates probability mass between the nodes through a Markov process. Requiring that linear and time-invariant filters additionally be invariant to this spatial evolution offers one way to incorporate the information from the underlying graph. This shift-invariance connotes a spatial meaning as opposed to the temporal one of time-invariance.

[4] This converges only in the distribution sense as it amounts to $\hat{a}_{j,k}(\omega) \to \delta(\omega - \omega_0)$, a non-measurable function.

4.1 Definition of Shift-Invariance in $\ell^2(\mathbb{Z} \times \mathcal{V})$

This section will investigate the design of filters that are invariant to a spatial graph evolution that is also time-invariant. As time-invariance is defined in Sect. 3, shift-invariance can be defined as commuting with a given graph-shift operator. This section will consider an arbitrary graph-shift operator, $\mathsf{S} \in \mathscr{B}\left(\ell^2(\mathcal{V})\right)$ that could be an adjacency matrix or Laplacian and potentially be self-adjoint, unitary, sparse, or any other desired property. Then, shift-invariance would mean commuting with the cyclic group generated by S, $\mathscr{S} = \langle \mathsf{S} \rangle$. Since by Theorem 2, any linear and time-invariant filter must be Laurent, considering time-invariant graph-shift operators (i.e. Laurent) ensures that any shift-invariant filter is also time-invariant (i.e. Laurent). This section proceeds with the following definition for shift-invariance.

Definition 12 $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathcal{V})\right)$ is called *shift-invariant* to $\mathsf{S} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathcal{V})\right)$ Laurent if A is in the commutant of $\mathscr{S} = \langle \mathsf{S} \rangle$,

$$\left\{ \mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathcal{V})\right) \mid \mathsf{A}\mathscr{S} = \mathscr{S}\mathsf{A} \right\}.$$

Laurent graph-shift operators convey a special physical significance. If the graph-shift operator S represents the weighted adjacency matrix or random walk matrix, then the kernel K_{s-t} captures the weights or transition probabilities for nodes at a temporal distance $s - t$. That is $\left[\mathsf{K}_{s-t}\right]_{j,k}$ is the weight or transition probability from node v_j at time s to node v_k at time t, and because the kernel depends only on the temporal distance between nodes, S captures a special stationary process as in Fig. 1.

4.2 Theoretical Results on Shift-Invariant Filters in $\ell^2(\mathbb{Z} \times \mathcal{V})$

The first results on the design of shift-invariant filters follow from standard results on simultaneous diagonalization of matrices [38].

Theorem 5 *Let* $\mathsf{A}, \mathsf{S} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathcal{V})\right)$ *be Laurent operators with Jordan spectral representations given pointwise for a.e.* $\omega \in [0, 1]$ *by*

$$\hat{\mathsf{S}}(\omega) = \sum_{k=0}^{m(\omega)} \lambda_k(\omega)\mathsf{P}_k(\omega) + \mathsf{N}_k(\omega) \tag{21}$$

and

$$\hat{\mathsf{A}}(\omega) = \sum_{j=0}^{p(\omega)} \nu_j(\omega)\mathsf{Q}_j(\omega) + \mathsf{M}_j(\omega) \tag{22}$$

respectively. Then, A *is shift-invariant with respect to* S *if and only if*

$$\mathsf{P}_k(\omega)\mathsf{Q}_j(\omega) = \mathsf{Q}_j(\omega)\mathsf{P}_k(\omega) \tag{23}$$
$$\mathsf{P}_k(\omega)\mathsf{M}_j(\omega) = \mathsf{M}_j(\omega)\mathsf{P}_k(\omega) \tag{24}$$
$$\mathsf{N}_k(\omega)\mathsf{Q}_j(\omega) = \mathsf{Q}_j(\omega)\mathsf{N}_k(\omega) \tag{25}$$
$$\mathsf{N}_k(\omega)\mathsf{M}_j(\omega) = \mathsf{M}_j(\omega)\mathsf{N}_k(\omega) \tag{26}$$

for all $k \in \{0, \ldots, m(\omega)\}$, $j \in \{0, \ldots, p(\omega)\}$, *and* $\omega \in [0, 1]$.

Proof It will help to begin with an intermediate result: if $\mathsf{A}, \mathsf{S} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$ are Laurent, then A commutes with $\mathscr{S}$ if and only if

$$\hat{\mathsf{A}}(\omega)\hat{\mathsf{S}}(\omega) = \hat{\mathsf{S}}(\omega)\hat{\mathsf{A}}(\omega) \tag{27}$$

almost everywhere for $\omega \in [0, 1]$.

$\Leftarrow$

First, assume that Eq. (27) is true. Then,

$$\begin{aligned}
\hat{\mathsf{A}}(\omega)\hat{\mathsf{S}}(\omega)\hat{\mathbf{x}}(\omega) &= \hat{\mathsf{S}}(\omega)\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega) \\
\left(\mathscr{F}^*\hat{\mathsf{A}}(\omega)\hat{\mathsf{S}}(\omega)\hat{\mathbf{x}}(\omega)\right)[t] &= \left(\mathscr{F}^*\hat{\mathsf{S}}(\omega)\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)\right)[t] \\
(\mathsf{AS}\mathbf{x})[t] &= (\mathsf{SA}\mathbf{x})[t]
\end{aligned}$$

It also follows from Eq. (27) that $\hat{\mathsf{A}}(\omega)\mathsf{S}^d(\omega) = \mathsf{S}^d(\omega)\hat{\mathsf{A}}(\omega)$ for any $d \in \mathbb{Z}$. From that, it can be shown that A commutes with any $\mathsf{S}^d \in \mathscr{S}$ by the same argument as for S.

$\Rightarrow$

Now, assume that A commutes with any $\mathsf{S}^d \in \mathscr{S}$.

$$\begin{aligned}
\left(\mathsf{AS}^d\mathbf{x}\right)[t] &= \left(\mathsf{S}^d\mathsf{A}\mathbf{x}\right)[t] \\
\mathscr{F}\left(\mathsf{AS}^d\mathbf{x}\right)[t] &= \mathscr{F}\left(\mathsf{S}^d\mathsf{A}\mathbf{x}\right)[t] \\
\hat{\mathsf{A}}(\omega)\hat{\mathsf{S}}^d(\omega)\hat{\mathbf{x}}(\omega) &= \hat{\mathsf{S}}^d(\omega)\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)
\end{aligned}$$

The intermediate result is proven.

Now, it suffices to prove that $\hat{\mathsf{A}}(\omega)$ commutes with $\hat{\mathsf{S}}(\omega)$ almost everywhere on $\omega \in [0, 1]$ if and only if Eqs. (23), (24), (25), and (26) hold for all $k \in \{0, \ldots, m(\omega)\}$, $j \in \{0, \ldots, p(\omega)\}$, and a.e. $\omega \in [0, 1]$.

$\Leftarrow$

First, assume that the respective projections and nilpotents commute. To make it more readable, the dependence on ω is dropped, but it is to be understood that this condition must hold pointwise almost everywhere for $\omega \in [0, 1]$.

$$\begin{aligned}\hat{\mathsf{A}}\hat{\mathsf{S}} &= \left(\sum_{j=0}^{p} \nu_j \mathsf{Q}_j + \mathsf{M}_j\right)\left(\sum_{k=0}^{m} \lambda_k \mathsf{P}_k + \mathsf{N}_k\right) \\ &= \sum_{j=0}^{p}\sum_{k=0}^{m} \nu_j\lambda_k \mathsf{Q}_j\mathsf{P}_k + \nu_j\mathsf{Q}_j\mathsf{N}_k + \lambda_k\mathsf{M}_j\mathsf{P}_k + \mathsf{M}_j\mathsf{N}_k \\ &= \sum_{j=0}^{p}\sum_{k=0}^{m} \nu_j\lambda_k \mathsf{P}_k\mathsf{Q}_j + \nu_j\mathsf{N}_k\mathsf{Q}_j + \lambda_k\mathsf{P}_k\mathsf{M}_j + \mathsf{N}_k\mathsf{M}_j \\ &= \left(\sum_{k=0}^{m} \lambda_k \mathsf{P}_k + \mathsf{N}_k\right)\left(\sum_{j=0}^{p} \nu_j \mathsf{Q}_j + \mathsf{M}_j\right) \\ &= \hat{\mathsf{S}}\hat{\mathsf{A}}\end{aligned}$$

$\Rightarrow$

Now, assume that $\hat{\mathsf{A}}(\omega)$ and $\hat{\mathsf{S}}(\omega)$ commute almost everywhere on $\omega \in [0, 1]$. Then, the resolvents commute: Let $z_1 \in \rho\left(\hat{\mathsf{A}}(\omega)\right.$ and $z_2 \in \rho\left(\hat{\mathsf{S}}(\omega)\right)$.

$$\begin{aligned}\left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right) &= \hat{\mathsf{A}}(\omega)\hat{\mathsf{S}}(\omega) - z_2\hat{\mathsf{A}}(\omega) - z_1\hat{\mathsf{S}}(\omega) + z_1z_2\mathsf{E} \\ \left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right) &= \hat{\mathsf{S}}(\omega)\hat{\mathsf{A}}(\omega) - z_2\hat{\mathsf{A}}(\omega) - z_1\hat{\mathsf{S}}(\omega) + z_1z_2\mathsf{E} \\ \left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right) &= \left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right)\left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\end{aligned}$$

Now, by taking the inverse of both sides, it follows that the resolvents commute.

$$\begin{aligned}\left[\left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right)\right]^{-1} &= \left[\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right)\left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)\right]^{-1} \\ \left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right)^{-1}\left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)^{-1} &= \left(\hat{\mathsf{A}}(\omega) - z_1\mathsf{E}\right)^{-1}\left(\hat{\mathsf{S}}(\omega) - z_2\mathsf{E}\right)^{-1} \\ \mathsf{R}_{\hat{\mathsf{S}}}(z_2, \omega)\,\mathsf{R}_{\hat{\mathsf{A}}}(z_1, \omega) &= \mathsf{R}_{\hat{\mathsf{A}}}(z_1, \omega)\,\mathsf{R}_{\hat{\mathsf{S}}}(z_2, \omega)\end{aligned}$$

Let $\gamma_1 \in \rho\left(\hat{\mathsf{A}}(\omega)\right)$ be a closed curve that encloses only $\nu_j(\omega)$ and $\gamma_2 \in \rho\left(\hat{\mathsf{S}}(\omega)\right)$ be a closed curve that encloses only $\lambda_k(\omega)$. Let $f_1(z_1)$ and $f_2(z_2)$ be two holomorphic functions on an open set which includes the curves γ_1 and γ_2 respectively. Then

$$\begin{aligned}&\left(-\frac{1}{2\pi i}\right)^2 \oint_{\gamma_1}\oint_{\gamma_2} f_1(z_1)\,f_2(z_2)\mathsf{R}_{\hat{\mathsf{S}}}(z_2, \omega)\,\mathsf{R}_{\hat{\mathsf{A}}}(z_1, \omega)\,dz_1dz_2 \\ &\quad= \left(-\frac{1}{2\pi i}\right)^2 \oint_{\gamma_1}\oint_{\gamma_2} f_1(z_1)\,f_2(z_2)\mathsf{R}_{\hat{\mathsf{A}}}(z_1, \omega)\,\mathsf{R}_{\hat{\mathsf{S}}}(z_2, \omega)\,dz_1dz_2\end{aligned}$$

The order of integration can be interchanged by Fubini's theorem.[5] This allows the integrals to factor.

$$\left(-\frac{1}{2\pi i}\oint_{\gamma_2} f_2(z_2)\mathsf{R}_{\hat{\mathsf{S}}}(z_2,\omega)\,dz_2\right)\left(-\frac{1}{2\pi i}\oint_{\gamma_1} f_1(z_1)\mathsf{R}_{\hat{\mathsf{A}}}(z_1,\omega)\,dz_1\right) \tag{28}$$
$$=\left(-\frac{1}{2\pi i}\oint_{\gamma_1} f_1(z_1)\mathsf{R}_{\hat{\mathsf{A}}}(z_1,\omega)\,dz_1\right)\left(-\frac{1}{2\pi i}\oint_{\gamma_2} f_2(z_2)\mathsf{R}_{\hat{\mathsf{S}}}(z_2,\omega)\,dz_2\right)$$

Recall the functional definition of the projection [37],

$$\mathsf{P}_k(\omega) = -\frac{1}{2\pi i}\oint_{\gamma_k} \mathsf{R}_{\mathsf{S}}(z,\omega)\,dz, \tag{29}$$

and of the nilpotent

$$\mathsf{N}_k(\omega) = -\frac{1}{2\pi i}\oint_{\gamma_k} (z-\lambda_k(\omega))\,\mathsf{R}_{\mathsf{S}}(z,\omega)\,dz. \tag{30}$$

where γ_k is a closed curve around $\lambda_k(\omega)$.

For $f_1 = f_2 = 1$, Eq. (28) produces $\mathsf{P}_k\mathsf{Q}_j = \mathsf{Q}_j\mathsf{P}_k$, that is Eq. (23).

For $f_1(z_1) = z_1 - \nu_j(\omega)$, $f_2 = 1$, Eq. (28) yields $\mathsf{P}_k\mathsf{M}_j = \mathsf{M}_j\mathsf{P}_k$, that is (24).

Similarly, the choice $f_1 = 1$ and $f_2(z_2) = z_2 - \lambda_k(\omega)$ turns Eq. (28) into (25), whereas the choice $f_1(z_1) = z_1 - \nu_j(\omega)$ and $f_2(z_2) = z_2 - \lambda_k(\omega)$ turns Eq. (28) into (26). □

Theorem 5 does not provide a constructive means by which to design shift-invariant filters. It provides a geometric constraint. For practical purposes, Theorem 5 is trivially satisfied by fixing the projections of A to match those of S,

$$\hat{\mathsf{A}}(\omega) = \sum_{k=0}^{m(\omega)} \hat{a}_k(\omega)\mathsf{P}_k(\omega) + \mathsf{N}_k(\omega). \tag{31}$$

This leaves $\mathscr{O}(n)$ design parameters, the eigenvalues of $\hat{\mathsf{A}}(\omega)$ for each $\omega \in [0,1]$. More precisely, the design parameters correspond to the simple eigenvalues of $\hat{\mathsf{S}}$.[6]

However, the complications begin here. In the proof of Theorem 5, the Jordan spectral representation only holds pointwise for $\omega \in [0,1]$. In general, not much can be said about the form of the eigenvalues, projections, and nilpotents for the Jordan spectral representation of an arbitrary graph-shift operator. As was shown in Theorem 4, the entries of $\hat{\mathsf{S}}$ can be any function $L^\infty([0,1])$, and the eigenvalues, projections, and nilpotents are functions of the entries of $\hat{a}_{j,k}$, which can be drawn

[5] The resolvent is an analytic function on the resolvent set [39], which means that the integrals are bounded on γ_1 and γ_2.

[6] If $\hat{\mathsf{S}}$ is degenerate, then there could be greater flexibility in the design of shift-invariant filters.

from a large class of functions that includes many poorly behaved functions. How the eigenvalues and invariant subspaces of an operator-valued function change as a function of some argument falls into the study of perturbation theory [37]. Most results deal in the realm of inequalities and special cases. The effect of perturbations on eigenvalues is better understood, but even for the case in which $\hat{\mathsf{S}}(\omega)$ is a continuous function of ω, there is no guarantee of unique continuous eigenfunctions. The algebraic and geometric multiplicity of eigenvalues can change with perturbations. Moreover, projection operators can blow-up as invariant subspaces collapse and reappear. For further reading, see e.g. [37]. All of this is to say that further analysis on the design of shift-invariant operators must be very deliberate and cautious.

Stronger assumptions about the form of S can lead to more well-behaved eigenvalues, projections, and nilpotents for $\hat{\mathsf{S}}$, namely holomorphicity. These results are captured in the following theorem.

Theorem 6 *Let* $\mathsf{S} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$ *be a Laurent operator such that for* $\varepsilon > 0$ *there exist constants* $c_1, c_2 > 0$ *such that*

$$\|\mathsf{K}_t\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \leq \frac{c_1}{(1+\varepsilon)^t} \tag{32}$$

for all $t > 0$, *and*

$$\|\mathsf{K}_t\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \leq c_2(1-\varepsilon)^t \tag{33}$$

for all $t < 0$. *Then, the analytic continuation of* $\hat{\mathsf{S}}(\omega)$,

$$\hat{\hat{\mathsf{S}}}(z) = \sum_{t=0}^{\infty} z^t \mathsf{K}_t, \tag{34}$$

is a holomorphic matrix-valued function on $U = \{z \in \mathbb{C} \mid 1-\varepsilon < |z| < 1+\varepsilon\}$. *Moreover, there are n holomorphic functions* $\{\lambda_k : U \to \mathbb{C}\}_{k \in \mathscr{V}}$ *with at most algebraic singularities such that*

$$\det\left(\hat{\hat{\mathsf{S}}}(z) - \lambda_k(z)\mathsf{E}\right) = 0 \tag{35}$$

for all $k \in \mathscr{V}$ *and* $z \in U$.

Proof Convergence will be shown element-wise in $\hat{\hat{\mathsf{S}}}$, which yields sequences $\left\{\left[\mathsf{K}_t\right]_{j,k}\right\}_{k \in \mathbb{N}}$. The Laurent expansion converges to a holomorphic function on an annulus $z \in \{z \in \mathbb{C} \mid r < |z| < R\}$ where.

$$R^{-1} = \limsup_{t \to \infty} \left|\left[\mathsf{K}_t\right]_{j,k}\right|^{1/t} \tag{36}$$

for $t > 0$, and

$$r = \limsup_{t \to -\infty} \left| \left[\mathsf{K}_t \right]_{j,k} \right|^{1/t} \tag{37}$$

for $t < 0$ for all $j, k \in \mathscr{V}$. For the stated decay rate of the kernels, this condition is satisfied on $1 - \varepsilon < |z| < 1 + \varepsilon$. The holomorphicity of the spectrum and nature of the singularities follows from Eq. (35), an algebraic equation for which the solutions vary analytically as a function of the elements of $\hat{\hat{\mathsf{S}}}$ (see e.g. [40]). □

Theorem 6 states that for operators with kernels that decay sufficiently fast, $\hat{\mathsf{S}} : [0, 1] \to \mathscr{B}\left(\ell^2(V)\right)$ can be analytically continued on an annulus that includes the torus ($\{z \in \mathbb{C} \mid |z| = 1\}$). This guarantees holomorphicity of $\hat{\mathsf{S}}(\omega) = \hat{\hat{\mathsf{S}}}(\mathrm{e}^{2\pi i \omega})$, which is the restriction of $\hat{\hat{\mathsf{S}}}$ to the torus. That is to say that $\hat{\mathsf{S}}$ is a holomorphic function of $\omega \in [0, 1]$. The significance of this result owes to the considerably stronger results on analytic perturbations of operators [37].

For analytic perturbations of finite-dimensional linear operators, the eigenvalue functions form λ-groups. These groups of eigenvalues are the multi-valued complex functions in the spectrum. For the purposes of this chapter, each λ-group along with any other λ-group it intersects in the complex plane has an associated total projection. Let $\hat{\mathsf{A}}(\omega) \in \mathscr{B}\left(\ell^2(\mathscr{V})\right)$. A projection associated with an eigenvalue $\lambda_k(\omega)$ is defined functionally for a closed curve $\Gamma_{k,\omega} \subset \mathbb{C}$ as

$$\mathsf{P}_k(\omega) = -\frac{1}{2\pi i} \oint_{\Gamma_{k,\omega}} \mathsf{R}_{\hat{\mathsf{A}}(\omega)}(z)\, dz. \tag{38}$$

For total projections, the closed curve is drawn so as to include the entire λ-group and any other intersecting λ-group. The total projection corresponds to the sum of the constituent projections, but the total projection is bounded and holomorphic on the annulus of holomorphy to include exceptional points. See Fig. 2 for further explanation of total projections. These results can be found in [37] and are reviewed here because the next theorem will require the concept of total projections. In the next section, this theory will be adapted for the practical design of filters.

4.3 Design of Linear and Shift-Invariant Filters on $\ell^2(\mathbb{Z} \times \mathscr{V})$

As in Sect. 3, the design of shift-invariant filters can be carried-out in the frequency domain where the action of the operator is the usual matrix-vector product. $\hat{\mathsf{A}}$ acts like a transfer function as in Eq. (14). However, in the case of shift-invariant filters, there are $\mathscr{O}(n)$ parameters because the invariant subspaces are fixed by the graph-shift operator as in Eq. (31),

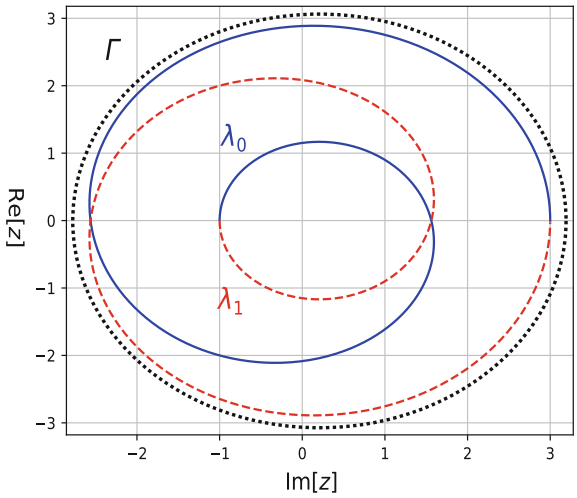


Fig. 2 Total projection of connected components of the spectrum. The spectrum of a bounded operator is the union of compact sets. Multi-valued functions (e.g. $\pm\sqrt{z}$) and odd period functions form λ-groups. The eigenvalues $\lambda_0(\omega) = e^{2\pi i\omega} + 2e^{3\pi i\omega}$ (*solid line*) and $\lambda_1(\omega) = e^{2\pi i\omega} - 2e^{3\pi i\omega}$ (*dashed line*) are plotted in the above figure. λ_0 and λ_1 together form a λ-group. This means that they cannot be partitioned by non-intersecting closed sets and require a single total projection represented by the closed curve Γ (*dotted line*). The intersections $\lambda_0(\omega) = \lambda_1(\omega)$ for $\omega \in [0, 1]$ correspond to algebraic multiplicities

$$(\mathsf{A}\mathbf{x})[t] = \left(\mathscr{F}^*\left[\left(\sum_{k=0}^{m(\omega)} \hat{a}_k(\omega)\mathsf{P}_k(\omega) + \mathsf{N}_k(\omega)\right)\hat{\mathbf{x}}(\omega)\right]\right)[t]. \tag{39}$$

As discussed above, Eq. (39) is in general very difficult to compute. This section will investigate two special cases of graph-shift operators for which concrete results can be stated: $\hat{\mathsf{S}}(\omega)$ constant and analytic.

For the case of $\hat{\mathsf{S}}(\omega)$ constant, it will help to proceed under the assumption that $\hat{\mathsf{S}}(\omega) = \hat{\mathsf{S}}$ has simple eigenvalues.

$$\hat{\mathsf{S}} = \sum_{k\in\mathscr{V}} \lambda_k \mathsf{P}_k \tag{40}$$

Extending the following results to the more general cases of semi-simple and degenerate eigenvalues is straight-froward but requires greater precision in the statement of theorems and proofs. Before proceeding with the theorem, consider the action of such a filter $\mathsf{S} = \mathscr{F}^*\hat{\mathsf{S}}\mathscr{F}$:

$$(\mathsf{S}\mathbf{x})[t] = \hat{\mathsf{S}}\mathbf{x}[t]. \tag{41}$$

S is an infinite-dimensional block diagonal operator with $\hat{\mathsf{S}}$ on the main diagonal. It has kernel function, $\mathsf{K}_0 = \hat{\mathsf{S}}$ and $\mathsf{K}_t = 0$ for $t \neq 0$.

$$\mathsf{S}\mathbf{x} = \begin{bmatrix} \ddots & & & & \\ & \hat{\mathsf{S}} & & & \\ & & \hat{\mathsf{S}} & & \\ & & & \hat{\mathsf{S}} & \\ & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \mathbf{x}[-1] \\ \mathbf{x}[0] \\ \mathbf{x}[1] \\ \vdots \end{bmatrix} \tag{42}$$

This would correspond to a graph-shift operator which acts only in space, and it can be modeled as a factor graph, $\mathsf{S} = I_{\mathbb{Z}} \otimes \hat{\mathsf{S}}$.

Theorem 7 *Let* $\hat{\mathsf{S}} \in \mathscr{B}\left(\ell^2(\mathscr{V})\right)$ *be a constant-valued operator with a Jordan spectral representation given by Eq.* (40) *and* $\mathsf{S} = \mathscr{F}^* \hat{\mathsf{S}} \mathscr{F}$. *Then, for*

$$\hat{A}(\omega) = \sum_{k \in \mathscr{V}} \hat{a}_k(\omega) \mathsf{P}_k, \tag{43}$$

$\mathsf{A} = \mathscr{F}^* \hat{A} \mathscr{F}$ *is shift-invariant to* S, *and* $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *if and only if* $\hat{a}_k \in L^\infty([0,1])$ *for all* $k \in \mathscr{V}$. *Moreover,* $\sigma(\mathsf{A}) = \cup_{\omega \in [0,1]} \left\{\hat{a}_k(\omega) \mid k \in \mathscr{V}\right\}$.

Proof A is shift-invariant to S by Theorem 5.[7]

$\Leftarrow$

Let $\hat{a}_k \in L^\infty([0,1])$ for all $k \in \mathscr{V}$. It is necessary to show that A is bounded.[8]

$$\begin{aligned} \|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} &= \operatorname*{ess\,sup}_{\omega\in[0,1]} \left\|\hat{\mathsf{A}}(\omega)\right\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\ &= \operatorname*{ess\,sup}_{\omega\in[0,1]} \left\|\sum_{k\in\mathscr{V}} \hat{a}_k(\omega)\mathsf{P}_k\right\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\ &\leq \operatorname*{ess\,sup}_{\omega\in[0,1]} \sum_{k\in\mathscr{V}} \left|\hat{a}_k(\omega)\right| \|\mathsf{P}_k\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\ &\leq \sum_{k\in\mathscr{V}} \left\|\hat{a}_k(\omega)\right\|_{L^\infty([0,1])} \|\mathsf{P}_k\|_{\mathscr{B}\left(\ell^2(\mathscr{V})\right)} \\ &< \infty \end{aligned}$$

$\Rightarrow$

Now, suppose that A is bounded and shift-invariant to S. It is necessary to show that $\left\|\hat{a}_k(\omega)\right\|_{\ell^\infty([0,1])} < \infty$ for all $k \in \mathscr{V}$, but suppose instead that it is not true for some $k' \in \mathscr{V}$. The goal is to find a $\|\tilde{\mathbf{x}}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})} = 1$ for which $\|\mathsf{A}\tilde{\mathbf{x}}\|_{\ell^2(\mathbb{Z}\times\mathscr{V})} = \infty$.

[7] Note that $\hat{\mathsf{A}}$ could be degenerate as in Theorem 5 by making $\hat{a}(\omega) = 0$ for some $k \in \mathscr{V}$ and $\omega \in [0,1]$, and this would still be a shift-invariant operator. Also, algebraic multiplicity (i.e. $\hat{a}_j(\omega) = \hat{a}_k(\omega)$ for $j \neq k \in \mathscr{V}$) would not pose a problem as the geometric multiplicity would match that of the algebraic multiplicity.

[8] It is tacitly assumed that the projections of a bounded operator $\hat{\mathsf{S}}$ are bounded.

This can be done by choosing $\hat{\tilde{\mathbf{x}}}$ in the invariant subspace of $\mathsf{P}_{k'}$ and supported only on the set on which $\hat{a}_{k'}$ achieves its essential supremum, $\Omega^* \in [0, 1]$ as shown below.

$$\begin{aligned}\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} &\geq \left\|\sum_{k\in\mathscr{V}} \hat{a}_k \mathsf{P}_k \hat{\tilde{\mathbf{x}}}\right\|_{L^2([0,1]\times\mathscr{V})} \\ &= \left\|\hat{a}_{k'}\right\|_{L^\infty([0,1])} \left\|\mathsf{P}_{k'}\hat{\tilde{\mathbf{x}}}\right\|_{L^2([0,1]\times\mathscr{V})}\end{aligned}$$

Since $\hat{\tilde{\mathbf{x}}}$ was selected arbitrarily in the invariant subspace associated with $\mathsf{P}_{k'}$, let it be chosen to maximize the operator norm. Then,

$$\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} \geq \left\|\hat{a}_{k'}\right\|_{L^\infty([0,1])} \|\mathsf{P}_{k'}\|_{\mathscr{B}\left(L^2([0,1]\times\mathscr{V})\right)},$$

which is not bounded. Then, it is possible to define a sequence of functions which converge to such a $\tilde{\mathbf{x}}$ which contradicts the assumption. The spectrum of A follows immediately from Theorem 3. □

Consider the operator A from Theorem 7. Following Eq. (14), its action on a signal $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ takes a special form.

$$\begin{aligned}(\mathsf{A}\mathbf{x})[t] &= \left(\mathscr{F}^*\left[\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)\right]\right)[t] \\ &= \int_{[0,1]} \mathrm{e}^{-2\pi i\omega t}\hat{\mathsf{A}}(\omega)\hat{\mathbf{x}}(\omega)d\omega \\ &= \int_{[0,1]} \mathrm{e}^{-2\pi i\omega t}\left(\sum_{k\in\mathscr{V}} \hat{a}_k(\omega)\mathsf{P}_k\right)\left(\sum_{s\in\mathbb{Z}} \mathrm{e}^{2\pi i\omega s}\mathbf{x}[s]\right)d\omega \\ &= \sum_{k\in\mathscr{V}}\sum_{s\in\mathbb{Z}}\left(\int_{[0,1]} \mathrm{e}^{2\pi i\omega(s-t)}\hat{a}_k(\omega)d\omega\right)\mathsf{P}_k\mathbf{x}[s] \\ &= \sum_{k\in\mathscr{V}}\left(\sum_{s\in\mathscr{V}} a_k[s-t]\mathsf{P}_k\mathbf{x}[s]\right) \\ &= \sum_{k\in\mathscr{V}}\left(a_k * \mathsf{P}_k\mathbf{x}\right)[t]\end{aligned}$$

Here, $a_k[t] = \left(\mathscr{F}^*\hat{a}_k\right)[t]$. If $\hat{a}_k \in L^2([0, 1]) \cap L^\infty([0, 1])$, then $a_k \in \ell^2(\mathbb{Z})$.[9] The action of A in the time-domain first projects the time-indexed signal $\mathbf{x}[t] \in \ell^2(\mathscr{V})$ onto subspaces defined by the projection operators, P_k and then convolves the signal with a function $a_k \in \ell^2(\mathbb{Z})$. Here, $\hat{a}_k$ acts as transfer function on the projected signal.

Now, shift-invariance with respect to holomorphic operators is considered.

[9] Since $\mathscr{F} : \ell^1(\mathbb{Z}) \to L^\infty([0, 1])$ is not surjective, $\mathscr{F}^*\hat{a}_k$ may not exist for some $\hat{a}_k \in L^\infty([0, 1])$. However, $\mathscr{F} : \ell^2(\mathbb{Z}) \to L^2([0, 1])$ is bijective.

Theorem 8 *Let* $\mathsf{S} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *satisfy the conditions of Theorem 6, and let* $\hat{\mathsf{S}} : [0, 1] \to \ell^2(\mathscr{V})$ *have total projection operators* $\{\mathsf{P}_k(\omega) \mid k = 0, \ldots, m(\omega)\}$ *where* $0 < m(\omega) \leq n = |\mathscr{V}|$. *Then, for*

$$\hat{\mathsf{A}}(\omega) = \sum_{k=0}^{m(\omega)} \hat{a}_k(\omega)\mathsf{P}_k(\omega), \tag{44}$$

$\mathsf{A} = \mathscr{F}^*\hat{\mathsf{A}}\mathscr{F}$ *is shift-invariant to* S *and* $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *if and only if* $\hat{a}_k \in L^\infty([0, 1])$ *for all* $k \in \mathscr{V}$. *Moreover,* $\sigma(\mathsf{A}) = \cup_{\omega\in[0,1]} \left\{\hat{a}_k(\omega) \mid k = 0, \ldots, m(\omega)\right\}$.

Proof The proof follows that of Theorem 7 by the boundedness of the total projections [37]. □

How realistic is it to expect a graph-shift operator which admits an analytic continuation? In almost all practical applications, it would in fact be the case. When modeling the weighted edges of an extended graph from a real-world process, it would be realistic to assume that the impact of events would decay to a negligible effect within finite time or that the conditional probabilities across time would be negligible for large enough time. This would satisfy the conditions of Theorem 6.

4.4 Example

We aim to implement a bandpass filter on a signal $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ using the framework of Sect. 4.3. In Sandryhaila and Moura [2], bandpass filtering on graph signals is defined according to the eigenvectors of the weighted adjacency matrix W and ordered by the so called graph total variation. A low-pass filter allows the projections of the signal onto the subspaces associated with small total variation eigenvectors to pass, and a high-pass filter the projections associated with high total variation eigenvectors. As in Sect. 3.4, a bandpass filter can attenuate unwanted frequencies of the time-series graph signal, but now, a bandpass filter can also attenuate subspaces of the graph signal.

Consider the example of Fig. 1, and let the kernel function be defined as follows:

$$\mathsf{K}_0 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \mathsf{K}_1 = \begin{bmatrix} -\frac{2}{5} & 0 \\ 0 & -\frac{2}{5} \end{bmatrix} \quad \mathsf{K}_2 = \begin{bmatrix} 0 & 0 \\ \frac{4}{5} & 0 \end{bmatrix} \quad \mathsf{K}_3 = \begin{bmatrix} 0 & \frac{2}{5} \\ 0 & 0 \end{bmatrix} \tag{45}$$

and $\mathsf{K}_t = \mathsf{0}$ otherwise, i.e. we assign weights to the edges of Fig. 1. By Theorem 3, this results in a spectral representation of the graph-shift operator,

$$\hat{\mathsf{S}}(\omega) = \begin{bmatrix} -\frac{2}{5}\mathrm{e}^{2\pi i\omega} & 1 + \frac{2}{5}\mathrm{e}^{6\pi i\omega} \\ 1 + \frac{4}{5}\mathrm{e}^{4\pi i\omega} & -\frac{2}{5}\mathrm{e}^{2\pi i\omega} \end{bmatrix}, \tag{46}$$

which is holomorphic on $\omega \in [0, 1]$ by Theorem 6. The spectral operator has eigenvalues,

$$\lambda_{\pm}(\omega) = -\frac{2}{5}e^{2\pi i\omega} \pm \sqrt{\left(1 + \frac{4}{5}e^{4\pi i\omega}\right)\left(1 + \frac{2}{5}e^{6\pi i\omega}\right)}, \tag{47}$$

projection operators,

$$\mathsf{P}_{\pm}(\omega) = \pm\frac{1}{2}\begin{bmatrix} 1 & \pm\sqrt{\frac{5+2e^{6\pi i\omega}}{5+4e^{4\pi i\omega}}} \\ \pm\sqrt{\frac{5+4e^{4\pi i\omega}}{5+2e^{6\pi i\omega}}} & 1 \end{bmatrix}, \tag{48}$$

and nilpotents $\mathsf{N}_{\pm}(\omega) = \mathsf{0}$ respectively.

Now consider the bandpass filter. Suppose that we are to pass the frequency content on $\Omega \subseteq [0, 1]$, and that the graph signal component associated with $\mathsf{P}_{-}(\omega)$ is to be attenuated. Then, an ideal bandpass filter $\mathsf{A} = \mathscr{F}^{*}\hat{\mathsf{A}}\mathscr{F}$ is defined by

$$\hat{\mathsf{A}}(\omega) = \begin{cases} \mathsf{P}_{+}(\omega) & \omega \in \Omega \\ \mathsf{0} & o.w. \end{cases}. \tag{49}$$

This would correspond to choosing $\hat{a}_{+}(\omega) = \chi_{\Omega}(\omega)$ and $\hat{a}_{-}(\omega) = 0$, which are bounded functions on $[0, 1]$. Therefore, $\mathsf{A} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$ by Theorem 8. The action of A would be

$$(\mathsf{A}\mathbf{x})[t] = \int_{\Omega} e^{-2\pi i\omega t}\mathsf{P}_{+}(\omega)\hat{\mathbf{x}}(\omega)d\omega. \tag{50}$$

In the limit as $d\omega(\Omega) \to 0$ so that $\Omega = \{\omega_0\}$, this would implement an ideal bandpass,[10]

$$(\mathsf{A}\mathbf{x})[t] = e^{-2\pi i\omega_0 t}\mathsf{P}_{+}(\omega_0)\hat{\mathbf{x}}(\omega_0). \tag{51}$$

5 Functions of Graph-Shift Operators on $\ell^2(\mathbb{Z} \times \mathscr{V})$

Shift-invariant filters enable the incorporation of prior information in the form of graph-shift operators into the design process. However, linear scaling of the design parameters does not take advantage of sparsity, nor does it scale to large graphs with $|\mathscr{V}| \gg 0$. This begs for an alternative filter design method with greater control on the design complexity.

This section will present the theory for designing filters which are themselves functions of graph-shift operators. In the context of graph signal processing, this concept emerges as defining filters which are polynomials of a given graph-shift operator.

[10] Again, this converges only in the distribution sense as it amounts to $\hat{a}_{+}(\omega) \to \delta(\omega - \omega_0)$, a non-measurable function.

Sandryhaila and Moura [1] observed that in finite-dimensions, all shift-invariant filters can be realized as finite degree polynomials of the graph-shift operator. The concept of defining filters as polynomials of a given graph-shift operator arose also in Shuman, et al. [3] and again in Defferrard, et al. [10]. Defining filters as polynomials of graph-shift operators brings with it a strong physical interpretability as graph operators act only on a neighborhood of each node. Thus, each polynomial order connotes a different physical scale of the action of an operator. Moreover, choosing only the coefficients of polynomials gives the designer much greater control on the complexity of the design problem and allows much of the computationally intensive portions of filtering to be computed offline.

5.1 Functional Calculus

In spectral theory, polynomials of finite-dimensional matrices is only a subset of a more comprehensive theory of functional calculus. The intuition is straight-forward. Power series allow one to arbitrarily approximate scalar functions with infinite polynomials, and polynomials of matrices act on the eigenvalues. In the case of matrices, the Cayley-Hamilton theorem states that for a matrix $\mathsf{A} \in \mathbb{C}^{n\times n}$, all matrices A^k for $k \geq n$ are in the span of $\left\{\mathsf{A}^k \mid k = 0, \ldots, n-1\right\}$. Thus, it is unnecessary to consider polynomials of degree greater than $n-1$ [38]. However, infinite-dimensional operators are not constrained by this result, and admit full power series. Polynomials are dense in the space of continuous functions, so arbitrary degree polynomials of operators can potentially yield a much larger class of functions. Functional calculus provides the theory by which to define operators that are functions (to include polynomials) of other operators. In general, there is the holomorphic functional calculus. For a proof, see e.g. [39].

Theorem 9 (Holomorphic functional calculus) *Let* $\mathsf{S} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$, $U \subset \mathbb{C}$ *be an open set such that* $\sigma\left(\mathsf{S}\right) \subset U$, $\phi : U \to \mathbb{C}$ *be holomorphic, and* $\Gamma \subset U$ *be a closed curve enclosing* $\sigma\left(\mathsf{S}\right)$. *Then, there exists a* $\phi(\mathsf{S}) \in \mathscr{B}\left(\ell^2\left(\mathbb{Z}\times\mathscr{V}\right)\right)$ *such that*

$$\phi(\mathsf{S}) := -\frac{1}{2\pi i}\int_{\Gamma}\phi(z)\mathsf{R}_{\mathsf{S}}\left(z\right)dz. \tag{52}$$

Moreover, $\sigma\left(\phi(\mathsf{S})\right) = \phi\left(\sigma\left(\mathsf{S}\right)\right)$, $\phi \mapsto \phi(\mathsf{S})$ *is a continous map from* $\sup_{\gamma\in\Gamma}|\phi(\gamma)|$ *to* $\|\cdot\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)}$, *and if* $\psi : \mathbb{C} \to \mathbb{C}$ *is holomorphic on* U, *then* $\phi(\mathsf{S})\psi(\mathsf{S}) = (\phi\circ\psi)(\mathsf{S})$.

Hidden in the statement of the theorem is that the open set can be the union of finitely many disjoint open sets so that each connected component of the spectrum could have its own open set $U = \cup_{k=0}^{m} U_k$. Moreover, $\phi : U \to \mathbb{C}$ only needs to be holomorphic on U. That is to say that it must admit a power series representation at all points $z \in U$, but the power series representations need not correspond on disjoint

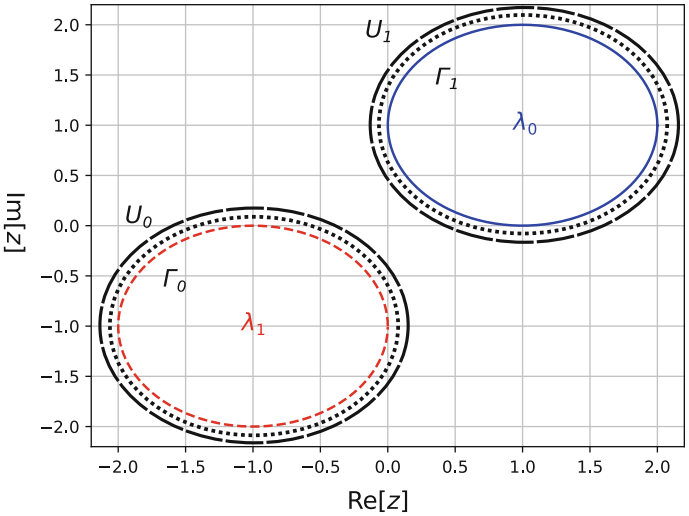


Fig. 3 Example domain for holomorphic functional calculus. Here, the spectrum comprises two disjoint sets $\{\lambda_0(\omega) \in \mathbb{C} : \omega \in [0, 1]\}$ (*solid line*) and $\{\lambda_1(\omega) \in \mathbb{C} : \omega \in [0, 1]\}$ (*dashed line*). A holomorphic function $\phi : U \to \mathbb{C}$ must be holomorphic on an open set $U \subset \mathbb{C}$, but that set can be the union of open sets, i.e. $U = U_0 \cup U_1$ (*long dashed line*) in the figure. Moreover, ϕ only needs to be holomorphic on U, which means that the restriction of ϕ to U_0 and the restriction of ϕ to U_1 can be different holomorphic functions, and they do not have to be holomorphic on $\mathbb{C} \setminus U$. The integration is done on an arbitrary curve $\Gamma \subset U$ that can be the union of curves $\Gamma = \Gamma_0 \cup \Gamma_1$ (*dotted line*)

open sets. The restriction of ϕ to U_0 could be one holomorphic function and another for the restriction of ϕ to U_1. Moreover, it need not have an analytic extension on $\mathbb{C} \setminus (U_0 \cup U_1)$. See Fig. 3 for further explanation.

5.2 *Theoretical Results on Filters that are Functions of Graph-Shift Operators*

Ultimately, the goal of this chapter is to design linear and time-invariant filters for time-varying graph signals. In Sect. 4, it was shown that shift-invariance incorporates spatial information in the form of a graph operator into the design of filters. In this section, the goal is to maintain that information through shift-invariance to a given graph-shift operator.

Theorem 10 *Let* $\mathsf{S} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *be Laurent,* $U \subset \mathbb{C}$ *be an open set such that* $\sigma(\mathsf{S}) \subset U$*, and* $\phi : U \to \mathbb{C}$ *be a holomorphic function. If* $\mathsf{A} = \phi(\mathsf{S})$*, then* $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ *is shift-invariant to* S *where* $\phi(\mathsf{S})$ *is defined according to Theorem 9.*

Proof We begin with shift-invariance. It will help to first prove that S commutes with $\mathsf{R}_{\mathsf{S}}(z)$ for all $z \in \rho(\mathsf{S})$.

$$\begin{aligned}
\mathsf{S}(\mathsf{S} - z\mathsf{E}) &= (\mathsf{S} - z\mathsf{E})\,\mathsf{S} \\
\mathsf{R}_{\mathsf{S}}(z)\,\mathsf{S}(\mathsf{S} - z\mathsf{E})\,\mathsf{R}_{\mathsf{S}}(z) &= \mathsf{R}_{\mathsf{S}}(z)(\mathsf{S} - z\mathsf{E})\,\mathsf{S}\mathsf{R}_{\mathsf{S}}(z) \\
\mathsf{R}_{\mathsf{S}}(z)\,\mathsf{S} &= \mathsf{S}\mathsf{R}_{\mathsf{S}}(z)
\end{aligned}$$

Now, consider shift-invariance

$$\begin{aligned}
\mathsf{A}\mathsf{S} &= \phi(\mathsf{S})\,\mathsf{S} \\
&= \left(\oint_{\Gamma} \phi(z)\mathsf{R}_{\mathsf{S}}(z)\,dz\right)\mathsf{S} \\
&= \oint_{\Gamma} \phi(z)\left(\mathsf{R}_{\mathsf{S}}(z)\,\mathsf{S}\right)dz \\
&= \oint_{\Gamma} \phi(z)\left(\mathsf{S}\mathsf{R}_{\mathsf{S}}(z)\right)dz \\
&= \mathsf{S}\left(\oint_{\Gamma} \phi(z)\mathsf{R}_{\mathsf{S}}(z)\,dz\right) \\
&= \mathsf{S}\phi(\mathsf{S}) \\
&= \mathsf{S}\mathsf{A}
\end{aligned}$$

Lastly, it is necessary to show that $\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} < \infty$.

$$\begin{aligned}
\|\phi(\mathsf{S})\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} &= \left\|\oint_{\Gamma} \phi(z)\mathsf{R}_{\mathsf{S}}(z)\,dz\right\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} \\
&\leq \sup_{z\in\Gamma}|\phi(z)|\left\|\oint_{\Gamma} \mathsf{R}_{\mathsf{S}}(z)\,dz\right\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} \\
&\leq M\sup_{z\in\Gamma}|\phi(z)| \\
&< \infty
\end{aligned}$$

ϕ is bounded by the maximum modulus theorem [41], and the third equality comes from Property 4 of Theorem 1. □

This shows that filters defined as functions of graph-shift operators are indeed shift-invariant; however, Theorem 10 is not constructive. The following theorem provides that.

Theorem 11 *Let* $\mathsf{S} \in \mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)$ *be Laurent,* $U \subset \mathbb{C}$ *be an open set such that* $\sigma(\mathsf{S}) \subset U$*, and* $\phi : U \to \mathbb{C}$ *be a holomorphic function. Further, let* $\hat{\mathsf{S}}$ *have a Jordan spectral representation given pointwise by*

$$\hat{\mathsf{S}}(\omega) = \sum_{k=0}^{m(\omega)} \lambda_k(\omega)\mathsf{P}_k(\omega) + \mathsf{N}_k(\omega) \tag{53}$$

for $\omega \in [0, 1]$ *and* $0 < m(\omega) \leq n$. *Then, for* $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$ *and* $\mathsf{A} = \phi(\mathsf{S})$,

$$(\mathsf{A}\mathbf{x})[t] = \left(\mathscr{F}^* \left\{ \sum_{k=0}^{m(\omega)} \left[(\phi \circ \lambda_k)(\omega)\mathsf{P}_k(\omega) + \left(\phi' \circ \lambda_k\right)(\omega)\mathsf{N}_k(\omega) \right] \hat{\mathbf{x}}(\omega) \right\} \right)[t]. \tag{54}$$

Moreover, $\sigma(\mathsf{A}) = \cup_{\omega \in [0,1]} \{(\phi \circ \lambda_k)(\omega) \mid k = 0, \ldots, m(\omega)\}$.

Proof The Jordan spectral representation of the resolvent for Eq. (53) has the following form:

$$\mathsf{R}_{\hat{\mathsf{S}}}(z, \omega) = \sum_{k=1}^{m(\omega)} (z - \lambda_k(\omega))^{-1} \mathsf{P}_k(\omega) + (z - \lambda_k(\omega))^{-2} \mathsf{N}_k(\omega) \tag{55}$$

for $z \in \rho\left(\hat{\mathsf{S}}(\omega)\right)$ and $\omega \in \mathbb{C}$ [37]. It is helpful to first establish the following identity.

$$\begin{aligned}
(\mathscr{F}\phi(\mathsf{S})\mathbf{x})(\omega) &= \left(\mathscr{F} \left[-\frac{1}{2\pi i} \oint_\Gamma \phi(z)\mathsf{R}_{\mathsf{S}}(z)\, dz \right] \mathbf{x} \right)(\omega) \\
&= \left(-\frac{1}{2\pi i} \oint_\Gamma \phi(z) \left(\mathscr{F}(\mathsf{S} - z\mathsf{E})^{-1} \mathbf{x} \right)(\omega) dz \right) \\
&= \left(-\frac{1}{2\pi i} \oint_\Gamma \phi(z) \left[\left(\hat{\mathsf{S}}(\omega) - z\mathsf{E} \right)^{-1} \hat{\mathbf{x}}(\omega) \right] dz \right) \\
&= \left[\phi\left(\hat{\mathsf{S}}\right) \right](\omega)\hat{\mathbf{x}}(\omega)
\end{aligned}$$

The implication here is that $\phi(\mathsf{S}) = \mathscr{F}^*\phi(\hat{\mathsf{S}})\mathscr{F}$. Then, this along with the following can be used with Eq. (55) to derive the desired result.

$$\begin{aligned}
\phi\left(\hat{\mathsf{S}}\right) &= \frac{1}{2\pi i} \oint_\Gamma \phi(z)\mathsf{R}_{\hat{\mathsf{S}}}(z, \omega)\, dz \\
&= \frac{1}{2\pi i} \oint_\Gamma \phi(z) \left(\sum_{k=1}^{m(\omega)} (z - \lambda_k(\omega))^{-1} \mathsf{P}_k(\omega) + (z - \lambda_k(\omega))^{-2} \mathsf{N}_k(\omega) \right) dz \\
&= \sum_{k=1}^{m(\omega)} \frac{1}{2\pi i} \oint_\Gamma \left[\frac{\phi(z)}{z - \lambda_k(\omega)} \mathsf{P}_k(\omega) + \frac{\phi(z)}{(z - \lambda_k(\omega))^2} \mathsf{N}_k(\omega) \right] dz \\
&= \sum_{k=0}^{m(\omega)} (\phi \circ \lambda_k)(\omega)\mathsf{P}_k(\omega) + \left(\phi' \circ \lambda_k\right)(\omega)\mathsf{N}_k(\omega)
\end{aligned}$$

The spectrum identity follows from Theorem 9, and it is known as the spectral mapping theorem [39]. □

5.3 Design of Filters that are Functions of Graph-Shift Operators

Although constructive, actually computing $\phi(\mathsf{S})$ requires explicit calculation of the spectrum. As discussed in Sect. 4.2, explicit calculation of the spectrum can be prohibitively difficult in all but special cases. Therefore, this section will proceed as in Sect. 4.3 with $\hat{\mathsf{S}}(\omega)$ constant.

Theorem 12 *Let $\hat{\mathsf{S}}(\omega) = \hat{\mathsf{S}} \in \mathscr{B}\left(\ell^2(\mathscr{V})\right)$ be a constant-valued operator with a Jordan spectral representation given by*

$$\hat{\mathsf{S}}(\omega) = \hat{\mathsf{S}} = \sum_{k \in \mathscr{V}} \lambda_k \mathsf{P}_k, \tag{56}$$

and $\mathsf{S} = \mathscr{F}^* \hat{\mathsf{S}} \mathscr{F}$. Further, let $U \subset \mathbb{C}$ be an open set such that $\sigma(\mathsf{S}) \subset U$, $\phi : U \to \mathbb{C}$ be a holomorphic function, $\mathsf{A} = \phi(\mathsf{S})$. Then, for $\mathbf{x} \in \ell^2(\mathbb{Z} \times \mathscr{V})$,

$$(\mathsf{A}\mathbf{x})[t] = \phi\left(\hat{\mathsf{S}}\right)\mathbf{x}[t] = \sum_{k \in \mathscr{V}} \phi(\lambda_k)\mathsf{P}_k \mathbf{x}[t], \tag{57}$$

and $\sigma(\mathsf{A}) = \{\phi(\lambda_k)\}_{k \in \mathscr{V}}$.

Proof This is a straightforward application of Theorem 11.

Theorem 12 indicates that $\phi(\mathsf{S})$ acts independent of time for a graph-shift operator S that acts independent of time as can be seen from the matrix representation.

$$\phi(\mathsf{S})\mathbf{x} = \begin{bmatrix} \ddots & & & & \\ & \phi\left(\hat{\mathsf{S}}\right) & & & \\ & & \phi\left(\hat{\mathsf{S}}\right) & & \\ & & & \phi\left(\hat{\mathsf{S}}\right) & \\ & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \mathbf{x}[-1] \\ \mathbf{x}[0] \\ \mathbf{x}[1] \\ \vdots \end{bmatrix} \tag{58}$$

The design of filters through functions of graph-shift operators admits only $\mathscr{O}(1)$ parameter, the holomorphic function $\phi : U \to \mathbb{C}$. Such operations on graph-shift operators also induce a physical meaning as actions on local neighborhoods. Graph-shifts induce local neighborhoods and pathways on a graph, and higher degree polynomials induce larger neighborhoods and pathways of correspondence between

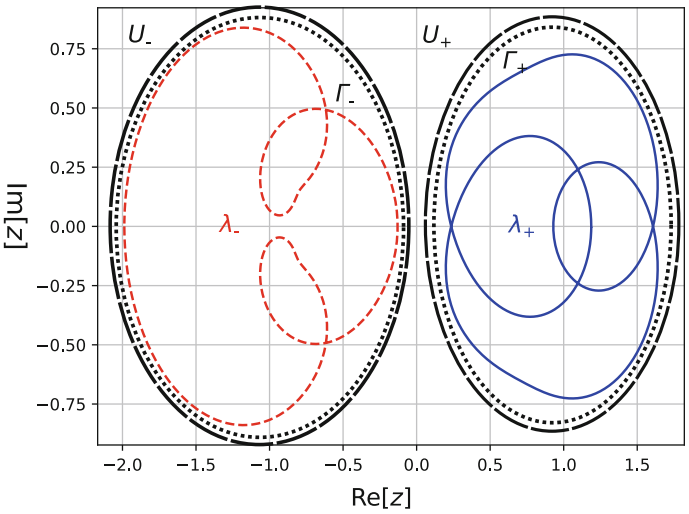


Fig. 4 Spectrum of example S of Fig. 1. The spectrum comprises two disjoint sets, $\{\lambda_+(\omega) \in \mathbb{C} : \omega \in [0, 1]\}$ (*solid line*) and $\{\lambda_-(\omega) \in \mathbb{C} : \omega \in [0, 1]\}$ (*dashed line*). As a result, we can define an open set $U = U_- \cup U_+$ (*long dashed line*) and closed contour $\Gamma = \Gamma_- \cup \Gamma_+$ (*dotted line*). Any holomorphic function $\phi : U \to \mathbb{C}$ need only be holomorphic on U

nodes on a graph. The class of holomorphic function allows the realization of fully local to fully connected operations on the graph to influence the filter.

5.4 Example

Again, we would like to implement a bandpass filter. However, in the case of functional calculus, the design parameter $\phi : U \to \mathbb{C}$ is a function on the spectrum, and it cannot act explicitly on the frequency domain as in the filters of Sects. 3.4 and 4.4. Therefore, more care is required to implement a bandpass filter according to Sect. 5.3.

Consider again the example of Fig. 1 with kernel function given in Sect. 4.4. The first step in designing a filter according to Theorem 11 will be defining an open set $U \subset \mathbb{C}$ such that $\sigma(\mathsf{S}) \subset U$. From Sect. 4.4, $\sigma(\mathsf{S}) = \cup_{\omega\in[0,1]} \{\lambda_+(\omega), \lambda_-(\omega)\}$. As seen in Fig. 4, $\lambda_+(\omega)$ and $\lambda_-(\omega)$ are disjoint on $\mathbb{C}$. This allows us to define $U = U_+ \cup U_-$ to satisfy $U_+ \cap U_- = \emptyset$ and $\cup_{\omega\in[0,1]}\lambda_+(\omega) \subset U_+$ and $\cup_{\omega\in[0,1]}\lambda_-(\omega) \subset U_-$.

Let U be defined as in Fig. 4 and suppose that as in Sect. 4.4, the goal is to attenuate the signal in the subspace of P_-. This can be done by letting, $\phi|_{(U_+)^c} = 0$, a holomorphic function on U_-. Now, consider a complex circular Gaussian,

$$\nu(z; \mu, \sigma) = \frac{1}{2\pi\sigma} \exp\left(-\frac{|z-\mu|^2}{2\sigma^2}\right), \tag{59}$$

with $\mu \in \mathbb{C}$ and $\sigma > 0 \in \mathbb{R}$ [42]. This also defines a holomorphic function, so let $\phi|_{U_+} = \nu(z; \mu, \sigma)$, and let $\mathsf{A} = \phi(\mathsf{S})$. Then, $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ by Theorem 11 and has the following action

$$(\mathsf{A}\mathbf{x})[t] = \int_0^1 \mathrm{e}^{-2\pi i \omega t} \nu(\lambda_+(\omega); \mu, \sigma) \mathsf{P}_+(\omega) \hat{\mathbf{x}}(\omega) d\omega. \tag{60}$$

In the limit, for $\mu = \lambda_+(\omega_0)$ with $\omega_0 \in [0, 1]$ and $\sigma \to 0$, A has the following action[11]

$$(\mathsf{A}\mathbf{x})[t] = \mathrm{e}^{-2\pi i \omega_0 t} \mathsf{P}_+(\omega_0) \hat{\mathbf{x}}(\omega_0). \tag{61}$$

In principle, it is observed that near ideal bandpass is possible using functional calculus.

6 Special Theory for Self-Adjoint Filters

As graphs can be either directed or undirected, methods that work for either enjoy an advantage in terms of universality; however, the analysis of self-adjoint matrices admits significantly stronger results and has led much of the work on graph signal processing to be done on undirected graphs. Thus far, the proposed approach works whether the graph is directed or undirected (or equivalently the graph-shift operator is self-adjoint). This section will briefly discuss the implications of self-adjoint graph-shift operators for Sects. 4 and 5.

The key advantage to analysis of self-adjoint graph-shift operators is the existence of an orthonormal basis from an eigendecomposition. In Sect. 4.2, this will yield semi-simple real-valued eigenvalues. In Sect. 5.2, this will yield a larger class of functions, namely all Borel measurable functions.

It is a common result in matrix analysis that a finite-dimensional self-adjoint operator can be diagonalized by a unitary operator [38]. It then follows that the Jordan spectral representation of a self-adjoint graph-shift operator $\hat{\mathsf{S}} \in \mathscr{B}\left(\ell^2(\mathscr{V})\right)$ can be written

$$\hat{\mathsf{S}} = \sum_{k \in \mathscr{V}} \lambda_k \mathbf{u}_k \mathbf{u}_k^* \tag{62}$$

where $\lambda_k \in \mathbb{R}$ and $\|\mathbf{u}_k\|_{\ell^2(\mathscr{V})} = 1$ for all $k \in \mathscr{V}$.[12] Additionally, the eigenvalues and operator norm of a self-adjoint operator $\mathsf{A} \in \mathscr{B}\left(\ell^2(\mathbb{Z} \times \mathscr{V})\right)$ are related as follows:

$$\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} = \max_{\lambda \in \sigma(\mathsf{A})} |\lambda|. \tag{63}$$

[11] This is not strictly admissible, but holds only in the distribution sense, since ϕ is no longer a measurable holomorphic function.

[12] Equation (62) is no longer strictly unique as it can include repeated eigenvalues and requires the choice of an appropriate basis for any geometric multiplicity.

The maximum of the spectrum is known as the spectral radius, and this follows from Gelfand's theorem [36]. This means that the norm for a shift-invariant filter $\mathsf{A} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$ designed according to Sect. 4.3 for a graph-shift $\mathsf{S} \in \mathscr{B}\left(\ell^2\left(\mathbb{Z} \times \mathscr{V}\right)\right)$ is

$$\|\mathsf{A}\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} = \max_{\omega\in[0,1],k\in\mathscr{V}} \left|\hat{a}_k(\omega)\right|. \tag{64}$$

In Shuman et al. [3], the authors propose that the orthogonal projections defined by the Laplacian substitute as the Fourier basis in a graph Fourier transform. The authors make use of the fact that one can find eigenvectors of the Laplacian L that form an orthonormal basis in $\ell^2\left(\mathscr{V}\right)$. Moreover, the basis vectors correspond to real-valued eigenvalues that can be ordered, from which the basis vectors can in turn be ordered. These results underlie the intuition behind the graph Fourier transform.

Additionally, for self-adjoint operators, there is the Borel functional calculus, which includes an even larger set of functions than the holomorphic functional calculus. For more detail on the Borel functional calculus, see e.g. [43]. The Borel functional calculus admits any Borel measurable function for self-adjoint operators. In addition to allowing a larger set of functions, the Borel functional calculus provides a control on the operator norm of the resultant operator. For a Borel measurable function ϕ,

$$\|\phi(\mathsf{A})\|_{\mathscr{B}\left(\ell^2(\mathbb{Z}\times\mathscr{V})\right)} = \max_{\lambda\in\phi(\sigma(\mathsf{A}))} |\lambda|. \tag{65}$$

7 Conclusion

This chapter addressed the filtering of time-varying graph signals for time-invariant extended graphs. The theoretical framework proposed yields three families of design methods for time-invariant and shift-invariant filters. Bandpass filtering was used as a design example. Future work will explore the implications of finite sampling in the time domain and pursue applications in statistical inference for social networks and brain imaging.

Acknowledgements Radu Balan was partially supported by the NSF grant DMS-1413249, ARO grant W911NF1610008, and the LTS grants H9823013D00560049 and H9823013D00560052.

References

1. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs. IEEE Trans. Signal Process. **61**(7), 1644–1656 (2013). https://doi.org/10.1109/TSP.2013.2238935
2. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs: frequency analysis. IEEE Trans. Signal Process. **62**(12), 3042–3054 (2014b). https://doi.org/10.1109/TSP.2014.2321121
3. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other

irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013). https://doi.org/10.1109/MSP.2012.2235192
4. A. Ortega, P. Frossard, J. Kovačević, J.M.F. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, in *Proceedings of the IEEE*, 106(5), pp. 808–828, (May 2018). https://doi.org/10.1109/JPROC.2018.2820126
5. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. Nature **521**(7553), 436–444 (2015)
6. S. Mallat, Group invariant scattering. Commun. Pure Appl. Math. **65**(10), 1331–1398 (2012). https://doi.org/10.1002/cpa.21413. https://doi.org/10.1002/cpa.21413/abstract, http://onlinelibrary.wiley.com.proxy-um.researchport.umd.edu/
7. S. Mallat, Understanding deep convolutional networks. Philos. Trans. R. Soc. A **374**(2065) (2016). https://doi.org/10.1098/rsta.2015.0203. http://rsta.royalsocietypublishing.org/content/374/2065/20150203
8. J. Bruna, S. Mallat, Invariant scattering convolution networks. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1872–1886 (2013). https://doi.org/10.1109/TPAMI.2012.230
9. J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, in *Proceedings of the International Conference on Learning Representations 2014* (Banff, Canada, 2014). http://arxiv.org/abs/1312.6203
10. M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in *Advances in Neural Information Processing Systems 29, Curran Associates, Inc.*, ed. by D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (2016), pp. 3844–3852. http://papers.nips.cc/paper/6081-convolutional-neural-networks-on-graphs-with-fast-localized-spectral-filtering.pdf
11. M. Henaff, J. Bruna, Y. LeCun, Deep convolutional networks on graph-structured data (2015). arXiv:150605163 [cs]
12. T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in *Proceedings of the International Conference on Learning Representations 2017* (Toulon, France, 2017). http://arxiv.org/abs/1609.02907
13. M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, P. Vandergheynst, Geometric deep learning: going beyond euclidean data. IEEE Signal Process. Mag. **34**(4), 18–42 (2017). https://doi.org/10.1109/MSP.2017.2693418
14. J. Lutzeyer, A. Walden, Comparing graph spectra of adjacency and laplacian matrices (2017). arXiv:171203769
15. A. Gavili, X.P. Zhang, On the shift operator, graph frequency, and optimal filtering in graph signal processing. IEEE Trans. Signal Process. **65**(23), 6303–6318 (2017). https://doi.org/10.1109/TSP.2017.2752689
16. B. Girault, P. Gonalves, E. Fleury, Translation on graphs: an isometric shift operator. IEEE Signal Process. Lett. **22**(12), 2416–2420 (2015). https://doi.org/10.1109/LSP.2015.2488279
17. X. Yan, B.M. Sadler, R.J. Drost, P.L. Yu, K. Lerman, Graph filters and the Z-Laplacian. IEEE J. Sel. Top. Signal Process. **11**(6), 774–784 (2017). https://doi.org/10.1109/JSTSP.2017.2730040
18. S. Sardellitti, S. Barbarossa, P.D. Lorenzo, On the graph Fourier transform for directed graphs. IEEE J. Sel. Top. Signal Process. **11**(6), 796–811 (2017). https://doi.org/10.1109/JSTSP.2017.2726979
19. J.A. Deri, J.M.F. Moura, Spectral projector-based graph Fourier transforms. IEEE J. Sel. Top. Signal Process. **11**(6), 785–795 (2017). https://doi.org/10.1109/JSTSP.2017.2731599
20. A. Loukas, A. Simonetto, G. Leus, Distributed autoregressive moving average graph filters. IEEE Signal Process. Lett. **22**(11), 1931–1935 (2015). https://doi.org/10.1109/LSP.2015.2448655
21. E. Isufi, A. Loukas, A. Simonetto, G. Leus, Autoregressive moving average graph filtering. IEEE Trans. Signal Process. **65**(2), 274–288 (2017b). https://doi.org/10.1109/TSP.2016.2614793
22. E. Isufi, P. Banelli, P. Di Lorenzo, G. Leus, Observing and tracking bandlimited graph processes (2017a). http://arxiv.org/abs/1712.00404, arXiv:171200404 [eess]
23. K. Qiu, X. Mao, X. Shen, X. Wang, T. Li, Y. Gu, Time-varying graph signal reconstruction. IEEE J. Sel. Top. Signal Process. **11**(6), 870–883 (2017). https://doi.org/10.1109/JSTSP.2017.2726969

24. D. Romero, V.N. Ioannidis, G.B. Giannakis, Kernel-based reconstruction of space-time functions on dynamic graphs. IEEE J. Sel. Top. Signal Process. **11**(6), 856–869 (2017). https://doi.org/10.1109/JSTSP.2017.2726976
25. X. Wang, M. Wang, Y. Gu, A distributed tracking algorithm for reconstruction of graph signals. IEEE J. Sel. Top. Signal Process. **9**(4), 728–740 (2015). https://doi.org/10.1109/JSTSP.2015.2403799
26. A. Sandryhaila, J.M.F. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process. Mag. **31**(5), 80–90 (2014a). https://doi.org/10.1109/MSP.2014.2329213
27. T. Kurokawa, T. Oki, H. Nagao, Multi-dimensional graph fourier transform (2017). http://arxiv.org/abs/1712.07811, arXiv:171207811 [cs, stat]
28. A. Loukas, D. Foucard, Frequency analysis of time-varying graph signals, in *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2016), pp. 346–350
29. F. Grassi, A. Loukas, N. Perraudin, B. Ricaud, A time-vertex signal processing framework: scalable processing and meaningful representations for time-series on graphs. IEEE Trans. Signal Process. **66**(3), 817–829 (2018). https://doi.org/10.1109/TSP.2017.2775589
30. M. Villafae-Delgado, S. Aviyente, Dynamic Graph Fourier Transform on temporal functional connectivity networks, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), pp. 949–953. https://doi.org/10.1109/ICASSP.2017.7952296
31. K. Smith, L. Spyrou, J. Escudero, Graph-variate signal analysis: framework and applications (2017). http://arxiv.org/abs/1703.06687, arXiv:170306687 [cs]
32. S. Segarra, A.G. Marques, A. Ribeiro, Optimal graph-filter design and applications to distributed linear network operators. IEEE Trans. Signal Process. **65**(15), 4117–4131 (2017). https://doi.org/10.1109/TSP.2017.2703660
33. B. Girault, Stationary graph signals using an isometric graph translation, in *2015 23rd European Signal Processing Conference (EUSIPCO)* (2015), pp. 1516–1520. https://doi.org/10.1109/EUSIPCO.2015.7362637
34. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Stationary graph processes and spectral estimation. IEEE Trans. Signal Process. **65**(22), 5911–5926 (2017). https://doi.org/10.1109/TSP.2017.2739099
35. T. Krner, *Fourier Analysis* (Cambridge University Press, New York, 1988)
36. E.B. Davies, *Linear Operators and Their Spectra*, vol. 106 (Cambridge University Press, New York, 2007)
37. T. Kato, *Perturbation Theory for Linear Operators*, 2nd edn. (Springer, Berlin, 1995)
38. Golub GH, C.F. Van Loan, *Matrix Computations*, 4th edn. (Johns Hopkins University Press, 2013)
39. N. Dunford, J.T. Schwartz, *Part I: General Theory*, Linear operators (Interscience, New York, 1966)
40. K. Knopp, *Part I. Theory of Functions (English translation)* (Dover, New York, 1947)
41. J. Brown, R. Churchill, *Complex Variables and Applications*, Brown-Churchill series (McGraw-Hill Higher Education, 2004)
42. B. Picinbono, Second-order complex random vectors and normal distributions. IEEE Trans. Signal Process. **44**(10), 2637–2640 (1996). https://doi.org/10.1109/78.539051
43. B. Simon, *Operator Theory, A Comprehensive Course in Analysis*, vol. 4 (American Mathematical Society, 2015)

Vertex-Frequency Energy Distributions



Ljubiša Stanković, Miloš Daković and Ervin Sejdić

Abstract Vertex-varying spectral content on graphs challenge the assumption of vertex invariance, and require vertex-frequency representations to adequately analyze them. The localization window in graph Fourier transform plays a crucial role in this analysis. An analysis of the window functions is presented. The corresponding spectrograms are considered from the energy condition point of view as well. Like in time-frequency analysis, the distribution of signal energy as a function of the vertex and spectral indices is an alternative way to approach vertex-frequency analysis. After an introduction to the second part of this chapter, a local smoothness definition, a definition of an ideal form of the vertex-frequency energy distributions, and two energy forms of the vertex-frequency representations are given. A graph form of the Rihaczek distribution is used as the basic distribution to define a class of reduced interference vertex-frequency energy distributions. These distributions reduce cross-terms effects and satisfy graph signal marginal properties. The theory is illustrated through examples.

1 Introduction

Graph signal processing is a challenging, but rapidly developing, topic. Many real world signals can be considered as graph signals, i.e., signals defined on a graph. Basic and advanced graph signal processing techniques are presented in [1–6]. Some of its applications in biomedical systems [7, 8] and big data analysis [1] are the best examples of its real-world potential.

L. Stanković (✉) · M. Daković
University of Montenegro, Podgorica, Montenegro
e-mail: ljubisa@ac.me

M. Daković
e-mail: milos@ac.me

E. Sejdić
University of Pittsburg, Pittsburg, PA, USA
e-mail: esejdic@ieee.org

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_11

In the case of large signals (graphs), we may not be interested in the analysis of the entire signal, but rather may be interested in its local behavior. A localized signal behavior can be examined via window functions. An exemplary analysis is signal averaging in a local neighborhood. This kind of processing corresponds to low-pass filtering in the classical time-domain signal analysis. Another example could be a classical time-frequency analysis [9–11], where we consider a local signal spectrum. In both examples, window functions are used in order to perform signal localization in time. Window functions are often symmetric, with a single maximum value at a considered time instant. Window functions can be easily shifted in time in order to analyze a signal's behavior at arbitrary time instants.

This concept of signal localization by using window functions can be extended to signals defined on graphs [12–16]. The extension is not straightforward since a simple operation like time shifting cannot be easily defined in a graph signal domain. Several solution approaches for this problem are defined.

A common approach is to utilize the signal spectrum to obtain the window functions for each graph vertex [6]. Another possibility is to define a window support as a local neighborhood for each vertex [16]. The localization window is defined by a set of vertices that contain the current vertex n and all vertices that are close to the vertex n. As in the classical signal analysis, a window should be narrow enough in order to provide good localization of the signal properties but wide enough to produce high resolution.

In this chapter, we will focus on the localized vertex spectrum of a graph signal. The basic concepts and localization methods are analyzed in Sect. 2. Vertex-frequency energy distributions are defined in Sect. 3, while their extension to the reduced interference vertex-frequency distributions is presented in Sect. 4.

2 Localized Graph Fourier Transform

2.1 Graph Fourier Transform

Consider a weighted undirected simple graph with N vertices, where edge weights are denoted by $w_{nm} > 0$ for an edge that connects a vertex n with a vertex m, and $w_{nm} = 0$ if vertices n and m are not connected with an edge. Edge weights are represented in a matrix form as a weight matrix $\mathbf{W}$ whose elements are w_{nm}. The diagonal matrix elements of $\mathbf{W}$ are zeros. The weighting matrix $\mathbf{W}$ is a symmetric matrix since the considered graph is undirected.

The signal $x(n)$, defined at each graph vertex n, is called the graph signal. Signal samples $x(n)$ can be arranged in a $N \times 1$ vector $\mathbf{x} = [x(1), x(2), \ldots, x(N)]^T$.

The graph Laplacian is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{1}$$

where $\mathbf{D}$ is a diagonal degree matrix with $d_{nn} = \sum_{m=1}^{N} w_{nm}$ on the main diagonal. The eigenvalue decomposition of the Laplacian matrix reads

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T, \tag{2}$$

where $\mathbf{U}$ is a matrix of eigenvectors (eigenvector $\mathbf{u}_k$ is the kth column of the matrix $\mathbf{U}$), and $\mathbf{\Lambda}$ is a diagonal matrix with eigenvalues λ_k on the main diagonal. Only simple eigenvalues of multiplicity one are assumed.

The spectrum of a graph signal $\mathbf{x}$ (the graph discrete Fourier transform GDFT) is defined as

$$\mathbf{X} = \text{GDFT}\{\mathbf{x}\} = \mathbf{U}^T\mathbf{x}, \tag{3}$$

where the vector $\mathbf{X}$ contains spectral coefficients associated to the kth eigenvalue and the corresponding eigenvector

$$X(k) = \mathbf{u}_k^T\mathbf{x} = \sum_{n=1}^{N} x(n)u_k(n). \tag{4}$$

The inverse transformation is obtained as $\mathbf{x} = \mathbf{U}\mathbf{X}$, with

$$x(n) = \sum_{k=1}^{N} X(k)u_k(n). \tag{5}$$

Here we will assume that the eigenvalues are of the multiplicity one. Approaches that extend GDFT to directed graphs and graphs with repeated eigenvalues may be found in [17–19].

2.2 Definition of the Localized Vertex Spectrum

The localized vertex spectrum (LVS) on a graph is an extension of the localized time (short time) Fourier transform (STFT). It can be calculated as the spectrum of a signal $x(n)$ multiplied by an appropriate localization window function $h_m(n)$

$$S(m,k) = \sum_{n=1}^{N} x(n)h_m(n)\ u_k(n). \tag{6}$$

It is assumed that the window function $h_m(n)$ should be such that it localizes the signal content around the vertex m. The local vertex spectrum in a matrix notation will be denoted as $\mathbf{S}$. Its elements are $S(m,k)$. The columns of $\mathbf{S}$ are

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

where $\mathbf{x}_{h_m}$ is the vector whose elements $x(n)h_m(n)$ are equal to the signal samples multiplied by the window function centered at the vertex m.

For $h_m(n) = 1$, the localized vertex spectrum is equal to the standard spectrum $S(m, k) = X(k)$ for each m, i.e., no vertex localization is performed. If $h_m(m) = 1$ and $h_m(n) = 0$ for $n \neq m$, the localized vertex spectrum is equal to the signal $S(m, k) = x(m)$ for each k and we do not have any spectral resolution.

Two methods for defining graph localization window functions $h_m(n)$ will be given. First we will present an analysis using the windows $h_m(n)$ defined in the vertex domain. In the next subsection, we will show how to create window functions $h_m(n)$ in the spectral domain.

2.3 Windows Defined Using the Vertex Neighborhood

The window $h_m(n)$ localized at vertex m can be defined using vertex neighborhood. The distance d_{mn} is equal to the length of the shortest walk from vertex m to vertex n. Note that d_{mn} are integers. Then the window function can be defined as

$$h_m(n) = g(d_{mn}),$$

where $g(d)$ corresponds to the basic window function in classical signal processing. For example, the Hann window can be used as

$$h_m(n) = \frac{1}{2}(1 + \cos(\pi d_{mn}/D)), \text{ for } 0 \leq d_{mn} < D,$$

where D is the assumed window width.

Window functions located at each vertex can be calculated in a matrix form. Vertices whose distance is $d_{mn} = 1$ are defined with an adjacency matrix $\mathbf{A}_1 = \mathbf{A}$. The adjacency matrix $\mathbf{A}$ is obtained from the weighting matrix $\mathbf{W}$, using the elements 1 if the vertices are connected and 0 if there is not an edge between the considered vertices. Vertices whose distance is $d_{mn} = 2$ are defined by the following matrix

$$\mathbf{A}_2 = (\mathbf{A} \odot \mathbf{A}_1) \circ (\mathbf{1} - \mathbf{A}_1) \circ (\mathbf{1} - \mathbf{I})$$

where $\odot$ is the logical (Boolean) matrix product, $\circ$ is the Hadamard (element-by-element) product, and $\mathbf{1}$ is a matrix with all ones. Matrix $\mathbf{A} \odot \mathbf{A}_1$ gives the information about all vertices that are connected with walks of length 2 and lower. Element-by-element multiplication by matrix $\mathbf{1} - \mathbf{A}_1$ removes the vertices connected with walks of length 1, while the multiplication by $\mathbf{1} - \mathbf{I}$ removes the diagonal elements.

For $d_{mn} = d \geq 2$ we have a recursive relation for the matrix that will give the information about the vertices at distance d

$$\mathbf{A}_d = (\mathbf{A} \odot \mathbf{A}_{d-1}) \circ (\mathbf{1} - \mathbf{A}_{d-1}) \circ (\mathbf{1} - \mathbf{I}).$$

The window matrix is then formed as

$$\mathbf{P}_D = g(0)\mathbf{I} + g(1)\mathbf{A}_1 + \cdots + g(D-1)\mathbf{A}_{D-1}.$$

The window weighted signal is formed using this matrix as

$$x_m(n) = h_m(n)x(n) = P_D(n,m)x(n).$$

The localized vertex-frequency representation is

$$S(m,k) = \sum_{n=1}^{N} x(n)h_m(n)\ u_k(n) = \sum_{n=1}^{N} x(n)P_D(n,m)\ u_k(n). \tag{7}$$

In matrix form, the above relation reads

$$\mathbf{S} = \mathbf{U}^T(\mathbf{P}_D \circ [\mathbf{x}\ \mathbf{x}\ \ldots\ \mathbf{x}]), \tag{8}$$

where $[\mathbf{x}\ \mathbf{x}\ \ldots\ \mathbf{x}]$ is a $N \times N$ matrix whose columns are signal vectors $\mathbf{x}$.

For a rectangular window $g(d) = 1$ the LVS can be calculated recursively with respect to the window width D as

$$\mathbf{S}_D = \mathbf{S}_{D-1} + \mathbf{U}^T(\mathbf{A}_{D-1} \circ [\mathbf{x}\ \mathbf{x}\ \ldots\ \mathbf{x}]). \tag{9}$$

Example: For numerical illustrations in this chapter we will use a graph and a signal on this graph presented in Fig. 1. The signal consists of parts of three eigenvectors. For the subset $\mathcal{V}_1$ which includes vertices from 1 to 13, eigenvector $k = 20$ is used. For the subset $\mathcal{V}_2$ with vertices 14–26, the signal is equal to eigenvector $k = 52$. Remaining vertices form $\mathcal{V}_3$, and the signal on this subset is equal to eigenvector $k = 36$. Amplitudes of the eigenvectors are scaled as well.

The window and modulated window (kernel) used for local vertex-frequency analysis for the vertices $m = 8$ (top) and $m = 31$ (bottom) and spectral indices $k = 1$ (left) and $k = 6$ (right) are shown in Fig. 2. The local vertex-frequency representation of the graph signal from Fig. 1 is shown in Fig. 3.

Comments

(1) *Weighted distance*: For a weighted graph, the distance from a vertex n to a vertex m could be defined by using the edge weights instead of the number of edges. For example we can define distance as a sum, or product of the associated edge weights. Then d_{nm} may assume non-integer values.

(2) *Interpolation*: Note that for the windowed signal $x(n)h_m(n)$, only $M \leq N$ samples are nonzero. It may be considered a classical zero padded signal. This means that for the reconstruction of this signal, we only need M spectral coefficients $S(n,k)$. The remaining coefficients can be calculated from the system of equations obtained by using the fact that $x(n)h_m(n) = 0$ outside the window support. In the classical

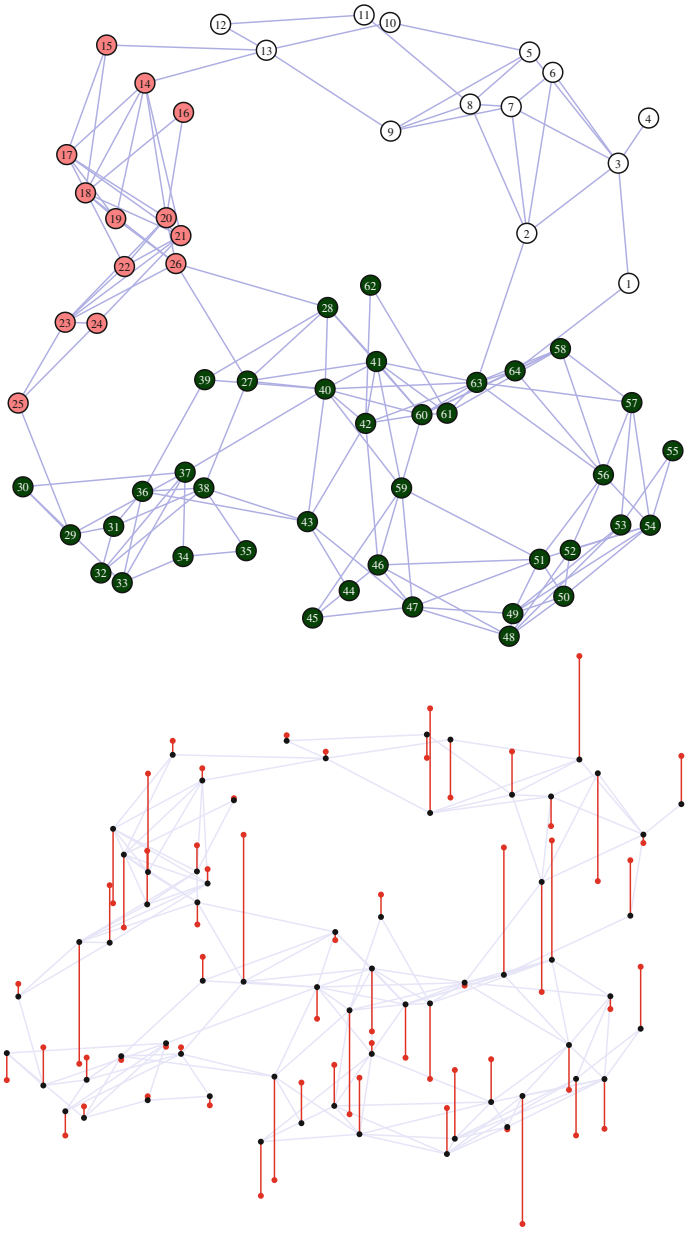


Fig. 1 Graph and signal on the graph. The signal is composed of three components. Component supports are presented with different vertex colors

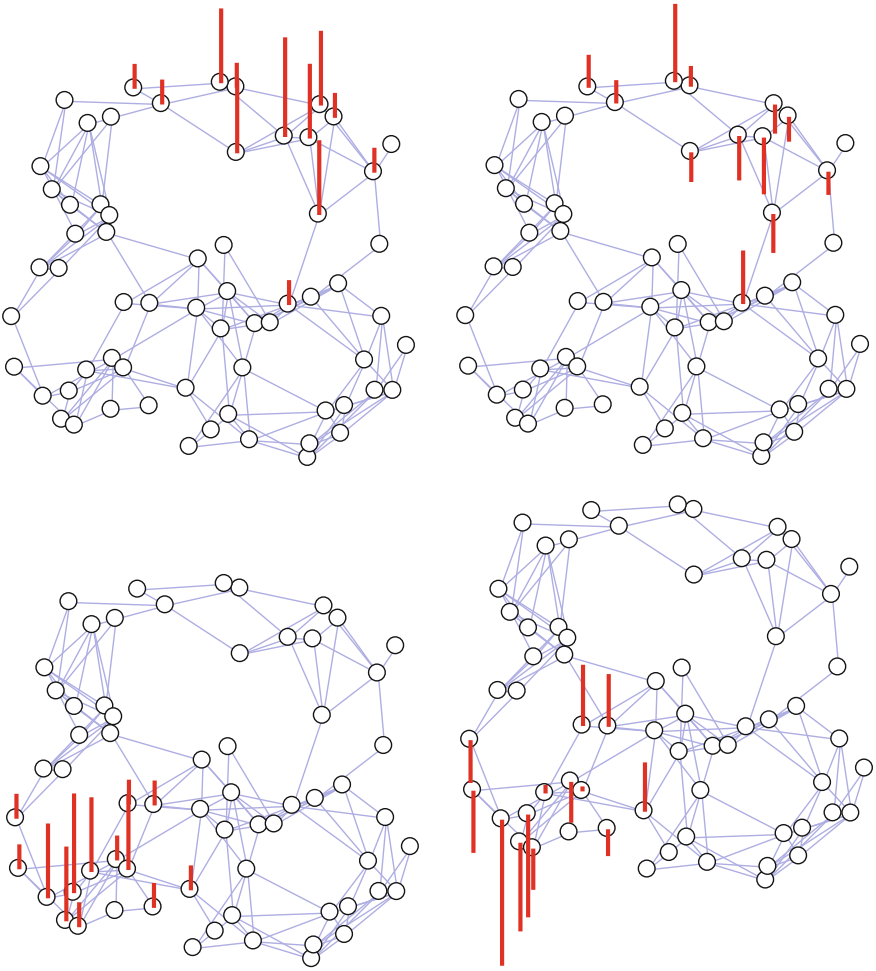


Fig. 2 Kernels with the vertex domain localized windows centered at vertices 8 (top) and 31 (bottom) and spectral indices 1 (left) and 6 (right)

signal analysis, it is common to calculate the STFT with M frequency indices for a window whose width is M. The same can be done in graph signal processing. The local vertex spectrum, at vertex m, can be written in a matrix form as

$$\begin{bmatrix} \mathbf{s}_m^{(B)} \\ \mathbf{s}_m^{(I)} \end{bmatrix} = \begin{bmatrix} \mathbf{B}^{(B)} \\ \mathbf{B}^{(I)} \end{bmatrix} \mathbf{x}_{h_m}^{(NZ)},$$

where $\mathbf{x}_{h_m}^{(NZ)}$ is vector with M nonzero elements of windowed signal $\mathbf{x}_{h_m}$, $\mathbf{B}^{(B)}$ is square $(M \times M)$ submatrix of the transformation matrix $\mathbf{U}^T$ with omitted columns

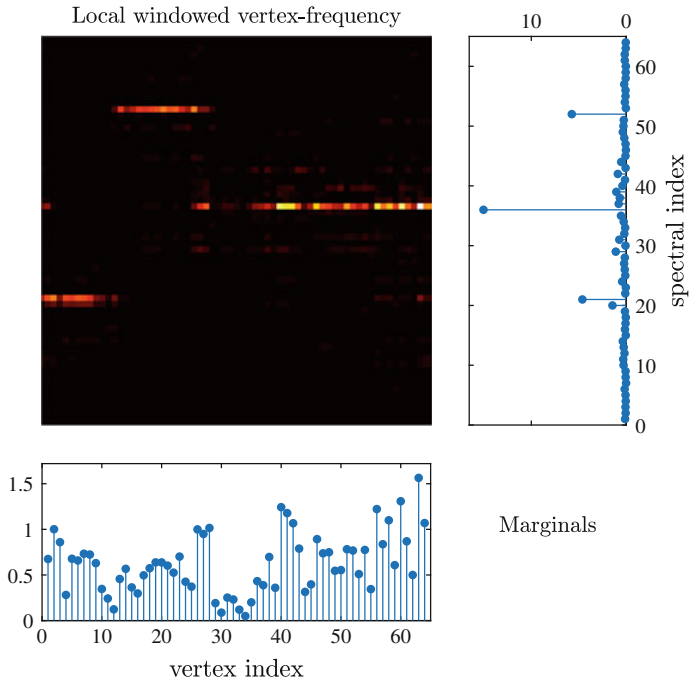


Fig. 3 Local vertex-frequency spectrum calculated using vertex neighborhood windows

that correspond to zero elements in $\mathbf{x}_{h_m}$, while the remaining $N - M$ rows are arranged into matrix $\mathbf{B}^{(I)}$. We have assumed that the local spectrum vector $\mathbf{s}_m$ is composed of two parts $\mathbf{s}_m^{(B)}$ and $\mathbf{s}_m^{(I)}$ in a such way that the matrix $\mathbf{B}^{(B)}$ is invertible. Then

$$\mathbf{s}_m^{(B)} = \mathbf{B}^{(B)}\mathbf{x}_{h_m}^{(NZ)}$$

and

$$\mathbf{s}_m^{(I)} = \mathbf{B}^{(I)}\mathbf{x}_{h_m}^{(NZ)} = \mathbf{B}^{(I)}(\mathbf{B}^{(B)})^{-1}\mathbf{s}_m^{(B)}$$

is the interpolation formula that enables calculation of $\mathbf{s}_m^{(I)}$ from $\mathbf{s}_m^{(B)}$. Note that if the condition number of matrix $\mathbf{B}^{(B)}$ is large, we can rearrange the spectral domain coefficients ordering until we get a satisfactory invertible matrix.

(3) *Directed graphs*: The vertex neighborhood may be defined as set of vertices that can be reached from the considered vertex by a walk whose length is at most D. Then we may use previous relations for the window calculation. This approach corresponds to one-sided windows in classical signal analysis.

If we want to define two-sided window, then we should also include all vertices from which we can reach the considered vertex by walk whose length is at most D. This means that for a directed graph we should assume that vertices with distance one form the considered vertex m are the vertices from which we can reach vertex m

with walk of length 1. In this case $\mathbf{A}_1 = \mathbf{A} + \mathbf{A}^T$ where addition is logical operation (Boolean OR). The matrix $\mathbf{A}_2$ is

$$\mathbf{A}_2 = (\mathbf{A} \odot \mathbf{A} + \mathbf{A}^T \odot \mathbf{A}^T) \circ (\mathbf{1} - \mathbf{I}) \circ (\mathbf{1} - \mathbf{A}_1).$$

This procedure could be continued for walks up to the desired maximal length D.

For a circular directed graph in this way, we will get the classical STFT with symmetric window.

(4) *Ordering*: In order to visualize local spectral content, we should order vertices in the corresponding graph. This ordering is not unique, and one possible way is to define the order according to the values of low order eigenvectors of the Laplacian. We can, for example, try to minimize the number of zero crossing in the low order eigenvectors by an appropriate vertex reordering. This can be achieved by reordering vertices such that the elements of $b(n)$ are nondecreasing, where $b(n)$ is defined as

$$b(n) = \sum_{k=2}^{K} (1 + \text{sign}(u_k(n)))2^{-k} \tag{10}$$

where K is the number of considered eigenvectors. Here we consider the sign of the eigenvector coefficients and group coefficients with the same sign. Then the sign of the next eigenvector is used to order coefficients in each group. This ordering is based on the vertex spectral similarity. Note that for $k = 1$ we have $\lambda_1 = 0$ and the corresponding eigenvector is constant.

2.4 Vertex Localization Windows Defined in the Spectral Domain

Consider two signals $x(n)$ and $y(n)$ on a graph. The GDFT of these signals are given by $X(k)$ and $Y(k)$. The generalized convolution $z(n)$ of signals $x(n)$ and $y(n)$ can be defined in the GDFT domain as

$$Z(k) = X(k)Y(k)$$
$$z(n) = x(n) * y(n).$$

A "shift" on a graph cannot be extended in a direct way from the classical signal processing theory. The generalized convolution is used to define a "shift" of a window on a graph [13]. A "shift" on a graph from vertex m to vertex n would be achieved by using the delta function located at the vertex m, defined as

$$\delta_m(n) = \delta(n - m). \tag{11}$$

The GDFT of this function is given by

$$\Delta(k) = \sum_{n=1}^{N} \delta_m(n) u_k(n) = u_k(m). \tag{12}$$

The window localized at the vertex m is equal to [13]

$$h_m(n) = h(n) * \delta_m(n) = \sum_{k=1}^{N} H(k) u_k(m) u_k(n), \tag{13}$$

where the window basic function $h(n)$ is defined in the spectral domain, for example, as

$$H(k) = C \exp(-\lambda_k \tau), \tag{14}$$

where C is the window amplitude and $\tau > 0$ is a constant that determines the window width. Two windows obtained using this method are presented in Fig. 4 (left).

The window localized at the vertex m satisfies the following properties:

- Symmetry $h_m(n) = h_n(m)$ follows from definition (13).
- Sum of $h_m(n)$ is equal to $H(1)$,

$$\sum_{n=1}^{N} h_m(n) = \sum_{k=1}^{N} H(k) u_k(m) \sum_{n=1}^{N} u_k(n) = \sum_{k=1}^{N} H(k) u_k(m) \delta(k-1) \sqrt{N} = H(1).$$

- Parseval's theorem for $h_m(n)$ is

$$\sum_{n=1}^{N} |h_m(n)|^2 = \sum_{k=1}^{N} |H(k) u_k(m)|^2. \tag{15}$$

The local vertex spectrum can be written as

$$S(m,k) = \sum_{n=1}^{N} x(n) h_m(n)\ u_k(n) = \sum_{n=1}^{N} \sum_{p=1}^{N} x(n) H(p) u_p(m) u_p(n)\ u_k(n). \tag{16}$$

The modulated version of the window (kernel) centered at vertex m and spectral index k is

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

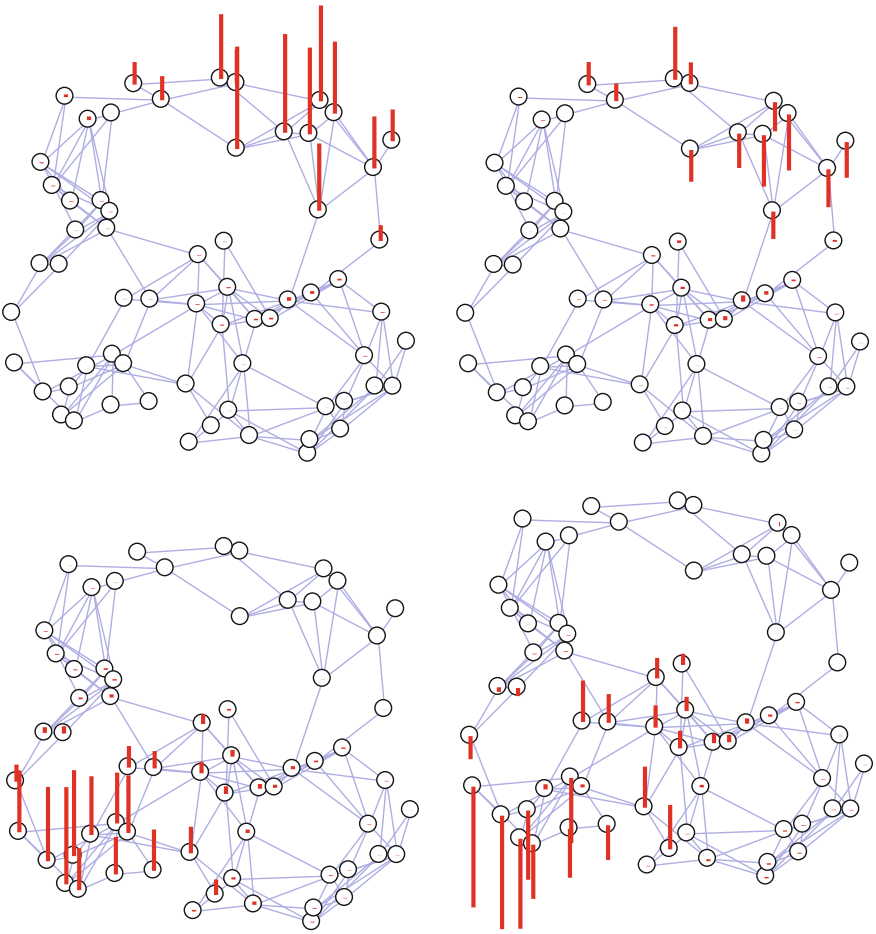


Fig. 4 Kernels with the spectral domain defined windows centered at vertices 8 (top) and 31 (bottom) and spectral indices 1 (left) and 6 (right)

It is shown in Fig. 4 for $k = 1$ (left) and $k = 6$ (right) for two vertex locations $m = 8$ (top) and $m = 31$ (bottom). The kernels with the vertex-domain localized windows, at the same vertex index and spectral index locations, are shown in Fig. 2. The local vertex-frequency representation with a spectral domain window is presented in Fig. 5.

Using the kernel notation, the local vertex spectrum, for a given vertex m, is equal to the projection of a signal $x(n)$ onto the kernel $\mathcal{H}_{m,k}(n)$,

$$S(m,k) = \sum_{n=1}^{N} \mathcal{H}_{m,k}(n)x(n).$$

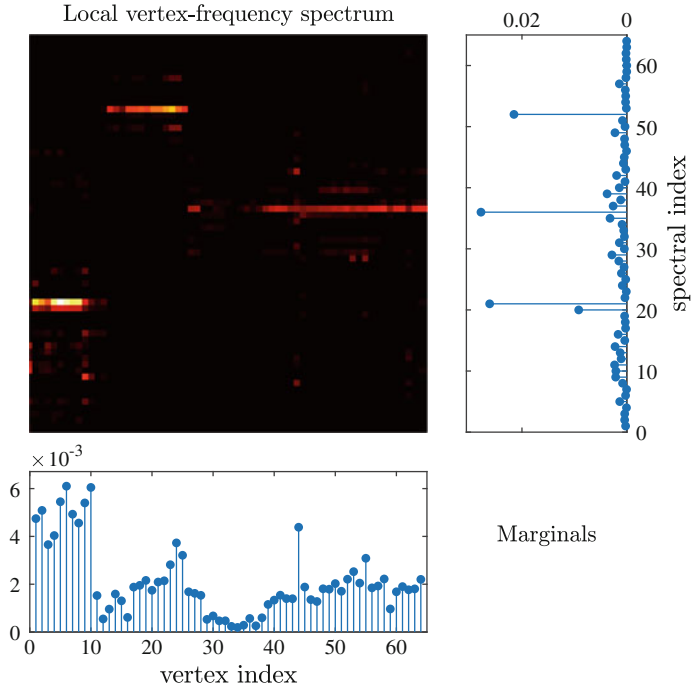


Fig. 5 Local vertex-frequency spectrum calculated using vertex-frequency localized kernels/windows defined in the spectral domain

2.5 *Inversion*

The inversion relation for both previous window forms can be considered in a unified way. The reconstruction of a signal $x(n)$ from its local spectrum $S(m, k)$ is performed by an inverse GDFT

$$x(n)h_m(n) = \sum_{k=1}^{N} S(m,k)\, u_k(n) \tag{17}$$

followed by a summation over all vertices m

$$x(n) = \frac{1}{\sum_{m=1}^{N} h_m(n)} \sum_{m=1}^{N} \sum_{k=1}^{N} S(m,k) u_k(n). \tag{18}$$

If the windows $h_m(n)$ satisfy the condition

$$\sum_{m=1}^{N} h_m(n) = 1,$$

the reconstruction is vertex independent. In that case

$$x(n) = \sum_{m=1}^{N} \sum_{k=1}^{N} S(m,k) u_k(n) = \sum_{k=1}^{N} X(k) u_k(n), \tag{19}$$

where

$$X(k) = \sum_{m=1}^{N} S(m,k)$$

is the spectral marginal of the local vertex spectrum.

The condition $\sum_{m=1}^{N} h_m(n) = 1$ can be achieved if the elements of matrix $\mathbf{A}_d$, $d = 1, 2, \ldots, D-1$ are normalized, prior to $\mathbf{P}_D$ matrix calculation in such a way that the sum over all columns is equal to 1. Then

$$\sum_{m=1}^{N} h_m(n) = \sum_{m=1}^{N} P_D(n,m) = \sum_{d=1}^{D-1} g(d) = const.$$

For the windows obtained using generalized graph shift, this conditions is always satisfied since $H(1) = const$.

In general, the local vertex spectrum $S(m,k)$ can be calculated over a reduced set of vertices $m \in \mathbb{M} \subset \mathcal{V}$. In this case, summation over m in the reconstruction formula should be performed over vertices $m \in \mathbb{M}$ only, and vertex independent reconstruction is achieved if $\sum_{m \in \mathbb{M}} h_m(n) = 1$.

Inversion of the local vertex spectrum, within the Gabor expansion framework, is obtained from

$$\sum_{m=1}^{N} \sum_{k=1}^{N} S(m,k) \mathcal{H}_{m,k}(n) = \sum_{m=1}^{N} \left(\sum_{k=1}^{N} S(m,k) h_m(n) u_k(n) \right) =$$

$$\sum_{m=1}^{N} \left(\sum_{i=1}^{N} \underset{k \to i}{\text{IGDFT}}\{S(m,k)\} \underset{k \to i}{\text{IGDFT}}\{h_m(n) u_k(n)\} \right) =$$

$$\sum_{m=1}^{N} \sum_{i=1}^{N} x(i) h_m(i) h_m(n) \delta(n-i) = \sum_{m=1}^{N} x(n) h_m^2(n) = x(n) \sum_{m=1}^{N} h_m^2(n),$$

where IGDFT denotes the inverse GDFT transform and Parseval's relation is used.

The inversion formula is then

$$x(n) = \frac{1}{\sum_{m=1}^{N} h_m^2(n)} \sum_{m=1}^{N} \sum_{k=1}^{N} S(m,k) h_m(n) u_k(n). \tag{20}$$

This kind of inversion is vertex invariant if

$$\sum_{m=1}^{N} h_m^2(n) = 1. \tag{21}$$

If the local vertex spectrum $S(m,k)$ is calculated over a reduced set of vertices $m \in \mathbb{M} \subset \mathcal{V}$, then vertex independent reconstruction condition is $\sum_{m\in\mathbb{M}} h_m^2(n) = 1$.

Filtering in the vertex-frequency domain can be implemented by using the vertex-frequency support function $B(m,k)$. The filtered local vertex spectrum is

$$S_f(m,k) = S(m,k)B(m,k)$$

and the filtered signal $x_f(n)$ is obtained by inversion of $S_f(m,k)$ using the presented inversion methods.

The support function $B(m,k)$ can be obtained, for example, by thresholding noisy values of the local vertex spectrum $S(m,k)$.

2.6 Uncertainty Principle

In classical signal analysis, the window is used to improve the signal localization in the joint time-frequency domain. However, the uncertainty principle prevents an ideal localization in both time and frequency. For the DFT, the uncertainty principle states that

$$\|\mathbf{x}\|_0 \|\mathbf{X}\|_0 \geq N.$$

It means that the product of the number of signal nonzero values $\|\mathbf{x}\|_0$ and the number of its DFT nonzero values $\|\mathbf{X}\|_0$ is greater or equal than the total number of signal samples N.

If the signal is windowed with a function $\mathbf{h}_m$ whose width is N_{h_m} then $\|\mathbf{x}\mathbf{h}_m\|_0 \leq N_{h_m}$. The uncertainty principle for the classical STFT, defined as $\mathbf{STFT}_m = \text{DFT}\{\mathbf{x}\mathbf{h}_m\}$, is

$$\|\mathbf{STFT}_m\|_0 \geq \frac{N}{N_{h_m}}.$$

This means that the window whose width is, for example, $N_{h_m} = N/4$ can not produce the STFT with less than 4 nonzero samples for considered instant m.

For graph signals, the general form of the uncertainty principle should be used. Consider a graph signal $\mathbf{x}$ and its transform $\mathbf{X}$ in the domain of orthonormal basis functions $u_k(n)$. The uncertainty principle states that [20–23]

$$\|\mathbf{x}\|_0 \|\mathbf{X}\|_0 \geq \frac{1}{\max_{k,m}\{|u_k(m)|^2\}}.$$

For the orthonormal DFT, when $u_k(n) = \frac{1}{\sqrt{N}} \exp(j2\pi nk/N)$, the classical DFT uncertainty principle form follows.

In graph signal processing, the basis functions can assume quite different forms than in the DFT case. In cases, when, for example, there is a vertex loosely connected with other vertices $\max\{|u_k(m)|^2\} \to 1$ and even $\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq 1$ is possible. This means that the graph signal can be well localized in both the vertex and the frequency domain.

For the graph presented in Fig. 1, $\max\{|u_k(m)|^2\} = 0.7513$ meaning that $\|\mathbf{x}\|_0\|\mathbf{X}\|_0 \geq 1.331$ is possible.

2.7 Local Vertex Spectrogram and Energy Condition

The local vertex spectrogram is defined as

$$|S(m,k)|^2 = \left| \sum_{n=1}^{N} x(n) h_m(n)\ u_k(n) \right|^2. \tag{22}$$

The vertex marginal property is (according to Parseval's theorem)

$$\sum_{k=1}^{N} |S(m,k)|^2 = \sum_{k=1}^{N} S(m,k) \sum_{n=1}^{N} x(n) h_m(n)\ u_k(n) = \sum_{n=1}^{N} |x(n) h_m(n)|^2. \tag{23}$$

This is a vertex smoothed signal power.

For the energy, we get

$$\sum_{m=1}^{N} \sum_{k=1}^{N} |S(m,k)|^2 = \sum_{n=1}^{N} \left(|x(n)|^2 \sum_{m=1}^{N} |h_m(n)|^2 \right). \tag{24}$$

If $\sum_{m=1}^{N} |h_m(n)|^2 = 1$ for all n then the spectrogram on the graph is energy unbiased

$$\sum_{m=1}^{N} \sum_{k=1}^{N} |S(m,k)|^2 = \sum_{n=1}^{N} |x(n)|^2 = E_x. \tag{25}$$

From the previous relations we can easily prove that the local vertex spectrum is a frame [24–27] since

$$\sum_{m=1}^{N} |h_m(n)|^2 = \sum_{k=1}^{N} |H(k)|^2 |u_k(n)|^2. \tag{26}$$

We can write

$$\frac{1}{N} H^2(1) \leq \sum_{m=1}^{N} |h_m(n)|^2 \leq \max_{m,k} \{|u_k(n)|^2\} \sum_{k=1}^{N} |H(k)|^2 = \mu^2 E_h, \tag{27}$$

where $\mu = \max_{m,k}\{|u_k(n)|\}$. This means that

$$\frac{1}{N}H^2(1)E_x \leq \sum_{m=1}^{N}\sum_{k=1}^{N}|S(m,k)|^2 \leq E_x\mu^2 E_h. \tag{28}$$

The shift $\mathcal{H}_{m,k}(n)$ is an equiangular tight frame (ETF) if $\mu = |u_k(n)| = \frac{1}{\sqrt{N}}$. Then $\sum_{m=1}^{N}|h_m(n)|^2 = \sum_{k=1}^{N}|H(k)|^2|u_k(n)|^2 = E_h/N$, and equality holds.

2.8 Optimization

The concentration of the local vertex spectrum representation can be measured using the normalized norm-one [28]

$$\mathcal{M} = \frac{1}{F}\sum_{m=1}^{N}\sum_{k=1}^{N}|S(m,k)| = \frac{\|\mathbf{S}\|_1}{\|\mathbf{S}\|_F}, \tag{29}$$

where $F = \|\mathbf{S}\|_F = \sqrt{\sum_{m=1}^{N}\sum_{k=1}^{N}|S(m,k)|^2}$ is the Frobenius norm of matrix $\mathbf{S}$. Any other norm $\|\mathbf{S}\|_p^p$ with $0 \leq p \leq 1$ can be used instead of $\|\mathbf{S}\|_1$. Norms with p close to 0 are noise sensitive. The norm with $p = 1$ is the only convex norm, allowing the gradient based normalization [28].

The concentration measure $\mathcal{M}(\tau) = \|\mathbf{S}\|_1/\|\mathbf{S}\|_F$ for the signal from Fig. 1 and the window given by (14), is shown in Fig. 6, for various τ. The optimal vertex frequency representation is also given in Fig. 6. The optimal τ can be obtained in a few steps in an iterative way $\tau_k = \tau_{k-1} - \alpha(\mathcal{M}(\tau_{k-1}) - \mathcal{M}(\tau_{k-2}))$.

The optimization of parameter τ can be done by using more advanced techniques [21, 22] based on the graph uncertainty principle.

2.9 Spectral Domain Localization

In the classical STFT, it is possible to perform localization using a window in the spectral domain. The dual form of STFT is obtained using inverse DFT. Similarly, for graph signals we can perform localization in the spectral domain and obtain a local-vertex spectrum as an inverse GDFT

$$S(m,k) = \sum_{p=1}^{N} X(p)H(p-k)\,u_k(m), \tag{30}$$

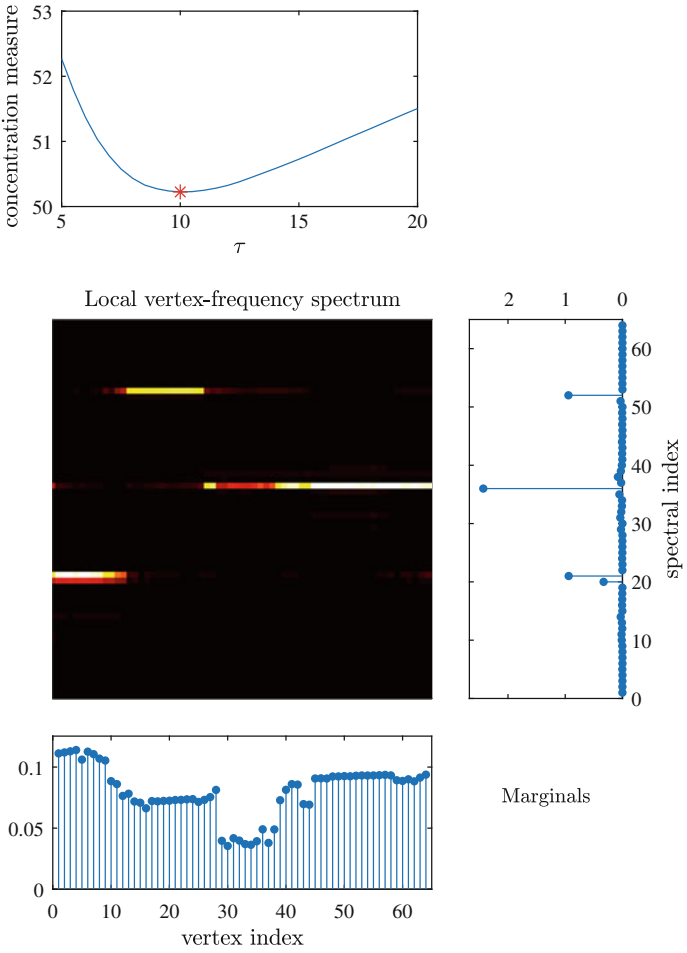


Fig. 6 Measure of the spectrogram concentration for various window parameter τ (top) and the corresponding optimal vertex-frequency representation with marginals (bottom)

where $H(p-k)$ is the frequency domain window and $X(p)$ is the GDFT of the considered signal.

3 Vertex-Frequency Energy Distributions

Vertex-frequency representations based on the distribution of energy over the vertex and spectral index space is presented in this section. It follows the concept of time-frequency energy distributions in time-frequency signal analysis. An ideal vertex-

frequency energy distribution will be discussed first. Special attention will be paid to local smoothness and marginal properties. Two forms of a distribution corresponding to the Rihaczek distribution will be introduced. Finally, a class of reduced interference vertex-frequency distributions, satisfying the marginal properties, will be presented.

3.1 Global Graph Signal Smoothness

The smoothness of a graph signal $\mathbf{x}$ is defined by using its quadratic form

$$E_x = \mathbf{x^T L x}.$$

The smoothness λ_x is equal to the quadratic form normalized by the signal energy

$$\lambda_x = \frac{\mathbf{x}^T \mathbf{L x}}{\mathbf{x}^T \mathbf{x}}.$$

In the case of classical time domain signals, the Laplacian on a circle graph represents the second order finite difference $y(n) = -x(n-1) + 2x(n) - x(n+1)$. This difference can be written in a matrix form as $\mathbf{y} = \mathbf{L x}$. It is obvious that the signal $x(n)$ with small changes should have small quadratic form $E_x = \sum_n ((x(n) - x(n-1))^2 + (x(n) - x(n+1))^2)/2$. This reasoning can be used for the graph signals as well. For these signals $E_x = \sum_{m=1}^{N} \sum_{n=1}^{N} w_{mn}((x(n) - x(m))^2/2$ (see Chap. I).

The eigenvector and eigenvalue relation is $\mathbf{L}\mathbf{u}_k = \lambda_k \mathbf{u}_k$ or

$$\mathbf{u}_k^T \mathbf{L}\mathbf{u}_k = \lambda_k \mathbf{u}_k^T \mathbf{u}_k = \lambda_k = E_{u_k}, \tag{31}$$

since $\mathbf{u}_k^T \mathbf{u}_k = 1$. The quadratic form of an eigenvector is equal to the corresponding eigenvalue. It can be used as a measure of the signal smoothness. The eigenvectors corresponding to small λ_k belong to the low-pass (slow-varying) part of a graph signal.

Examples of signals with a small, a moderate and a large smoothness λ_x are presented in Fig. 7. In order to make an analogy with the classical time-domain signal processing we presented the time-domain signals in Fig. 7 (left). They can be considered as the signals on a circular graph. The smooth signals, with small smoothness factors, have similar signal values on the neighboring vertices (time instants), while the similarity of neighboring signal values does not hold for fast-varying signals with high smoothness indices.

For a signal of the form

$$x(n) = \sum_{i=1}^{M} x_i(n) = \sum_{i=1}^{M} \alpha_i u_{k_i}(n)$$

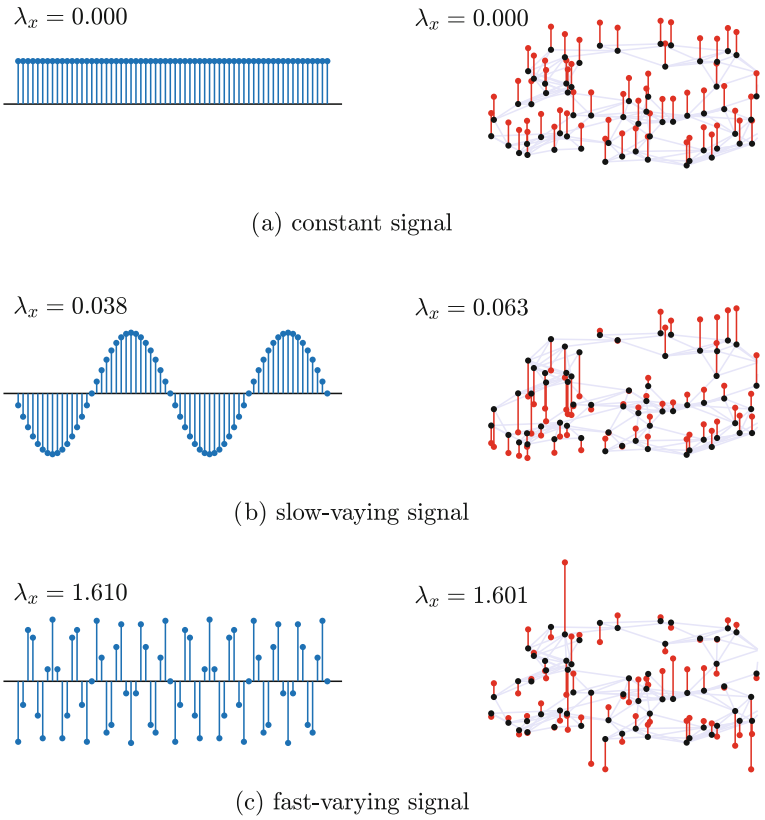


Fig. 7 An example of a constant, a slow-varying, and a fast-varying signal in the time domain (left) and in the graph domain (right). The global signal smoothness λ_x is given for each case

the global smoothness is

$$\lambda_x = \frac{\sum_{i=1}^{N} \alpha_i^2 \lambda_{k_i}}{\sum_{i=1}^{N} \alpha_i^2}.$$

It is obvious that $\lambda_{\min} \leq \lambda_x \leq \lambda_{\max}$, where $\lambda_{\min} = \min\{\lambda_{k_1}, \lambda_{k_2}, \ldots \lambda_{k_M}\}$ and $\lambda_{\max} = \max\{\lambda_{k_1}, \lambda_{k_2}, \ldots \lambda_{k_M}\}$.

The graph signal smoothness plays an important role in the graph topology learning. The smoothness is also used in the vertex ordering and the graph clustering.

3.2 Local Graph Signal Smoothness

Consider a single component graph signal

$$x(n) = \alpha u_k(n),$$

where $u_k(n)$ is the kth Laplacian eigenvector. The smoothness of this signal is equal to the corresponding eigenvalue

$$\frac{\mathbf{x}^T\mathbf{L}\mathbf{x}}{\mathbf{x}^T\mathbf{x}} = \frac{\alpha^2\mathbf{u}_k^T\mathbf{L}\mathbf{u}_k}{\alpha^2\mathbf{u}_k^T\mathbf{u}_k} = \lambda_k.$$

The smoothness can be related to the frequency in classical signal analysis. For a circular undirected graph the eigenvectors are periodic functions $u_k(n) = \cos(2\pi nk/N + \phi)$, with frequency $\omega = 2\pi k/N$, and the smoothness follows as the eigenvalue from the equation $\mathbf{L}\mathbf{u}_k = \lambda_k\mathbf{u}_k$ as

$$\lambda_k = 4\sin^2(\omega/2) = \mathbf{u}_k^T\mathbf{L}\mathbf{u}_k.$$

This means that the eigenvalue corresponds to the squared frequency. The relation is via $2\sin^2(\omega/2)$ function due to the discretization (discrete-time to continuous-time frequency mapping using the first-order finite difference). In the continuous-time case (for a small ω in the discrete-time domain) we would have $\lambda_k \approx \omega^2$.

In classical signal analysis, for signals whose frequency changes, the instantaneous frequency is defined and used instead of the frequency. Various definitions of the instantaneous frequency are introduced [9–11]. The most straightforward one would be to define the instantaneous frequency as the frequency of a sinusoidal signal which best fits the signal behavior at and in the very close vicinity of the considered instant t. In that case, the method for frequency estimation of the signal in three close points could be used, as described in [29]. The signal $x(t+\tau)$ is approximated with a second order polynomial around $x(t)$,

$$x(t+\tau) \approx x(t) + x'(t)\tau + x''(t)\tau^2/2.$$

By comparing this signal with a general sinusoidal signal at an instant t and small τ

$$\begin{aligned} A\cos(\omega_t(t+\tau)+\phi) \approx{}& A\cos(\omega_t t+\phi) - A\omega_t\sin(\omega_t t+\phi)\tau \\ & - A\omega_t^2\cos(\omega_t t+\phi)\tau^2/2 \end{aligned}$$

we can conclude that for $x(t) \neq 0$ the sinusoidal signal with frequency

$$\omega_t^2 = \omega^2(t) = \lambda(t) = -\frac{x''(t)}{x(t)}$$

fits the signal around this instant. In this way we can define the squared instantaneous frequency $\omega^2(t)$ of a signal at an instant t, assuming that the conditions for the second order polynomial approximation holds. For $x(t) = 0$, the instantaneous frequency

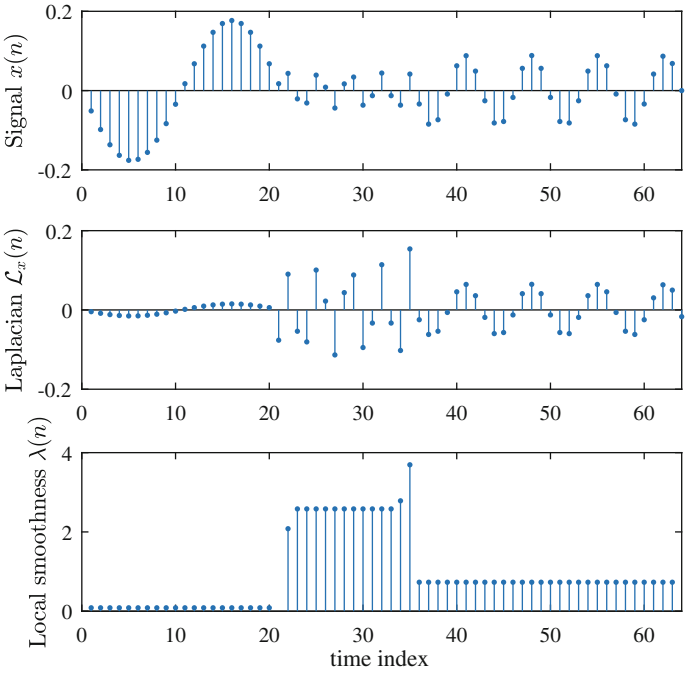


Fig. 8 An example of the signal local smoothness in the time domain (vertex domain on a circular graph)

can be calculated using the higher-order derivatives ratio $x^{(n+2)}(t)/x^{(n)}(t)$ assuming that $x^{(n)}(t) \neq 0$.

The discrete-time form of the instantaneous frequency relation is

$$\lambda(n) = \omega^2(n) = -\frac{x(n-1) - 2x(n) + x(n+1)}{x(n)} = \frac{\mathcal{L}_x(n)}{x(n)}, \tag{32}$$

where $\mathcal{L}_x(n)$ are the elements of vector $\mathbf{Lx}$ (for a circular graph). An example of a time-domain signal that contains a slow-varying component at the beginning, a fast-varying in the middle, and a moderate-varying component at the end is presented in Fig. 8.

The last relation will be used to introduce the local smoothness for a general graph as

$$\lambda(n) = \frac{\mathcal{L}_x(n)}{x(n)}, \tag{33}$$

with the assumption $x(n) \neq 0$.

Some of the local smoothness properties are listed next [30].

1. For a monocomponent signal

$$x(n) = \alpha u_k(n)$$

the local smoothness is constant and it is equal to its global smoothness

$$\lambda(n) = \frac{\alpha \mathcal{L}_{u_k}(n)}{\alpha u_k(n)} = \lambda_k,$$

since $\mathcal{L}_{u_k}(n) = \lambda_k u_k(n)$ holds for each element of the matrix equation $\mathbf{L}\mathbf{u}_k = \lambda_k \mathbf{u}_k$. In classical signal analysis this means that the instantaneous frequency of a sinusoidal signal (as a basis function) is equal to the component frequency.

2. A piecewise monocomponent signal could be defined as

$$x(n) = \alpha_i u_{k_i}(n) \text{ for } n \in \mathcal{V}_i, \quad i = 1, 2, \ldots, M$$

where $\mathcal{V}_i$ are disjunct subsets of vertices and $u_{k_i}(n)$ are eigenvectors.
The local smoothness for this signal should be

$$\lambda(n) = \frac{\alpha_i \mathcal{L}_{u_{k_i}}(n)}{\alpha_i u_{k_i}(n)} = \lambda_{k_i} \text{ for } n \in \mathcal{V}_i, \quad i = 1, 2, \ldots, M \tag{34}$$

This is true for all interior vertices, where the vertex n and its neighborhood (used for the Laplacian calculation) belong to the considered subset $\mathcal{V}_i$.
An example of a piecewise monocomponent signal is presented in Fig. 1. Three subsets of vertices $\mathcal{V}_1$, $\mathcal{V}_2$, and $\mathcal{V}_3$ are marked with colors. The component spectral indices are $k_1 = 20$, $k_2 = 52$, and $k_3 = 36$.
For subset $\mathcal{V}_1$, which includes vertices from 1 to 13, the boundary vertices are 1, 2, and 13. The remaining vertices are the interior vertices, where relation (34) holds. For subset $\mathcal{V}_2$ (vertices from 14 to 26), the boundary vertices are 15, 16, 25, and 26. For subset $\mathcal{V}_3$ (vertices from 27 to 64), the boundary vertices are 27, 28, 29, 63 and 64.
The local smoothness of the considered signal is calculated and presented in Fig. 9. Results obtained for the interior vertices are exact. They are presented with dots. The local smoothness at the boundary vertices is not exact, as expected, since it includes vertices with different signal components. They are presented with the cross marks.

3. An ideal vertex-frequency distribution $I(n,k)$ should behave as

$$I(n,k) \sim |x(n)|^2 \delta(\lambda_k - \lambda(n)),$$

assuming that the local smoothness is equal to an eigenvalue, or that it is rounded to the nearest eigenvalue.
The ideal vertex frequency distribution for the graph and the signal presented in Fig. 1 is shown in Fig. 10.
We can conclude, that a distribution, behaving as an ideal vertex-frequency distribution, can be used as an estimator of the local smoothness as

$$\hat{\lambda}(n) = \arg\max_k \{I(n,k)\}.$$

This estimator is common and widely used in classic time-frequency analysis [9–11].

4. For an M component graph signal $x(n) = \sum_{i=1}^{M} x_i(n) = \sum_{i=1}^{M} \alpha_i u_{k_i}(n)$, the local smoothness is

$$\lambda(n) = \frac{\sum_{i=1}^{M} \alpha_i \mathcal{L}_{u_{k_i}}(n)}{\sum_{i=1}^{M} \alpha_i u_{k_i}(n)} = \frac{\sum_{i=1}^{M} \alpha_i \lambda_{k_i} u_{k_i}(n)}{\sum_{i=1}^{M} \alpha_i u_{k_i}(n)}.$$

The ideal vertex-frequency representation should not be based on the local smoothness $\lambda(n)$ of the complete multicomponent signal, but on the smoothness of each individual signal component $x_i(n)$ denoted by $\lambda_i(n)$. Its form is

$$I(n,k) \sim \sum_{i=1}^{M} |x_i(n)|^2 \delta(\lambda_k - \lambda_i(n)).$$

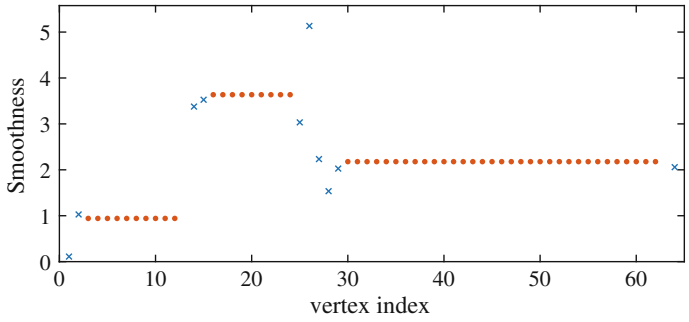


Fig. 9 Local smoothness for the signal given in Fig. 1. The results for the interior vertices are presented with points and the results for the boundary vertices are presented by crosses

The concept of ideal vertex-frequency distribution can be extended to piecewise multicomponent signals.

5. The local smoothness property holds for a general vertex-frequency distribution $G(n,k)$ if

$$\frac{\sum_{k=1}^{N}\lambda_k G(n,k)}{\sum_{k=1}^{N} G(n,k)} = \lambda(n). \tag{35}$$

Example: The ideal time-frequency distribution $I(n,k) = |x(n)|^2\delta(\lambda_k - \lambda(n))$ satisfies the local smoothness property if $\lambda(n) \in \{\lambda_1, \lambda_2, \ldots, \lambda_N\}$ for all n.

6. The local smoothness bandwidth for a vertex-frequency distribution $G(n,k)$ that satisfies Property 5 is defined by

$$\begin{aligned}\sigma_\lambda^2(n) &= \frac{\sum_{k=1}^{N}(\lambda_k - \lambda(n))^2 G(n,k)}{\sum_{k=1}^{N} G(n,k)} \\ &= \frac{\sum_{k=1}^{N}\lambda_k^2 G(n,k)}{\sum_{k=1}^{N} G(n,k)} - \lambda^2(n).\end{aligned} \tag{36}$$

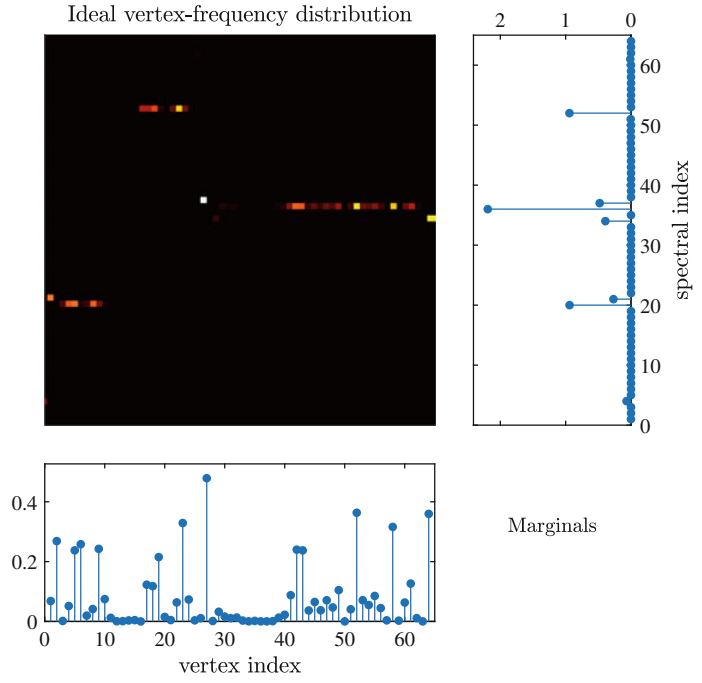


Fig. 10 Ideal vertex-frequency distribution

3.3 Vertex-Frequency Power Distribution

We will present two forms of vertex-frequency distributions. For the first form, we will follow the electric circuit reasoning, while for the second form, the signal processing definition of energy will be used.

3.4 Vertex-Frequency Power in Electric Circuit

Consider a resistive electric circuit and the corresponding graph, where the edge weights w_{nm} are equal to the corresponding conductances $1/R_{nm}$. The potential at the vertex n is denoted by $x(n)$. The power in all edges connected to the vertex n is equal to the sum of all $(x(n)-x(j))^2/R_{nj}$ or $w_{nj}(x(n)-x(j))^2$. Its value is

$$p(n)=\sum_{j=1}^{N} w_{nj}(x(n)-x(j))^2.$$

The power within the whole network is

$$P=\sum_{n=1}^{N}\sum_{j=1}^{N}\frac{1}{2}w_{nj}(x(n)-x(j))^2.$$

The factor $\frac{1}{2}$ is the result of the fact that all edges are taken twice in the summation over all vertices in the circuit. This relation can also be obtained by introducing the external current generators $i_G(n)$ at each vertex. These generators are needed to obtain the actual potentials $x(n)$. The vector of all external currents is denoted by $\mathbf{i}_G$. According to Kirchoff's first law $\mathbf{i}_G=\mathbf{L}\mathbf{x}$. The total power in a circuit is then calculated as

$$P=\mathbf{x}\mathbf{i}_G=\mathbf{x}(\mathbf{L}\mathbf{x})=\sum_{n=1}^{N}x(n)\sum_{j=1}^{N}w_{nj}(x(n)-x(j)).$$

It can be written as

$$P = \sum_{n=1}^{N}\sum_{k=1}^{N}\sum_{j=1}^{N}\frac{1}{2}w_{nj}(x(n)-x(j))X(k)(u_k(n)-u_k(j)).$$

The total power is obtained as a sum of the terms

$$P(n,k) = \sum_{j=1}^{N}\frac{1}{2}w_{nj}(x(n)-x(j))X(k)(u_k(n)-u_k(j))$$

over the vertex and frequency indices as

$$P = \sum_{n=1}^{N}\sum_{k=1}^{N}P(n,k).$$

Therefore, the value of $P(n,k)$ can be considered as a vertex-frequency power distribution of signal $x(n)$ over this graph, The marginal properties of this distribution are:

$$\begin{aligned}&\sum_{n=1}^{N}P(n,k) = \lambda_k|X(k)|^2 = X_D^2(k) \text{ and}\\ &\sum_{k=1}^{N}P(n,k) = \sum_{j=1}^{N}\frac{1}{2}w_{nj}(x(n)-x(j))^2 = x_D^2(n),\end{aligned} \tag{37}$$

where $x_D^2(n) = \mathcal{L}_x(n)x(n)$.

We have obtained that the spectral power is of the form $\lambda_k|X(k)|^2$. For $k=1$, this power is zero-valued, since $\lambda_1 = 0$ and the corresponding eigenvector $u_1(n)$ is constant. A constant potential does not produce any power in the network, since the voltage between each pair of vertices is 0. This kind of power, proportional to the frequency (squared), is present in the Teager energy operator.

We can define inverse Laplacian of a signal using the transform $X^2(k)/\lambda_k$. Since the Laplacian of a signal, with the transform $\lambda_k X(k)$, is a kind of second order derivation on a graph, its inverse can be considered as a kind of (double) integration on a graph.

Example: Consider the graph signal from Fig. 1. Its vertex-frequency power distribution is shown in Fig. 11. The marginal values of this distribution (37) are exact.

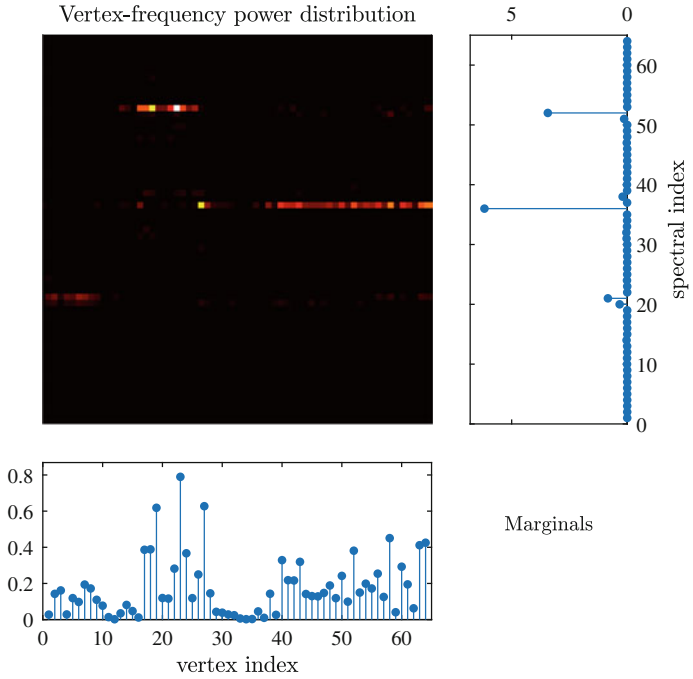


Fig. 11 Vertex-frequency power distribution

3.5 Signal Energy Vertex-Frequency Distributions

Energy in signal processing is commonly defined as

$$E = \sum_{n=1}^{N} x^2(n) = \sum_{n=1}^{N} x(n) \sum_{k=1}^{N} X(k) u_k(n).$$

It can be written as

$$E = \sum_{n=1}^{N} \sum_{k=1}^{N} x(n) X(k) u_k(n) = \sum_{n=1}^{N} \sum_{k=1}^{N} E(n,k),$$

where the vertex-frequency energy distribution is defined by [31]

$$E(n,k) = x(n) X(k) u_k(n) = \sum_{m=1}^{N} x(n) x(m) u_k(m) u_k(n). \tag{38}$$

This corresponds to the Rihaczek distribution in time-frequency analysis.

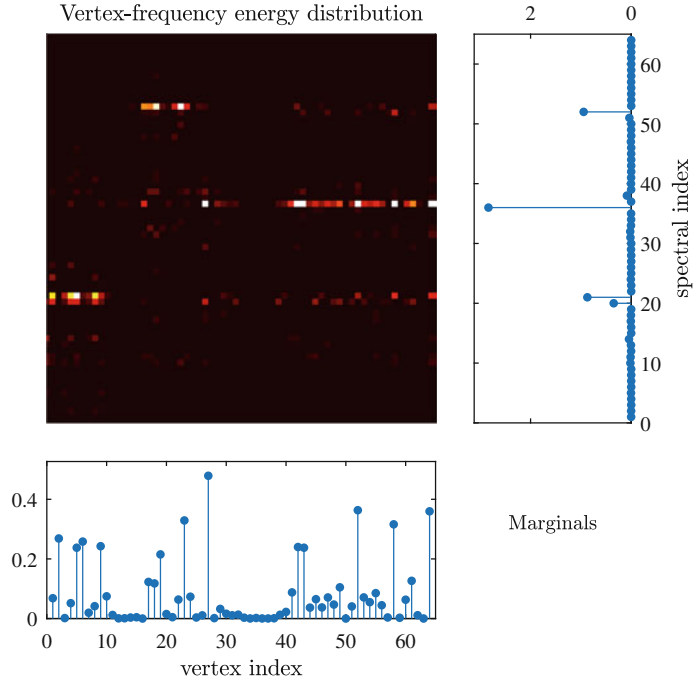


Fig. 12 Vertex-frequency energy distribution

The marginal properties of this distribution are

$$\sum_{n=1}^{N} E(n,k) = |X(k)|^2 \quad \text{and} \quad \sum_{k=1}^{N} E(n,k) = x^2(n).$$

They correspond to the signal power and squared spectra of the graph signal $x(n)$.

The vertex-frequency distribution defined by $E(n,k) = x(n)X(k)u_k(n)$ satisfies the local smoothness property (35). For this distribution

$$\frac{\sum_{k=1}^{N} \lambda_k E(n,k)}{\sum_{k=1}^{N} E(n,k)} = \frac{\sum_{k=1}^{N} \lambda_k x(n)X(k)u_k(n)}{\sum_{k=1}^{N} x(n)X(k)u_k(n)} = \frac{x(n)\mathcal{L}_x(n)}{x^2(n)} = \frac{\mathcal{L}_x(n)}{x(n)} = \lambda(n),$$

since $\sum_{k=1}^{N} \lambda_k X(k)u_k(n)$ are the elements of the inverse GDFT of $\mathbf{\Lambda X}$. This inverse transform is equal to

$$\mathbf{U\Lambda X} = \mathbf{U\Lambda}(\mathbf{U}^T\mathbf{U})\mathbf{X} = (\mathbf{U\Lambda U}^T)(\mathbf{UX}) = \mathbf{Lx}$$

with elements $\mathcal{L}_x(n)$. Therefore, $\sum_{k=1}^{N} \lambda_k X(k)u_k(n) = \mathcal{L}_x(n)$.

For this vertex-frequency energy distribution, the local smoothness bandwidth (36) may easily be written in terms of the elements of $\mathbf{L}^2\mathbf{x}$, $\mathbf{Lx}$, and $\mathbf{x}$, since $\sum_{k=1}^{N} \lambda_k^2 X(k) u_k(n) = \mathcal{L}_{\mathcal{L}_x}(n)$.

Example: The distribution $E(n, k)$ of the graph signal from Fig. 1, along with the marginal properties, is shown in Fig. 12. The marginal properties are satisfied up to the computer's precision. The localization of energy is better than in the cases obtained with the localized windows in Figs. 5 and 11. This distribution does not use a localization window. Although the first component of the signal analyzed in Fig. 12 exists in vertices $n \in \mathcal{V}_1 = \{1, 2, \ldots, 13\}$ only, the distribution $E(n, k)$ is nonzero for higher vertex indices n at $k_1 = 20$. This is the cross-term effect, well-known in classical time-frequency analysis. The same effect can be seen at higher vertex indices n at $k_1 = 52$ as well.

4 Reduced Interference Vertex-Frequency Energy Distributions

The general class of time-frequency energy distributions is extended to graph signals in this section with the aim to reduce the cross-term interferences, while preserving the marginal properties [32]. After a review of the classical Cohen class of distribution, conditions for the vertex-frequency marginal properties are derived. Few examples of the vertex-frequency energy distributions are given.

4.1 Review of the Classical Cohen Class of Distributions

Although it is known that any distribution can be used as the basis for the Cohen class of distribution, the Wigner distribution is commonly used [9–11]. Having in mind that the Wigner distribution is not suitable for the graph framework extension, here we will use the Rihaczek distribution as the basis. Since this kind of the Cohen class of distributions is not presented in common literature on time-frequency analysis, a short review of the Cohen class of distributions is presented. The Rihaczek distribution is [9–11]

$$R(t, \omega) = x(t) X^*(\omega) \exp(-j\omega t).$$

Its ambiguity domain form (a two-dimensional Fourier transform of $R(t, \omega)$ over t and ω) is

$$A(\theta, \tau) = \frac{1}{2\pi} \int_u X(u) X^*(u - \theta) \exp(j(u - \theta)\tau) du.$$

The Cohen class of distributions, with the Rihaczek distribution as the basic distribution, is defined by

$$C(t, \omega) = \frac{1}{2\pi} \int_\theta \int_\tau A(\theta, \tau) c(\theta, \tau) \exp(-j\omega\tau) \exp(j\theta t) d\tau d\theta,$$

where $c(\theta, \tau)$ is the kernel function. Using the defined ambiguity domain form of the Rihaczek distribution $A(\theta, \tau)$ we get

$$C(t, \omega) = \frac{1}{4\pi^2} \int_u \int_v X(u) X^*(v) e^{jut} e^{-jvt} \int_\tau c(u - v, \tau) e^{-j\tau\omega} e^{j\tau u} d\tau du dv. \quad (39)$$

The frequency-frequency domain form of the Cohen class of distributions, with the Rihaczek distribution as the basis, is

$$C(t, \omega) = \int_u \int_v X(u) X^*(v) e^{jut} e^{-jvt} \phi(u - v, \omega - u) \frac{du dv}{4\pi^2},$$

where

$$\phi(u - v, \omega - u) = \int_\tau c(u - v, \tau) e^{-j\tau\omega} e^{j\tau u} d\tau.$$

The marginal properties are met if the kernel $c(\theta, \tau)$ satisfies the conditions

$$c(\theta, 0) = 1 \text{ and } c(0, \tau) = 1.$$

4.2 Reduced Interference Distributions on Graphs

We will first consider the frequency-frequency domain of the general energy distributions satisfying the marginal properties. The frequency domain definition of the presented energy distribution (38) is

$$E(n, k) = x(n) X^*(k) u_k^*(n) = \sum_{p=1}^{N} X(p) X^*(k) u_p(n) u_k^*(n).$$

Although the basis functions are commonly real-valued, here we used complex-valued notation for possible generalization to the adjacency matrices and directed graphs.

Therefore, the general graph distribution form is

$$G(n, k) = \sum_{p=1}^{N} \sum_{q=1}^{N} X(p) X^*(q) u_p(n) u_q^*(n) \phi(p, k, q). \quad (40)$$

For $\phi(p, k, q) = \delta(q - k)$ the graph Rihaczek distribution (38) follows. The unbiased energy condition

$$\sum_{k=1}^{N} \sum_{n=1}^{N} G(n, k) = E_x$$

is satisfied if

$$\sum_{k=1}^{N} \phi(p, k, p) = 1.$$

The distribution $G(n, k)$ may satisfy the vertex and frequency marginal properties:

- The vertex marginal property is satisfied if

$$\sum_{k=1}^{N} \phi(p, k, q) = 1$$

since

$$\sum_{k=1}^{N} G(n, k) = \sum_{p=1}^{N} \sum_{q=1}^{N} X(p) X^*(q) u_p(n) u_q^*(n) = |x(n)|^2.$$

The same condition is required for the vertex moment property

$$\sum_{n=1}^{N} \sum_{k=1}^{N} n^m G(n, k) = \sum_{n=1}^{N} n^m |x(n)|^2.$$

- The frequency marginal property is satisfied if

$$\phi(p, k, p) = \delta(p - k).$$

Then the sum over vertex index produces

$$\sum_{n=1}^{N} G(n, k) = \sum_{p=1}^{N} |X(p)|^2 \phi(p, k, p) = |X(k)|^2,$$

since

$$\sum_{n=1}^{N} u_p(n) u_q^*(n) = \delta(p - q),$$

that is, the eigenvectors are orthonormal. If the frequency marginal property holds, then the frequency moment property holds as well,

$$\sum_{n=1}^{N} \sum_{k=1}^{N} k^m G(n, k) = \sum_{k=1}^{N} k^m |X(k)|^2.$$

The reduced interference vertex-frequency distribution $G(n, k)$ satisfies the local smoothness property (35) if

$$\frac{\sum_{k=1}^{N}\lambda_k G(n,k)}{\sum_{k=1}^{N} G(n,k)} = \frac{\mathcal{L}_x(n)}{x(n)} = \lambda(n). \tag{41}$$

This means that

$$\frac{\sum_{k=1}^{N}\sum_{p=1}^{N}\sum_{q=1}^{N} X(p)X^*(q)u_p(n)u_q^*(n)\lambda_k\phi(p,k,q)}{\sum_{k=1}^{N}\sum_{p=1}^{N}\sum_{q=1}^{N} X(p)X^*(q)u_p(n)u_q^*(n)\phi(p,k,q)} = \lambda(n)$$

if

$$\sum_{k=1}^{N}\phi(p,k,q) = 1 \quad \text{and} \quad \sum_{k=1}^{N}\lambda_k\phi(p,k,q) = \lambda_p.$$

4.3 Reduced Interference Distribution Kernels

A few examples of the reduced interference kernels that satisfy marginal properties, will be presented next.

Choi–Williams (exponential) kernel: The classic form of this kernel is

$$c(\theta,\tau) = \exp(-\theta^2\tau^2/(2\sigma^2)).$$

The frequency-frequency form of this kernel is

$$\phi(\theta,\omega) = \mathrm{FT}_\tau\{c(\theta,\tau)\} = \exp(-\omega^2\sigma^2/(2\theta^2))|\sigma/\theta|\sqrt{2\pi}.$$

Its shifted version would be

$$\phi(u-v,\omega-u) = \frac{\sigma\sqrt{2\pi}}{|v-u|}\exp\left(-\sigma^2\frac{(\omega-u)^2}{2(v-u)^2}\right).$$

A straightforward extension to the graph signal processing would be to use the relation $\lambda \sim \omega^2$, with appropriate exponential kernel normalization. We have implemented this form and concluded that it produces results similar to the simplified form that satisfies the marginal properties and decays in the frequency-frequency domain. The form of this kernel is

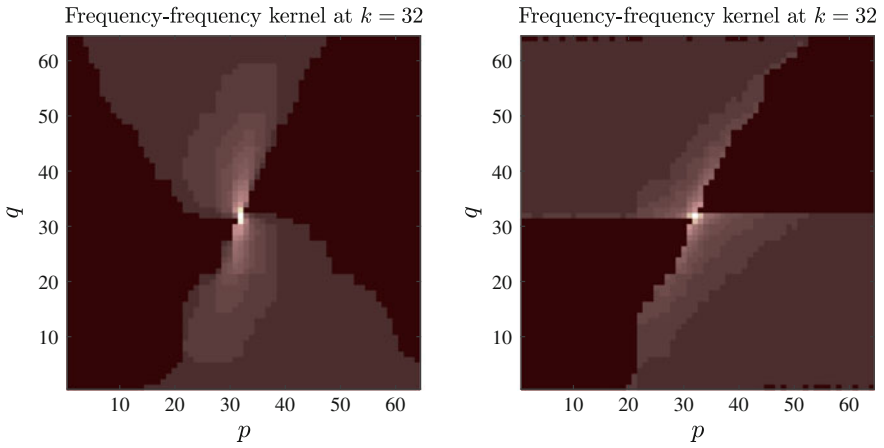


Fig. 13 Frequency-frequency domain exponential (left) and sinc (right) kernels at $k = N/2 = 32$

$$\phi(p,k,q) = \frac{1}{s(q,\,p)} \exp\Big(-\alpha \frac{|\lambda_p - \lambda_k|}{|\lambda_p - \lambda_q|}\Big),$$

where

$$s(q,\,p) = \sum_{k=1}^{N} \exp\Big(-\alpha \frac{|\lambda_p - \lambda_k|}{|\lambda_p - \lambda_q|}\Big)$$

for $q \neq p$ and $\phi(p,k,\,p) = \delta(k-p)$. It satisfies both marginal properties.

The vertex-frequency distribution with the exponential kernel (Fig. 13 (left)) is presented in Fig. 14. This kind of distribution presents correctly the signal components, preserving the marginal properties and reducing the cross-term effects as compared to Fig. 12.

Sinc kernel: The simplest reduced interference kernel in the frequency-frequency domain, that would satisfy the marginal properties, is the sinc kernel. Its form is

$$\phi(p,k,q) = \begin{cases} \frac{1}{1+2|p-q|} & \text{for } |k-p| \leq |p-q| \\ 0 & \text{otherwise,} \end{cases}$$

This kernel, with appropriate normalization, is shown in Fig. 13 (right), for $k = 32$. A vertex-frequency representation with this kernel would be similar to the one shown in Fig. 14. Additional examples can be found in [32].

Separable Kernels

If the kernel is separable, such that

$$\phi(p,k,q) = g(k-p)g(k-q),$$

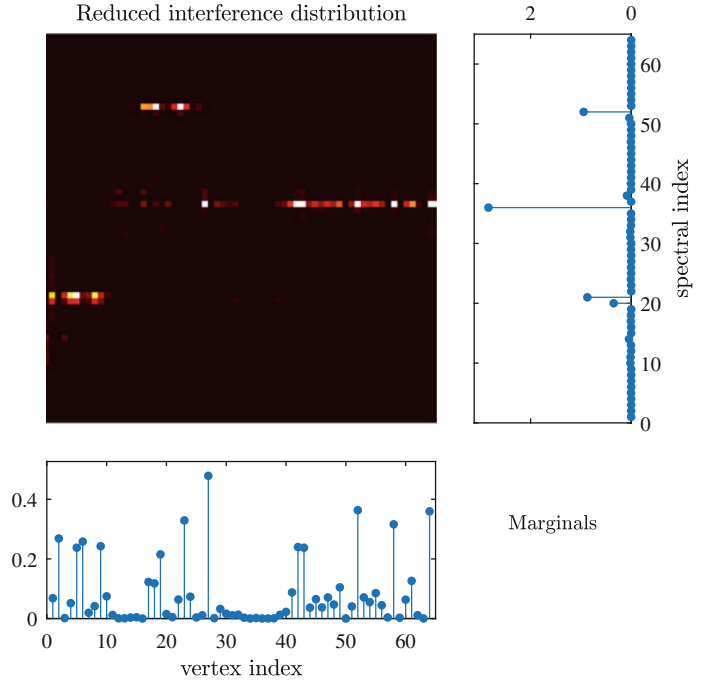


Fig. 14 Vertex-frequency reduced interference distribution using the kernel from Fig. 13(left) with its marginal values equal to $|x(n)|^2$ and $|X(k)|^2$, respectively

then we can write $G(n,k) = |\sum_{p=1}^{N} X(p)g(k-p)u_p(n)|^2$. This is a frequency domain definition of the graph spectrogram. The relation between the vertex domain spectrogram (6) and the frequency-frequency domain distribution is complex.

The separable kernels cannot satisfy the marginal properties, since $\delta(k-p) = \phi(p,k,p) = g^2(k-p)$ means $g(k-p) = \delta(k-p)$. These kernels do not satisfy $\sum_{k=1}^{N} \phi(p,k,q) = 1$ for all p and q.

Vertex-Vertex Shift Domain Distribution

The general vertex-frequency distribution can be written for the vertex-vertex shift domain as a dual form to (40)

$$G(n,k) = \sum_{m=1}^{N}\sum_{l=1}^{N} x(m)x^*(l)u_k(m)u_k^*(l)\varphi(m,n,l), \tag{42}$$

where $\varphi(m,n,l)$ is the kernel in this domain (the same mathematical form as the frequency-frequency domain kernel). The frequency marginal is satisfied if

$$\sum_{n=1}^{N} \varphi(m,n,l) = 1$$

holds. The vertex marginal is met if

$$\varphi(m,n,m)=\delta(m-n).$$

The relation of this distribution with the vertex domain spectrogram (6) is simple using

$$\varphi(m,n,l)=h_n(m)h_n^*(l)$$
$$=\sum_{p=1}^{N}\sum_{q=1}^{N}H(p)H^*(q)u_p(m)u_p(n)u_q^*(l)u_q^*(n).$$

This kernel is defined by the frequency domain window form $H(p)$. It cannot satisfy both marginal properties. The unbiased energy condition

$$\sum_{n=1}^{N}\varphi(m,n,m)=1$$

reduces to (21).

Classical Time-Frequency Analysis

The approach presented in this chapter can be extended to the directed graphs and adjacency matrices as well. The classical Fourier and time-frequency analysis follow from a directed ring graph. The adjacency matrix decomposition produces complex-valued eigenvectors of form $u_k(n)=\exp(j2\pi nk/N)/\sqrt{N}$.

4.4 Local Smoothness Estimation

For the considered signal, presented in Fig. 1, we calculate local signal smoothness by definition (33) and as the maximum positions

$$\hat{\lambda}(n)=\arg\max_k\{G(n,k)\}$$

of the various vertex-frequency representations. The results are presented in Fig. 15 and in Table 1. The local smoothness is calculated only at the vertices where signal has significant values (with instantaneous power $|x(n)|^2$ higher than 0.03 of the maximal instantaneous power $\max_n|x(n)|^2$). The exact local smoothness is presented with a line and the estimated local smoothness at the considered vertices is represented by dots. The number of outliers (vertices where estimated smoothness is not equal to the exact one) is calculated. The mean squared error of the estimations is also presented in each considered case.

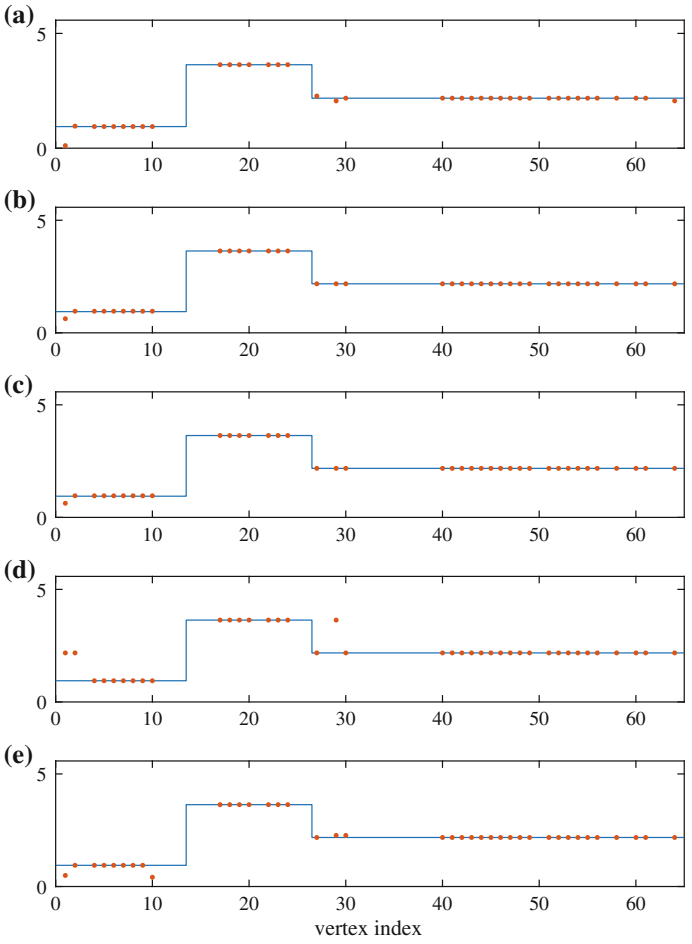


Fig. 15 Local signal smoothness for the graph signal presented in Fig. 1 estimated by using: **a** the Laplacian of the graph signal (33), **b** the vertex-frequency energy distribution (Fig. 12), **c** the reduced interference vertex frequency distribution (Fig. 14), **d** the graph spectrogram with a vertex domain window (Fig. 2), and **e** the graph spectrogram with a spectral domain window (Fig. 5). The smoothness, presented with dots, is calculated only at the vertices where $|x(n)|^2 > 0.03 \max_n |x(n)|^2$. Solid line is the exact local smoothness in the considered example

Next we considered the same signal and graph example with a white Gaussian noise added to the signal samples. The signal-to-noise ratio (SNR) is 3.6 dB. The results are given in Table 2. We can see that the local smoothness estimation based on the signal Laplacian is very sensitive to noise, while the vertex-frequency based estimations are robust with a slightly increased number of outliers in the noisy case.

Table 1 The number of outliers and mean squared error of the local smoothness for a noise-free signal

Calculation method	Number of outliers	MSE
Laplacian of the signal	4	0.019
Rihaczek's distribution	1	0.003
Reduced interference distribution	1	0.003
LVS with vertex domain window	3	0.133
LVS with spectral domain window	4	0.013

Table 2 The number of outliers and the mean squared error of the local smoothness for a noisy signal with SNR = 3.6 dB

Calculation method	Number of outliers	MSE
Laplacian of the signal	36	0.874
Rihaczek's distribution	5	0.119
Reduced interference distribution	3	0.019
LVS with vertex domain window	4	0.087
LVS with spectral domain window	8	0.120

Concentration measures obtained as the ℓ_1-norm of the Rihaczek and the reduced interference distributions [28] are 19.63 and 13.26, respectively. They confirm the fact that the reduced interference distribution has improved the concentration in the vertex-frequency domain by reducing the cross-terms.

5 Conclusion

Vertex-frequency representations of graph signals have been presented. In the first part of the chapter, the linear signal forms, along with the corresponding window forms and spectrograms, are analyzed. In the second part of the chapter, the local smoothness factor is introduced. The energy vertex-frequency distributions are presented. These distributions do not require localization windows. Energy distributions are extended to the general reduced interference distribution class. This class of graph signal distributions reduces the cross-terms and satisfies the graph signal marginal properties. The theory is illustrated through examples.

Acknowledgements Ervin Sejdić acknowledges the support of the NSF CAREER grant number 1652203.

References

1. A. Sandryhaila, J.M.F. Moura, Big data analysis with signal processing on graphs: representation and processing of massive data sets with irregular structure. IEEE Signal Process. Mag. **31**(5), 80–90 (2014)
2. S. Chen, R. Varma, A. Sandryhaila, J. Kovačević, Discrete signal processing on graphs: sampling theory. IEEE Trans. Signal Process. **63**(24), 6510–6523 (2015)
3. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs. IEEE. Trans. Signal Process. **61**(7), 1644–1656 (2013)
4. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs: frequency analysis. IEEE. Trans. Signal Process. **62**(12), 3042–3054 (2014)
5. A.G. Marques, S. Segarra, G. Leus, A. Ribeiro, Stationary graph processes and spectral estimation. IEEE Trans. Signal Process. **65** (2017). https://doi.org/10.1109/TSP.2017.2739099
6. D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process. Mag. **30**(3), 83–98 (2013)
7. I. Jestrović, J.L. Coyle, E. Sejdić, Differences in brain networks during consecutive swallows detected using an optimized vertex-frequency algorithm. Neuroscience **344**, 113–123 (2017)
8. I. Jestrović, J.L. Coyle, E. Sejdić, A fast algorithm for vertex-frequency representations of signals on graphs. Signal Process. **131**, 483–491 (2017)
9. L. Stanković, M. Daković, T. Thayaparan, *Time-Frequency Signal Analysis with Applications* (Artech House, Boston, 2013)
10. L. Cohen, *Time-Frequency Analysis* (Prentice Hall PTR, Upper Saddle River, 1995)
11. B. Boashash (ed.), *Time-Frequency Signal Analysis and Processing, A Comprehensive Reference* (Academic Press, Cambridge, 2015)
12. D.I. Shuman, B. Ricaud, P. Vandergheynst, A windowed graph Fourier transform, in *SSP Workshop* (2012), pp. 133–136
13. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**(2), 260–291 (2016)
14. X.W. Zheng, Y.Y. Tang, J.T. Zhou, H.L. Yuan, Y.L. Wang, L.N. Yang, J.J. Pan, Multi-windowed graph Fourier frames, in*2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 2 (2016), pp. 1042–1048
15. M. Tepper, S. Guillermo, A short-graph Fourier transform via personalized pagerank vectors, in *Proceedings of the IEEE ICASSP* (2016)
16. L. Stanković, M. Daković, E. Sejdić, Vertex-frequency analysis: a way to localize graph spectral components. IEEE Signal Process. Mag. **34**, 176–182 (2017)
17. S. Sardellitti, S. Barbarossa, P. Di Lorenzo, On the graph Fourier transform for directed graphs. IEEE J. Sel. Topics Signal Process. **11**(6), 796–811 (2017)
18. J.A. Deri, J.M. Moura, Spectral projector-based graph Fourier transforms. IEEE J. Sel. Topics Signal Process. **11**(6), 785–795 (2017)
19. R. Shafipour, A. Khodabakhsh, G. Mateos, E. Nikolova, A digraph Fourier transform with spread frequency components, in *5th IEEE Global Conference on Signal and Information Processing*, Montreal, Canada, 14–16 November 2017, arXiv:1705.10821
20. M. Elad, A.M. Bruckstein, A generalized uncertainty principle and sparse representation in pairs of bases. IEEE Trans. Inf. Theory **48**(9), 2558–2567 (2002). https://doi.org/10.1109/TIT.2002.801410
21. M. Tsitsvero, S. Barbarossa, P. Di Lorenzo, Signals on graphs: uncertainty principle and sampling. IEEE Trans. Signal Process. **64**(18), 4845–4860 (2016)
22. A. Agaskar, Y.M. Lu, A spectral graph uncertainty principle. IEEE Trans. Inf. Theory **59**(7), 4338–4356 (2013)
23. N. Perraudin, B. Ricaud, D. Shuman, P. Vandergheynst, Global and local uncertainty principles for signals on graphs. APSIPA Trans. Signal Inf. Process. **7**, E3 (2018). https://doi.org/10.1017/ATSIP.2018.2

24. H. Behjat, U. Richter, D. Van De Ville, L. Sornmo, Signal-adapted tight frames on graphs. IEEE Trans. Signal Process. **64**(22), 6017–6029 (2016)
25. D. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
26. A. Sakiyama, Y. Tanaka, Oversampled graph Laplacian matrix for graph filter banks. IEEE Trans. Signal Process. **62**(24), 6425–6437 (2014)
27. B. Girault, Stationary graph signals using an isometric graph translation, in *European Signal Processing Conference (EUSIPCO)* (2015), pp. 1516–1520
28. L. Stanković, A measure of some time-frequency distributions concentration. Signal Process. **81**(3), 621–631 (2001)
29. T. Thayaparan, L.J. Stanković, M. Daković, V. Popović-Bugarin, Micro-doppler parameter estimation from a fraction of the period. IET Signal Process. **4**(3), 201–212 (2010)
30. M. Daković, L. Stanković, E. Sejdić, Local graph signal smoothness and vertex-frequency representations, Submitted to Mathematical Problems in Engineering (2018)
31. L. Stanković, M. Daković, E. Sejdić, Vertex-frequency energy distributions. IEEE Signal Process. Lett. **25**(3), 358–362 (2018)
32. L. Stanković, M. Daković, E. Sejdić, Reduced interference vertex-frequency distributions. IEEE Signal Process. Lett. **25**(9), 1393–1397 (2018)

Part III
Applications

Shape Analysis of Carpal Bones Using Spectral Graph Wavelets



Majid Masoumi, Mahsa Rezaei and A. Ben Hamza

Abstract Graph signal processing is an emerging field that provides powerful tools for analyzing signals defined on graphs. In this chapter, we present a graph signal processing approach to shape analysis of carpal bones of the human wrist by exploiting local structure information among shape features for the purpose of quantitative shape comparison. We represent the cortical surface of a carpal bone in the spectral geometric setting using the Laplace-Beltrami operator and spectral graph wavelets. We propose a global spectral graph wavelet (GSGW) descriptor that is isometric invariant, efficient to compute, and combines the advantages of both low-pass and band-pass filters. We perform experiments on shapes of the carpal bones of ten women and ten men from a publicly-available database of wrist bones. Using one-way multivatiate analysis of variance (MANOVA) and permutation testing, our extensive results that the proposed GSGW framework gives a much better performance compared to the graph spectral signature (GPS) embedding approach for comparing shapes of the carpal bones across populations.

Keywords Spectral graph wavelets · Carpal bones · Shape analysis

1 Introduction

In human anatomy, the wrist (or carpus) is a complex joint that connects the hand to the forearm, and is composed of eight carpal bones arranged in two rows of four bones each. Each carpal bone has a unique shape and plays a significant functional role in the wrist stability and mobility. Changes in the shape of a carpal bone may be a sign of wrist injuries or disorders, such as arthritis and carpal tunnel syndrome, and hence understanding and analyzing variations of bone shapes is essential to the diagnosis of wrist pathologies. Quantitative shape analysis of carpal bones not only helps identify unique phenotypes across populations, but also allows for the detection of abnormal

M. Masoumi · M. Rezaei · A. B. Hamza (✉)
Institute for Information Systems Engineering, Concordia University, Montreal, Canada
e-mail: hamza@ciise.concordia.ca

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_12

wrist pathologies as well as the investigation of biomechanical properties of the wrist joints [1]. This analysis has become possible thanks in large part to the availability of databases of normal and abnormal pathologies [2].

The vast majority of quantitative shape comparison techniques rely on finding compact shape descriptors (or signatures) that capture the intrinsic geometric structure of shapes in such a way that the shape comparison problem reduces to the relatively simple problem of comparing such shape descriptors [3]. The recent surge of interest in the spectral analysis of the Laplace-Beltrami operator (LBO) has resulted in a considerable number of spectral shape signatures that have been successfully applied to a broad range of areas, including shape analysis [4–10], multimedia protection [11], and medical imaging [12]. The diversified nature of these applications is a powerful testimony of the practical usage of spectral shapes signatures, which are usually defined as feature vectors representing local and/or global characteristics of a shape and may be broadly classified into two main categories: local and global descriptors. Local descriptors (also called point signatures) are defined on each point of the shape and often represent the local structure of the shape around that point, while global descriptors are usually defined on the entire shape capturing its global structure.

Most point signatures may easily be aggregated to form global descriptors by integrating over the entire shape. One of the simplest global spectral shape signatures is Shape-DNA [4], which is an isometry-invariant global descriptor defined as a truncated sequence of the LBO eigenvalues arranged in increasing order of magnitude. Gao et al. [13] developed a variant of Shape-DNA, referred to as compact Shape-DNA (cShape-DNA), which is an isometry-invariant signature resulting from applying the discrete Fourier transform to the area-normalized eigenvalues of the LBO. Chaudhari et al. [12] proposed a global point signature (GPS) embedding for quantifying the overall bone shape, and it is obtained by setting the LBO eigenfunctions in the GPS signature [5] to unity. More precisely, the GPS embedding is defined as a truncated sequence of inverse square roots of the area-normalized eigenvalues of the LBO. In addition to providing an efficient representation for comparing shapes of the carpal bones across populations, the GPS embedding has several desirable properties for shape analysis of carpal bones, including invariance to Euclidean and isometric transformations.

While the GPS signature [5] is invariant under isometric deformations of the shape, it suffers, however, from the problem of eigenfunctions' switching whenever the associated eigenvalues are close to each other. This problem was lately well handled by the heat kernel signature (HKS) [14], which is a temporal descriptor defined as an exponentially-weighted combination of the LBO eigenfunctions. HKS is a local shape descriptor that has a number of desirable properties, including robustness to small perturbations of the shape, efficiency and invariance to isometric transformations. From the graph Fourier perspective, it can be seen that HKS is highly dominated by information from low frequencies, which correspond to macroscopic properties of a shape. To give rise to substantially more accurate matching than HKS, the wave kernel signature (WKS) [15] was proposed as an alternative in an effort to allow access to high-frequency information.

More recently, vertex-frequency analysis on graphs via the Fourier transform in the spectral graph-theoretic setting has received a great deal of interest [16, 17]. While the Fourier transform has been widely used as a reliable tool in signal processing applications for many years, wavelet analysis has been shown to provide some key advantages over the Fourier transform, making it an interesting alternative for many applications. In particular, unlike the Fourier transform, wavelet analysis is able to perform local analysis and also makes it possible to perform a multiresolution analysis. Classical wavelets are constructed by translating and scaling a mother wavelet, which is used to generate a set of functions through the scaling and translation operations. The wavelet transform coefficients are then obtained by taking the inner product of the input function with the translated and scaled waveforms. Applying wavelets directly to graphs (or triangle meshes in geometry processing) is, however, not straightforward due in large part to the fact that it is unclear how to apply the scaling operation on a signal (or function) defined on the mesh vertices. To tackle this problem, Coifman et al. [18] introduced the diffusion wavelets, which generalize the classical wavelets by allowing for multiscale analysis on graphs. The construction of diffusion wavelets interacts with the underlying graph through repeated applications of a diffusion operator, which induces a scaling process. Hammond et al. [19] showed that the wavelet transform can be performed in the graph Fourier domain, and proposed a spectral graph wavelet transform that is defined in terms of the eigensystem of the graph Laplacian matrix. Recently, a spectral graph wavelet signature (SGWS) was introduced in [20], and it has shown superior performance over HKS and WKS in 3D shape analysis. SGWS is a multiresolution local descriptor that is not only isometric invariant, but also compact, easy to compute and combines the advantages of both band-pass and low-pass filters.

In this chapter, we present a global spectral graph wavelet (GSGW) framework that represents the shape of the cortical surface of a carpal bone by a global shape descriptor defined as an area-weighted sum of all local spectral graph wavelet signatures at each surface point. The resulting global descriptor is not only isometric invariant, but also efficient to compute and requires less memory storage. Using one-way multivariate analysis of variance (MANOVA) and permutation testing, we show through experiments on a publicly-available database that our proposed GSGW approach yields better performance compared to existing methods in providing an efficient way for comparing shapes of the carpal bones across populations.

The outline of this chapter is as follows. In Sect. 2, we briefly provide a brief overview the Laplace-Beltrami operator and its spectral analysis in the discrete domain. In Sect. 3, we present a global spectral graph wavelet framework, and we discuss in detail its main algorithmic steps. Experimental results and a discussion are presented in Sects. 4 and 5. Finally, we conclude in Sect. 6 and point out some future work directions.

2 Background

A 3D shape is usually modeled as a triangle mesh $\mathbb{M}$ whose vertices are sampled from a Riemannian manifold. A triangle mesh $\mathbb{M}$ may be defined as a graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ or $\mathbb{G} = (\mathcal{V}, \mathcal{T})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ is the set of vertices, $\mathcal{E} = \{e_{ij}\}$ is the set of edges, and $\mathcal{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_l\}$ is the set of triangles. Each edge $e_{ij} = [\mathbf{v}_i, \mathbf{v}_j]$ connects a pair of vertices $\{\mathbf{v}_i, \mathbf{v}_j\}$. Two distinct vertices $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{V}$ are adjacent (denoted by $\mathbf{v}_i \sim \mathbf{v}_j$ or simply $i \sim j$) if they are connected by an edge, i.e. $e_{ij} \in \mathcal{E}$. A triangle mesh representing a carpal bone is shown in Fig. 1 (left).

2.1 Laplace-Beltrami Operator

Given a compact Riemannian manifold $\mathbb{M}$, the space $L^2(\mathbb{M})$ of all smooth, square-integrable functions on $\mathbb{M}$ is a Hilbert space endowed with inner product $\langle f_1, f_2 \rangle = \int_{\mathbb{M}} f_1(\mathbf{x}) f_2(\mathbf{x})\, da(\mathbf{x})$, for all $f_1, f_2 \in L^2(\mathbb{M})$, where $da(x)$ (or simply dx) denotes the measure from the area element of a Riemannian metric on $\mathbb{M}$. Given a twice-differentiable, real-valued function $f : \mathbb{M} \to \mathbb{R}$, the Laplace-Beltrami operator (LBO) is defined as $\Delta_{\mathbb{M}} f = -\mathrm{div}(\nabla_{\mathbb{M}} f)$, where $\nabla_{\mathbb{M}} f$ is the intrinsic gradient vector field and div is the divergence operator [21, 22]. The LBO is a linear, positive semi-definite operator acting on the space of real-valued functions defined on $\mathbb{M}$, and it is a generalization of the Laplace operator to non-Euclidean spaces.

Discretization: A real-valued function $f : \mathcal{V} \to \mathbb{R}$ defined on the mesh vertex set may be represented as an m-dimensional vector $\mathbf{f} = (f(i)) \in \mathbb{R}^m$, where the ith component $f(i)$ denotes the function value at the ith vertex in $\mathcal{V}$. Using a mixed finite element/finite volume method on triangle meshes [23], the value of $\Delta_{\mathbb{M}} f$ at a vertex $\mathbf{v}_i$ (or simply i) can be approximated using the cotangent weight scheme as follows:

$$\Delta_{\mathbb{M}} f(i) \approx \frac{1}{a_i} \sum_{j \sim i} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} \big(f(i) - f(j)\big), \tag{1}$$

where α_{ij} and β_{ij} are the angles $\angle(\mathbf{v}_i \mathbf{v}_{k_1} \mathbf{v}_j)$ and $\angle(\mathbf{v}_i \mathbf{v}_{k_2} \mathbf{v}_j)$ of two faces $\mathbf{t}^{\alpha} = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_1}\}$ and $\mathbf{t}^{\beta} = \{\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_{k_2}\}$ that are adjacent to the edge $[i, j]$, and a_i is the area of the Voronoi cell (shaded area) at vertex i, as shown in Fig. 1 (right). It should be noted that the cotangent weight scheme is numerically consistent and preserves several important properties of the continuous LBO, including symmetry and positive semi-definiteness [24].

Spectral Analysis: The $m \times m$ matrix associated to the discrete approximation of the LBO is given by $\mathbf{L} = \mathbf{A}^{-1}\mathbf{W}$, where $\mathbf{A} = \mathrm{diag}(a_i)$ is a positive definite diagonal matrix (mass matrix), and $\mathbf{W} = \mathrm{diag}(\sum_{k \neq i} c_{ik}) - (c_{ij})$ is a sparse symmetric matrix (stiffness matrix). Each diagonal element a_i is the area of the Voronoi cell at vertex i, and the weights c_{ij} are given by

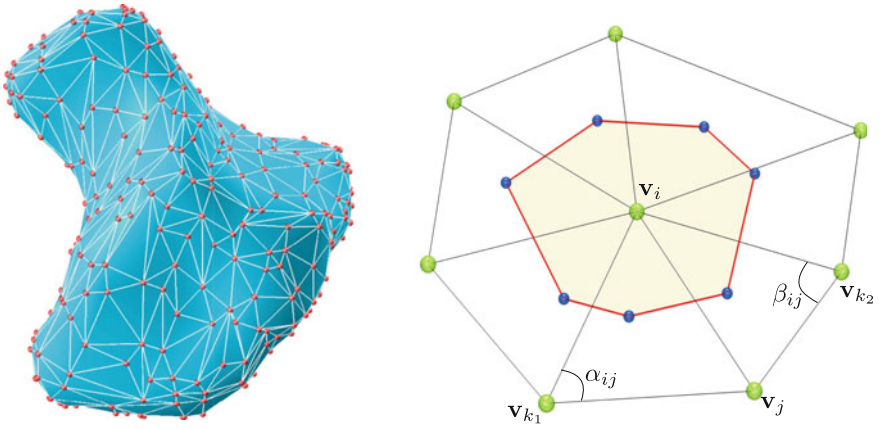


Fig. 1 Triangular mesh representation of a carpal bone (left); Cotangent scheme angles (right)

$$c_{ij} = \begin{cases} \dfrac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} & \text{if } i \sim j \\ 0 & \text{o.w.} \end{cases} \tag{2}$$

where α_{ij} and β_{ij} are the opposite angles of two triangles that are adjacent to the edge $[i, j]$.

The eigenvalues and eigenvectors of $\mathbf{L}$ can be found by solving the generalized eigenvalue problem $\mathbf{W}\boldsymbol{\varphi}_\ell = \lambda_\ell \mathbf{A}\boldsymbol{\varphi}_\ell$ using, for instance, the Arnoldi method of ARPACK,[1] where λ_ℓ are the eigenvalues and $\boldsymbol{\varphi}_\ell$ are the unknown associated eigenfunctions (i.e. eigenvectors which can be thought of as functions on the mesh vertices). We may sort the eigenvalues in ascending order as $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_m$ with associated orthonormal eigenfunctions $\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2, \ldots, \boldsymbol{\varphi}_m$, where the orthogonality of the eigenfunctions is defined in terms of the $\mathbf{A}$-inner product, i.e.

$$\langle \boldsymbol{\varphi}_k, \boldsymbol{\varphi}_\ell \rangle_{\mathbf{A}} = \sum_{i=1}^{m} a_i \varphi_k(i) \varphi_\ell(i) = \delta_{k\ell}, \quad \text{for all } k, \ell = 1, \ldots, m. \tag{3}$$

We may rewrite the generalized eigenvalue problem in matrix form as $\mathbf{W}\Phi = \mathbf{A}\Phi\Lambda$, where $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_m)$ is an $m \times m$ diagonal matrix with the λ_ℓ on the diagonal, and Φ is an $m \times m$ orthogonal matrix whose ℓth column is the unit-norm eigenvector $\boldsymbol{\varphi}_\ell$.

The successful use of the LBO eigenvalues and eigenfunctions in shape analysis is largely attributed to their isometry invariance and robustness to noise. Moreover, the eigenfunctions associated to the smallest eigenvalues capture well the large-scale properties of a shape. As shown in Fig. 2, the (non-trivial) eigenfunctions of the LBO

[1] ARPACK (ARnoldi PACKage) is a MATLAB library for computing the eigenvalues and eigenvectors of large matrices.

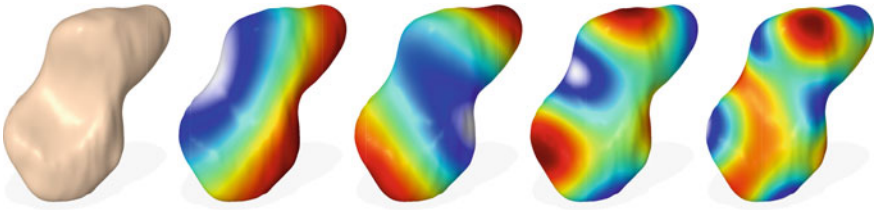


Fig. 2 From left to right: 3D scaphoid bone and selected eigenfunctions (2nd, 4th, 8th and 18th) of the LBO mapped into the surface of the bone

encode important information about the intrinsic global geometry of a shape. Notice that the eigenfunctions associated with larger eigenvalues oscillate more rapidly. Blue regions indicate negative values of the eigenfunctions and red colors regions indicate positive values, while green and yellow regions in between.

3 Method

In this section, we provide a detailed description of our GSGW framework for the analysis of the cortical surface of a carpal bone using spectral graph wavelets. We start by defining the spectral graph wavelet transform on a Riemannian manifold. We show how to build local descriptors from spectral graph wavelets and its subcomponent functions. Then, we propose a novel global shape descriptor defined as an area-weighted sum of all local spectral graph wavelet signatures at each mesh vertex. Finally, we provide the main algorithmic steps of our carpal bone analysis framework.

3.1 Local Descriptors

Graph Fourier Transform: For any graph signal $f : \mathcal{V} \to \mathbb{M}$, the forward and inverse graph Fourier transforms (also called manifold harmonic and inverse manifold harmonic transforms) are defined as

$$\hat{f}(\ell) = \langle f, \varphi_\ell \rangle = \sum_{i=1}^{m} a_i f(i) \varphi_\ell(i), \quad \ell = 1, \ldots, m \tag{4}$$

and

$$f(i) = \sum_{\ell=1}^{m} \hat{f}(\ell) \varphi_\ell(i) = \sum_{\ell=1}^{m} \langle f, \varphi_\ell \rangle \varphi_\ell(i), \quad i \in \mathcal{V}, \tag{5}$$

respectively, where $\hat{f}(\ell)$ is the value of $\hat{f}$ at eigenvalue λ_ℓ (i.e. $\hat{f}(\ell) = \hat{f}(\lambda_\ell)$). In particular, the graph Fourier transform of a delta function δ_j centered at vertex j is given by

$$\hat{\delta}_j(\ell) = \sum_{i=1}^{n} a_i \delta_j(i) \varphi_\ell(i) = \sum_{i=1}^{n} a_i \delta_{ij} \varphi_\ell(i) = a_j \varphi_\ell(j),$$

The forward and inverse graph Fourier transforms may be expressed in vector form as follows:

$$\hat{\mathbf{f}} = \Phi^{\mathsf{T}} \mathbf{A} \mathbf{f} \quad \text{and} \quad \mathbf{f} = \Phi \hat{\mathbf{f}}, \tag{6}$$

where $\mathbf{f} = (f(i))$ and $\hat{\mathbf{f}} = (\hat{f}(\ell))$ are m-dimensional vectors, whose elements are given by (4) and (5), respectively. The vector $\hat{\mathbf{f}}$ represents the signal's graph Fourier series expansion in the area-weighted eigenvector basis and describes the frequency components of the graph signal $\mathbf{f}$.

The inverse graph Fourier transform reconstructs the graph signal by combining graph frequency components, as shown in Fig. 3, which demonstrate the ability of the LBO eigenfunctions in rendering the shape-based features. As can be seen, the lower-order eigenfunctions capture the global structure of shape, while by increasing the number of eigenfunctions more details of the curvature of the bone are captured.

The normalized mean squared error between the original bone surface and its graph Fourier reconstruction is shown in Fig. 4, where the x-axis is the number of eigenfunctions of the LBO. As can be seen, a relatively small number of eigenfunctions (i.e. between 20 and 30) would be enough to efficiently capture the features of the carpal bone surface to analyze shape differences in a population study. By features, we mean the points on the carpal bones that contain salient information about the shape (e.g. protrusions). The extracted features should be robust to transformations. Our global spectral graph wavelet descriptor is a dense descriptor that makes use of the LBO eigenvalues and eigenfunctions, which are invariant to isometric transformation. Hence, our framework provides a robust descriptor for describing the carpal bones that helps facilitate the statistical analysis among bone shapes. Rendering a carpal bone surface in a lower-dimension has some advantages, including the ability of being invulnerable to tessellation noise or image segmentation.

Spectral Graph Wavelet Transform: Wavelets are useful in describing functions at different levels of resolution. To characterize the localized context around a mesh vertex $j \in \mathcal{V}$, we assume that the signal on the mesh is a unit impulse function, that



Fig. 3 From left to right: 3D hamate bone in a healthy man and its graph Fourier reconstruction using 10, 30, 75 and 150 eigenfunctions of the LBO

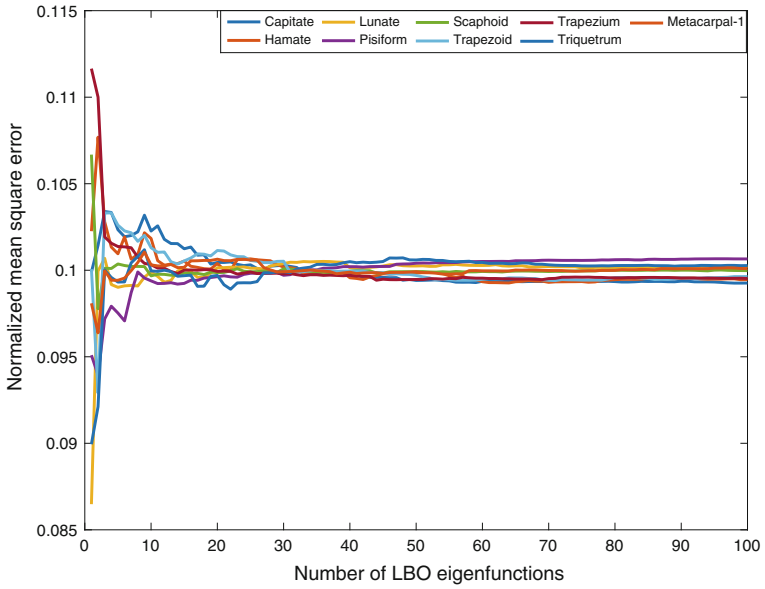


Fig. 4 Normalized mean squared error between the original carpal bone and its graph Fourier reconstruction as a function of the LBO eigenfunctions

is $f(i) = \delta_j(i)$ at each mesh vertex $i \in \mathcal{V}$. The spectral graph wavelet coefficients are expressed as

$$W_{\delta_j}(t, j) = \langle \boldsymbol{\delta}_j, \psi_{t,j} \rangle = \sum_{\ell=1}^{m} a_j^2 g(t\lambda_\ell)\varphi_\ell^2(j), \tag{7}$$

and that the coefficients of the scaling function are

$$S_{\delta_j}(j) = \sum_{\ell=1}^{m} a_j^2 h(\lambda_\ell)\varphi_\ell^2(j). \tag{8}$$

Following the multiresolution analysis, the spectral graph wavelet and scaling function coefficients are collected to form the spectral graph wavelet signature at vertex j as follows:

$$\mathbf{s}_j = \{\mathbf{s}_L(j) \mid L = 1, \ldots, R\}, \tag{9}$$

where R is a resolution parameter, and $\mathbf{s}_L(j)$ is the shape signature at resolution level L given by

$$\mathbf{s}_L(j) = \{W_{\delta_j}(t_k, j) \mid k = 1, \ldots, L\} \cup \{S_{\delta_j}(j)\}. \tag{10}$$

The wavelet scales t_k ($t_k > t_{k+1}$) are selected to be logarithmically equispaced between maximum and minimum scales t_1 and t_L, respectively. Thus, the resolution level L determines the resolution of scales to modulate the spectrum. At resolution $R = 1$, the spectral graph wavelet signature $\mathbf{s}_j$ is a 2-dimensional vector consisting of two elements: one element, $W_{\delta_j}(t_1, j)$, of spectral graph wavelet function coefficients and another element, $S_{\delta_j}(j)$, of scaling function coefficients. And at resolution $R = 2$, the spectral graph wavelet signature $\mathbf{s}_j$ is a 5-dimensional vector consisting of five elements (four elements of spectral graph wavelet function coefficients and one element of scaling function coefficients). In general, the dimension of a spectral graph wavelet signature $\mathbf{s}_j$ at vertex j can be expressed in terms of the resolution R as follows:

$$p = \frac{(R+1)(R+2)}{2} - 1. \tag{11}$$

Hence, for a p-dimensional signature $\mathbf{s}_j$, we define a $p \times m$ spectral graph wavelet signature matrix as $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_m)$, where $\mathbf{s}_j$ is the signature at vertex j and m is the number of mesh vertices. In our implementation, we used the Mexican hat wavelet

$$g(x) = \frac{2}{\sqrt{3}\,\pi^{1/4}}(1 - x^2)\exp\left(-\frac{x^2}{2}\right), \tag{12}$$

as a kernel generating function. In addition, we used the scaling function h given by

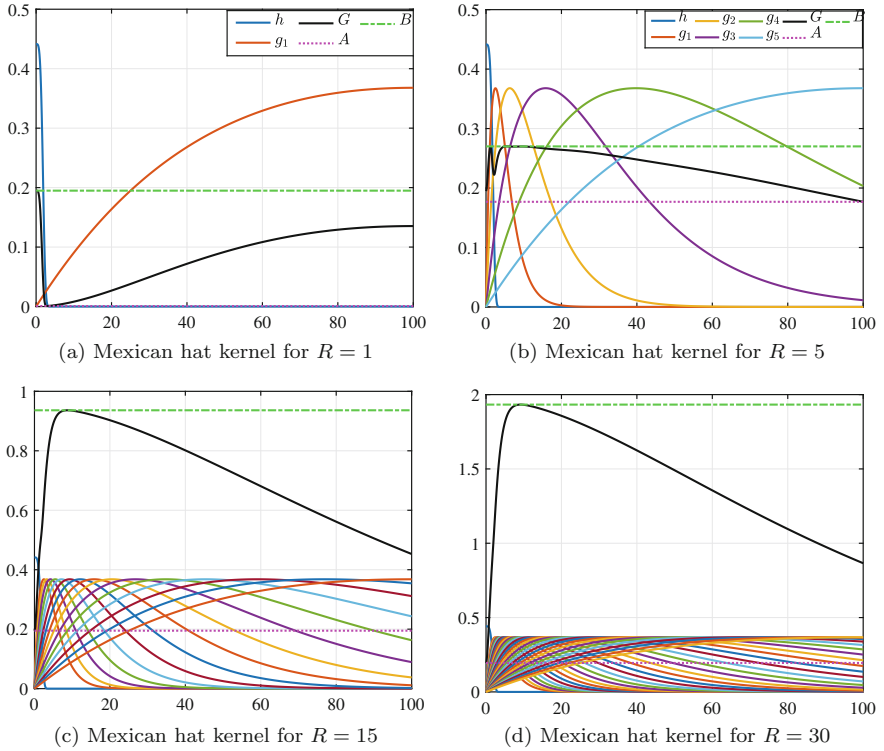
$$h(x) = \gamma \exp\left(-\left(\frac{x}{0.6\lambda_{\min}}\right)^4\right), \tag{13}$$

where $\lambda_{\min} = \lambda_{\max}/20$ and γ is set such that $h(0)$ has the same value as the maximum value of g. The maximum and minimum scales are set to $t_1 = 2/\lambda_{\min}$ and $t_L = 2/\lambda_{\max}$, where $\lambda_{\min}$ and $\lambda_{\max}$ are the smallest and largest LBO eigenvalues, respectively.

The geometry captured at each resolution R of the spectral graph wavelet signature can be viewed as the area under the curve G shown in Fig. 5. For a given resolution R, we can understand the information from a specific range of the spectrum as its associated areas under G. As the resolution R increases, the partition of spectrum becomes tighter, and thus a larger portion of the spectrum is highly weighted.

3.2 Global Descriptor

A commonly used methodology for building a global shape descriptor is by aggregating local signatures using the bag-of-features (BoF) paradigm [6]. The BoF model represents each object in the dataset as a collection of unordered feature descriptors extracted from local areas of the shape, just as words are local features of a document. A baseline BoF approach quantizes each local descriptor to its nearest cluster



(a) Mexican hat kernel for $R = 1$ (b) Mexican hat kernel for $R = 5$

(c) Mexican hat kernel for $R = 15$ (d) Mexican hat kernel for $R = 30$

Fig. 5 Spectrum modulation using different kernel functions at various resolutions. The dark line is the squared sum function G, while the dash-dotted and the dotted lines are upper and lower bounds (B and A) of G, respectively

center using K-means clustering and then encodes each shape as a histogram over cluster centers by counting the number of assignments per cluster. These cluster centers form a visual vocabulary or codebook whose elements are often referred to as visual words or codewords. Although the BoF paradigm has been shown to provide significant levels of performance, it does not, however, take into consideration the spatial relations between features, which may have an adverse effect not only on its descriptive ability but also on its discriminative power. To circumvent these challenges, we represent a shape $\mathbb{M}$ by a p-dimensional vector

$$\mathbf{x} = \mathbf{S}\mathbf{a} = \sum_{i=1}^{m} a_i \mathbf{s}_i, \tag{14}$$

where $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_m)$ is a $p \times m$ matrix of local spectral graph wavelet signatures and $\mathbf{a} = (a_1, \ldots, a_m)^\mathsf{T}$ is an m-dimensional vector of mesh vertex areas (i.e. each element a_i is the area of the Voronoi cell at mesh vertex i).

We refer to the p-dimensional vector $\mathbf{x}$ as the global spectral graph wavelet (GSGW) descriptor of the carpal bone surface. The GSGW descriptor enjoys a number of desirable properties including simplicity, compactness, invariance to isometric deformations, and computational feasibility. Moreover, GSGW combines the advantages of both band-pass and low-pass filters.

3.3 Proposed Algorithm

Our proposed carpal bone analysis algorithm consists of two main steps. In the first step, we represent each bone in the dataset by a spectral graph wavelet signature matrix, which is a feature matrix consisting of local descriptors. More specifically, let $\mathcal{D}$ be a dataset of n carpal bones modeled by triangle meshes $\mathbb{M}_1, \ldots, \mathbb{M}_n$. We represent each surface $\mathbb{M}_i$ in the dataset $\mathcal{D}$ by a $p \times m$ spectral graph wavelet signature matrix $\mathbf{S}_i$, whose columns are p-dimensional local signatures and m is the number of mesh vertices.

In the second step, we compute the p-dimensional global spectral graph wavelet descriptor $\mathbf{x}_i = \mathbf{S}_i \mathbf{a}_i$ of each carpal bone $\mathbb{M}_i$, for $i = 1, \ldots, n$. Subsequently, the feature vectors $\mathbf{x}_i$ of all n shapes in the dataset are arranged into a $n \times p$ data matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\intercal$.

To assess the performance of the proposed GSGW framework, we employed two commonly-used evaluation criteria, namely MANOVA and permutation testing. MANOVA is a multivariate data analysis technique used to determine whether there are any statistical differences between independent groups on more than one continuous dependent variable, while a permutation test in a non-parametric test that resamples the observed data many times in order to determine a p-value for the test. The p-value is the probability of obtaining an effect at least as extreme as the one in our observed data when the null hypothesis is true, and it basically measures how compatible our data are with the null hypothesis. A small p-value provides enough evidence that we can reject the null hypothesis. Algorithm 1 summarizes the main algorithm steps of our GSGW approach.

Algorithm 1 GSWS approach

Input: Dataset $\mathcal{D} = \{\mathbb{M}_1, \ldots, \mathbb{M}_n\}$ of n carpal bones

1: **for** $i = 1$ to n **do**
2: Compute the $p \times m$ spectral graph wavelet matrix $\mathbf{S}_i$ for each carpal bone $\mathbb{M}_i$, where m is the number of vertices
3: Compute the p-dimensional vector $\mathbf{x}_i = \mathbf{S}_i \mathbf{a}_i$, where $\mathbf{a}_i$ is an m-dimensional vector of vertex areas
4: **end for**
5: Arrange all the feature vectors $\mathbf{x}_i$ into a $n \times p$ data matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\intercal$
6: Perform MANOVA and permutation test on $\mathbf{X}$ to quantify the statistical differences between carpal bones

Output: p-values for MANOVA and permutation test.

4 Experimental Results

In this section, we evaluate the performance of our proposed GSGW approach on the analysis of carpal bone surfaces via extensive experiments. The effectiveness of our method is validated by performing a comprehensive comparison with the global point signature embedding approach [12].

Datasets: In order to evaluate the performance of our GSGW framework on carpal bone surfaces, a total of 20 men and women with average age of 25 years old from a publicly-available benchmark [2] have been chosen. In this dataset, the bones of the wrist are obtained from CT volume images. More precisely, the carpal bones undergo segmentation by detecting the 2D outer cortical bone contours in each image slice. For each bone, the contours are identified and then aggregated into a single 3D point cloud. Finally, a triangular mesh is constructed using the acquired points. Each triangular mesh consists of an edge set (i.e. connectivity list) and vertex locations (i.e. vertex set). We also performed uniform sampling on triangular meshes to have an equal number of vertices. As shown in Fig. 6, the carpal bones of the right wrist in a healthy male are eight irregularly shaped bones that are organized into two rows: proximal and distal. In the proximal row, the bones are scaphoid, lunate, triquetrum and pisiform. In the distal row, the bones are trapezium, trapezoid, capitate and hamate. The five metacarpal bones connect the wrist with the fingers, and articulate proximally with the carpal bones and distally with the fingers (one metacarpal for each finger). The first metacarpal bone is associated with the thumb, while the fifth metacarpal bone is associated with the little finger. Since the trapeziometacarpal joint of the thumb is a common site of osteoarthritis, the first metacarpal bone is also considered in our analysis. The forearm's radius and ulna bones, which support the many muscles that manipulate the bones of the hand and wrist, are also depicted in Fig. 6.

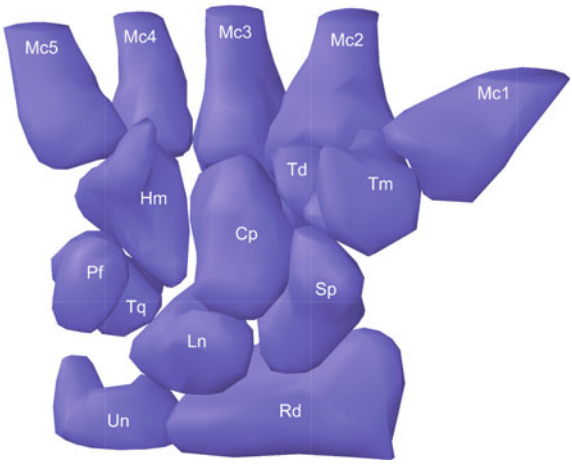


Fig. 6 Carpal bone anatomy of a healthy male from a palmar view. The carpus consists of eight carpal bones, which are arranged in proximal and distal rows. The proximal row contains scaphoid (Sp), lunate (Ln), triquetrum (Tq) and pisiform (Pf), while the distal row contains trapezium (Tm), trapezoid (Td), capitate (Cp) and hamate (Hm). The distal row adjoins the five metacarpals (Mc1-5) of the wrist. The radius (Rd) and ulna (Un) are also shown

Performance Evaluation Measures: To compare the shapes of the carpal bones in women versus men, we computed the GSGW descriptor for each carpal bone (eight in total) and the first metacarpal bone for each subject for both the right and left wrists. For fair comparison between the proposed GSGW approach and GPS embedding method, we followed the same settings described in [2]. In order to quantify the difference between the sexes for the carpal bone shapes, we compared GSGW to GPS embedding for each bone of the right and left wrist separately for the two groups (ten women versus ten men) using MANOVA and permutation testing.

For permutation testing, gender labels of the samples are randomly shuffled for 1000 times to get the correct distribution of a test statistic under a null hypothesis. We report the p-values generated by MANOVA and permutation testing. For $p < 0.05$, there would be a statistically significant difference between the two groups.

Baseline Method: For the wrist benchmark [2] used for experimentation, we report the comparison results of our method against the GPS embedding approach [12].

Implementation Details: The experiments were conducted on a desktop computer with an Intel Core i5 processor running at 3.10 GHz and 8 GB RAM; and all the algorithms were implemented in MATLAB. The appropriate dimension (i.e. length or number of features) of a shape signature is problem-dependent and usually determined experimentally. For fair comparison, we used the same parameters that have been employed in the baseline method, and in particular the dimensions of shape descriptors. In our setup, a total of 31 eigenvalues and associated eigenfunctions of the LBO were computed. We also set the resolution parameter to $R = 30$, resulting in a 495-dimensional GSGW descriptor.

4.1 Carpal Bone Dataset

The carpal bone dataset consists of 360 mesh models from 20 classes [2]. The bones of the wrist are obtained from the CT volume images, and then the carpal bones are rendered and represented as triangular mesh models. Each class contains 18 objects with distinct postures. Moreover, each model in the dataset has approximately $m = 1502$ vertices.

Results: In our GSGW approach, each surface in the carpal bone dataset is represented by a 495×1502 matrix of spectral graph wavelet signatures, resulting in a data matrix $\mathbf{X}$ of size 495×360. Figure 7 shows the spectral graph wavelet descriptors of two carpal bones (capitate and lunate) from two different classes of the carpal bone dataset. As can be seen, the global descriptors are quite different and hence may be used to efficiently discriminate between surfaces in statistical analysis tasks.

We compared the proposed GSGW method to the GPS embedding approach by performing MANOVA and non-parametric permutation testing. The results are summarized in Table 1 for the right wrist and Table 2 for the left wrist. In these tables, the numbers marked with an asterisk indicate that the p-value exceeds the significance level of 0.05.

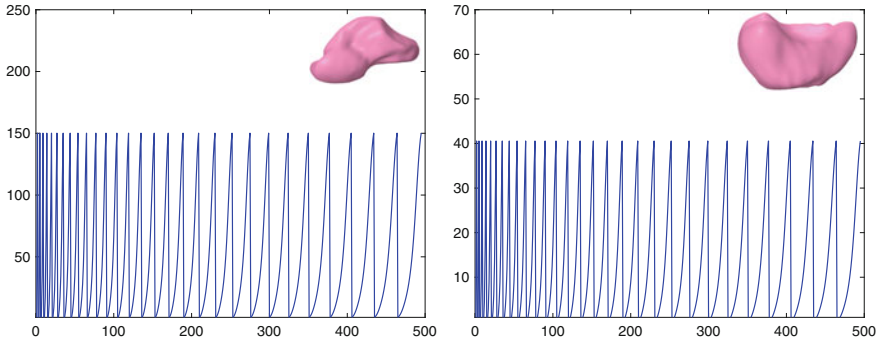


Fig. 7 Global spectral graph wavelet descriptors of two carpal bones: capitate (left) and lunate (right)

Table 1 Comparison of carpal bone surfaces of the right wrist between males and females using MANOVA and permutation test. Boldface numbers indicate the better performance, while the numbers marked with an asterisk indicate that the p-value exceeds 0.05

Bone	MANOVA		Permutation test	
	GPS	GSGW	GPS	GSGW
Capitate	**0.0226**	0.0319	0.0493	**0**
Hamate	0.0065	**2.37e-11**	0.0097	**0.0020**
Lunate	0.0379	**0.0069**	0.0210	**0**
Pisiform	**0.0428**	0.0441	0.0365	**0.0040**
Scaphoid	0.0135	**0.0003**	0.0255	**0**
Trapezoid	**0.0004**	0.0088	0.0087	**0**
Trapezium	0.0007	**1.80e-5**	**0.0101**	0.0200
Triquetrum	0.0015	**1.96e-5**	**0.0124**	0.200*
Metacarpal-1	0.0137	**0.0009**	0.0245	**0.0040**

As can be seen, our method achieves better analytical performance than the GPS embedding method for both right and left wrist. For the right wrist, the GSGW approach yields the lower p-value compared to GPS embedding for six carpal bones out of nine using MANOVA, and for seven bones out of nine using permutation testing. In addition, the p-value in the MANOVA test for some bones (e.g. hamate) has decreased to 6.5×10^{-3}. For the left wrist, our GSGW approach significantly improves the results by yielding a lower p-value in the MANOVA test for all bones, except for Metacarpal-1. Also, unlike GPS embedding, our method achieved a much lower p-value in the permutation test for all carpal bones. Moreover, the p-value in the MANOVA test for some bones (e.g. hamate) has plummeted to 2.5×10^{-3}. To speed-up experiments, all shape descriptors were computed offline, albeit their computation is quite inexpensive due in large part to the fact that only a relatively small number of the LBO eigenfunctions need to be computed.

Table 2 Comparison of carpal bone surfaces of the left wrist between males and females using MANOVA and permutation test. Boldface numbers indicate the better performance, while the numbers marked with an asterisk indicate that the p-value exceeds 0.05

Bone	MANOVA		Permutation test	
	GPS	GSGW	GPS	GSGW
Capitate	0.0148	**0.0065**	0.0416	**0.0100**
Hamate	0.0048	**6.64e-8**	0.0089	**0.0020**
Lunate	0.0532*	**2.41e-5**	0.0766*	**0.0020**
Pisiform	0.0102	**2.56e-5**	0.0201	**0**
Scaphoid	0.0012	**3.19e-5**	0.0120	**0.0060**
Trapezoid	0.0022	**0.0013**	0.0136	**0.0020**
Trapezium	0.0023	**0.0002**	0.0140	**0**
Triquetrum	0.0449	**0.0005**	0.0836*	**0**
Metacarpal-1	**0.0015**	0.0482	0.0129	**0.0100**

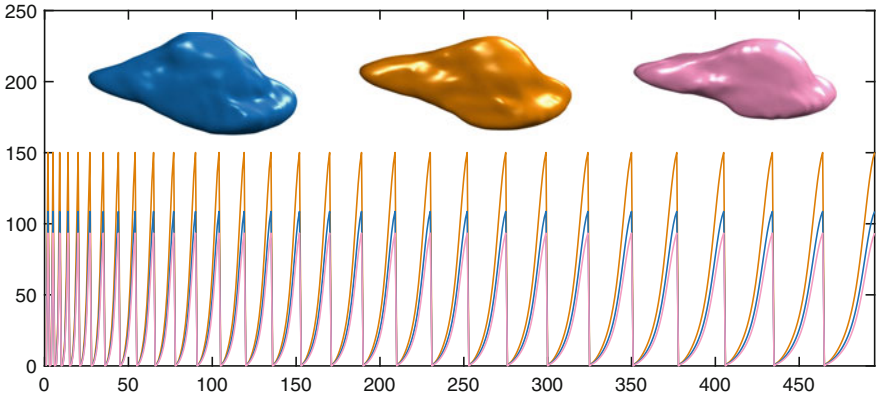


Fig. 8 Global spectral graph wavelet descriptors for three capitate bones of women's left wrists

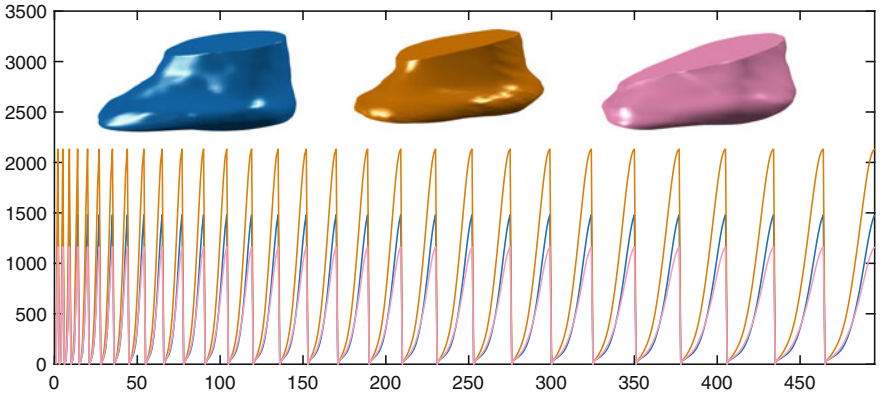


Fig. 9 Global spectral graph wavelet descriptors for three metacarpal bones of women's left wrists

To further assess the discriminative power of our approach, we computed the GSGW descriptors of carpal bone surfaces from the same class. As shown in Figs. 8 and 9, even for very similar carpal bones with a slightly difference, the proposed GSGW approach is able to distinguish between the shapes.

5 Discussion

Using the eigensystem of Laplace-Beltrami operator and spectral graph wavelets, we presented a global descriptor that characterizes the shape of the cortical surface of a carpal bone. This global descriptor enjoys a number of desirable properties including simplicity, compactness, invariance to isometric deformations, and computational feasibility. Moreover, the experimental results on the carpal bone dataset demonstrate the effectiveness of the proposed GSGW framework.

To assess the performance of the GSGW approach against orientation misalignment, we randomly selected three shapes from each group in the carpal bone dataset. Then, we performed scaling, translation and rotation on these shapes using factors of 10, 50 and 100, respectively. The global spectral graph wavelet descriptors of the transformed and reference shapes are computed along with the normalized mean squared error between them. We achieved an average of 6.1×10^{-8} in terms of NMSE for all experiments, indicating that our method is independent of position, rotation and scaling. Such nice attributes may ameliorate the assessment of the carpal bone shape and hence make it invariant to orientation misalignment.

Our GSGW approach is able to decompose a carpal bone surface into its constituent intrinsic components and represent each surface by a global spectral graph wavelet descriptor. This helps facilitate the statistical analysis of the bones of the wrists. Moreover, our method needs only a small number of eigenfunctions to identify the variations in the surface, and can easily distinguish not only between carpal bones of different groups but also bones from the same group. This property tremendously simplifies the problem and speeds-up the computation of the shape descriptor, particularly when dealing with shapes consisting of thousands of vertices, albeit increasing the number of mesh vertices tends to provide slightly better results. Our experiments show that 20–30 eigenfunctions are enough to capture the discriminative features of 3D shapes. It should be pointed out that the extracted features derived from the wrist bones using our proposed GSGW framework may also be used to develop subject-specific prostheses or implants [1].

While the carpal bone dataset used is this study consists of a small size of individuals, we plan in the not too distant future to test the GSGW approach on a larger population. The carpal bone dataset includes subjects who are healthy, young and who lacked remarkable pathological findings. So, it would be interesting to investigate surface differences in wrist bones when, for instance, a group is suffering from specific pathologies like osteoarthritis or wrist instability. Moreover, this dataset does not include information related to the dominant hand of the individuals. Hence, we may consider this information to analyze the bone shape differences in an effort to

find the probable association. Since men and women in the carpal bone dataset are in the same age range, the factors related to age can also be investigated by considering a different age range in two groups of men and women. In addition, the other factors that may change the carpal bone shapes of individuals include the body size, genetic, metabolic and environmental factors, which are not taken into consideration in our experiments because they are not available in the carpal bone dataset.

The GSGW framework can be efficiently used to assess bone shape changes across populations as well as the identification of abnormalities in the wrist bones. Carpal bone erosion is one of the most common anatomical consequences and erosive changes in rheumatoid arthritis, which needs to be regularly monitored in order to track illness progression [25]. The capitate, lunate, triquetrum and scaphoid are usually the most affected carpal bones. Evaluating group differences in carpal bone surfaces based on sex can be used to track the bone erosion status in rheumatoid arthritis. Furthermore, in paleoanthropology studies, the evolutionary development and diversification (i.e. phylogenetic relationships) among mammalian species can be analyzed with carpal bone morphology. Therefore, quantifying the bone shape differences due to evolution is of high clinical interest [26].

6 Conclusion

We presented a spectral graph wavelet framework for quantitative shape comparison of carpal bones. In particular, we proposed a a conceptually simple yet powerful and theoretically motivated global shape representation for the cortical surface of a carpal bone. We performed a statistical analysis using MANOVA and permutation testing on a database of carpal bones of the human wrist in an effort to compare shapes of the carpal bones across populations. The proposed GSGW descriptor enjoys a number of desirable properties including simplicity, compactness, invariance to isometric deformations, and computational feasibility. Our extensive results show that our approach not only captures the similarity between feature descriptors, but also substantially outperforms existing methods. In the future work, we plan to generalize the GSGW framework by including other factors that may change the shape of carpal bones.

References

1. J. Crisco, J. Coburn, D. Moore, M. Upal, Carpal bone size and scaling in men versus in women. J. Hand Surg. **30**(1), 35–42 (2005)
2. D. Moore, J. Crisco, T. Trafton, E. Leventhal, A digital database of wrist bone anatomy and carpal kinematics. J. Biomech. **40**(11), 2537–2542 (2007)
3. A. Bronstein, M. Bronstein, R. Kimmel, *Numerical Geometry of Non-rigid Shapes* (Springer, Berlin, 2008)

4. M. Reuter, F. Wolter, N. Peinecke, Laplace-Beltrami spectra as Shape-DNA of surfaces and solids. Comput.-Aided Design **38**(4), 342–366 (2006)
5. R. Rustamov, Laplace-Beltrami eigenfunctions for deformation invariant shape representation, in *Proceedings of the Symposium on Geometry Processing* (2007), pp. 225–233
6. A. Bronstein, M. Bronstein, L. Guibas, M. Ovsjanikov, Shape Google: geometric words and expressions for invariant shape retrieval. ACM Trans. Graph. **30**(1), 00 (2011)
7. S. Biasotti, A. Cerri, M. Abdelrahman, M. Aono, A. Ben Hamza, M. El-Melegy, A. Farag, V. Garro, A. Giachetti, D. Giorgi, A. Godil, C. Li, Y.-J. Liu, H. Martono, C. Sanada, A. Tatsuma, S. Velasco-Forero, C.-X. Xu, SHREC'14 track: retrieval and classification on textured 3D models, in *Proceedings of the Eurographics Workshop on 3D Object Retrieval* (2014), pp. 111–120
8. M. Masoumi, C. Li, A. Ben Hamza, A spectral graph wavelet approach for nonrigid 3D shape retrieval. Pattern Recognit. Lett. **83**, 339–348 (2016)
9. E. Rodola, L. Cosmo, O. Litany, M.M. Bronstein, A.M. Bronstein, N. Audebert, A. Ben Hamza, A. Boulch, U. Castellani, M.N. Do, A.-D. Duong, T. Furuya, A. Gasparetto, Y. Hong, J. Kim, B.L. Saux, R. Litman, M. Masoumi, G. Minello, H.-D. Nguyen, V.-T. Nguyen, R. Ohbuchi, V.-K. Pham, T.V. Phan, M. Rezaei, A. Torsello, M.-T. Tran, Q.-T. Tran, B. Truong, L. Wan, C. Zou11, SHREC'17 track: Deformable shape retrieval with missing parts, in *Proceedings of the Eurographics Workshop on 3D Object Retrieval 2017* (2017), pp. 1–9
10. M. Masoumi, A. Ben Hamza, Spectral shape classification: a deep learning approach. J. Vis. Commun. Image Represent. **43**, 198–211 (2017)
11. E. Elsheh, A. Ben Hamza, Secret sharing approaches for 3D object encryption. Expert Syst. Appl. **38**(11), 13906–13911 (2011)
12. A. Chaudhari, R. Leahy, B. Wise, N. Lane, R. Badawi, A. Joshi, Global point signature for shape analysis of carpal bones. Phys. Med. Biol. **59**, 961–973 (2014)
13. Z. Gao, Z. Yu, X. Pang, A compact shape descriptor for triangular surface meshes. Comput.-Aided Design **53**, 62–69 (2014)
14. J. Sun, M. Ovsjanikov, L. Guibas, A concise and provably informative multi-scale signature based on heat diffusion. Comput. Graph. Forum **28**(5), 1383–1392 (2009)
15. M. Aubry, U. Schlickewei, D. Cremers, The wave kernel signature: A quantum mechanical approach to shape analysis, in *Proceedings of the Computational Methods for the Innovative Design of Electrical Devices* (2011), pp. 1626–1633
16. D. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**(2), 260–291 (2016)
17. L. Stankovic, E. Sejdic, M. Dakovic, Vertex-frequency energy distributions. IEEE Signal Process. Lett. (2017)
18. R. Coifman, S. Lafon, Diffusion maps. Appl. Comput. Harmon. Anal. **21**(1), 5–30 (2006)
19. D. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
20. C. Li, A. Ben Hamza, A multiresolution descriptor for deformable 3D shape retrieval. Vis. Comput. **29**, 513–524 (2013)
21. S. Rosenberg, *The Laplacian on a Riemannian Manifold* (Cambridge University Press, New York, 1997)
22. H. Krim, A. Ben Hamza, *Geometric Methods in Signal and Image Analysis* (Cambridge University Press, New York, 2015)
23. M. Meyer, M. Desbrun, P. Schröder, A. Barr, Discrete differential-geometry operators for triangulated 2-manifolds. Vis. Math. III **3**(7), 35–57 (2003)
24. M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, Discrete Laplace operators: no free lunch, in *Proceedings of the Eurographics Symposium Geometry Processing* (2007), pp. 33–37
25. A. Crowley, J. Dong, A. McHaffie, A. Clarke, Q. Reeves, M. Williams, E. Robinson, N. Dalbeth, F. McQueen, Measuring bone erosion and edema in rheumatoid arthritis: a comparison of manual segmentation and ramris methods. J. Magn. Reson. Imaging **33**(2), 364–371 (2011)
26. M. Tocheri, C. Orr, S. Larson, T. Sutikna, E. Saptomo, R. Due, T. Djubiantono, M. Morwood, W. Jungers, The primitive wrist of homo floresiensis and its implications for hominin evolution. Science **317**(5845), 1743–1745 (2007)

Estimating the Complexity of the Cerebral Cortex Folding with a Local Shape Spectral Analysis



Hamed Rabiei, Frédéric Richard, Olivier Coulon and Julien Lefèvre

Abstract The human cerebral cortex is a highly folded structure that can be modeled by a surface extracted in vivo from magnetic resonance imaging data. The folding complexity of this surface has been shown to be a relevant biological measurement, often estimated locally and referred to by the term "gyrification index" (GI). There is, however, no universal agreement on the notion of surface complexity and various methods have been presented that evaluate different aspects of cortical folding. In this chapter, we show how a local spectral analysis of the cortical surface mean curvature can address this problem and provide two well-defined gyrification indices. Specifically, we extended the concept of graph windowed Fourier transform to the framework of surfaces modeled by triangular meshes. The intrinsic nature of the method allows us to compute the folding complexity at different spatial scales. We show that our approach overcomes a major flaw in other more classical GI estimators, namely the impossibility to differentiate deep cortical folds from shallower but more oscillating ones. We applied our method on synthetic data as well as on a database of 124 healthy adult subjects and showed that it verifies important properties and capture important aspects of cortical gyrification. A comparison with other GI definitions is also provided.

H. Rabiei · O. Coulon (✉) · J. Lefèvre
Institut de Neurosciences de la Timone, Aix-Marseille Université,
CNRS UMR7289 Marseille, France
e-mail: olivier.coulon@univ-amu.fr

H. Rabiei
e-mail: hamed.rabiei10@gmail.com

J. Lefèvre
e-mail: julien.lefevre@univ-amu.fr

H. Rabiei · F. Richard
Aix Marseille Univ, CNRS, Centrale Marseille, I2M, Marseille, France
e-mail: frederic.richard@univ-amu.fr

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_13

1 Cortical Folding Complexity and Gyrification Indices

The human cerebral cortex is a very convoluted sheet-like structure of variable thickness (1.5–4.5 mm) that shows a large number of folds, the cortical sulci, separated by concave regions called gyri. When observed with Magnetic Resonance Imaging (MRI) it is often approximated by its inner or outer surface, modelled with a triangular mesh of spherical topology (see Fig. 1). The developmental process that leads from a smooth surface at mid-gestation to this very convoluted surface after birth is called gyrification [1–4]. It is known now that the shape of the cortex can be a marker of normal or pathological development or aging [5–10]. In particular, quantifying the complexity of the cortical folds has been proven valuable and a number of methods have been proposed to estimate what is called a gyrification index (GI) either at the local (at every point) or the global (one value for the entire cortex) level. Such methods can be categorized in two classes: surface area (perimeter)-based methods [5–7, 11–14] and curvature-based methods [10, 15–17]. Methods in the first category compute the gyrification index as a ratio between the local area of the cortical surface and that of a reference surface. For instance, Toro et al. [5] defined a local GI as a ratio between the area of the surface contained in a spherical neighborhood of each cortical point and the area of the great disc of the sphere. Schaer et al. [6] proposed a ratio between the area of a region of interest, determined by intersection of a sphere with the convex hull of the cortical surface, and that of the corresponding patch on the surface as a local GI. Methods in the second category rely on the notion of curvature, in particular mean curvature, which assigns positive values to points on gyri and negative values to points on sulci [18]. Note that Gaussian curvature or shape index have also been used [16]. Although informative, curvature maps are too local to deliver a helpful insight into the surface folding [15], and smoothed version have been proposed to overcome this limitation [15].

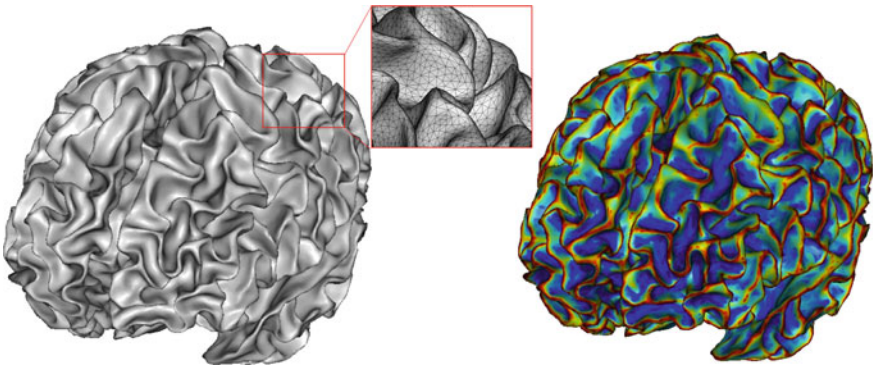


Fig. 1 Left: cortical surface modelled by a triangulated mesh. Right: mean curvature of the same surface

Although both categories of methods have been used in the neuroimaging literature, a number of strong limitations have been pointed out: difficulty to adapt to the brain size variability across the population [4]; impossibility to distinguish between deep folds and rapidly oscillating folds with equal surface areas which leads to high GI values for deep areas although they are not necessary showing complex folding (e.g. the insula or the central sulcus, see [5, 6]); difficulty to discriminate between normal and aberrant cortical development [10]; inconsistent (if not contradictory) results across methods [19]. If some of these limitations are intrinsic to the nature of the methods, one general problem that generates inconsistency and sometimes unwanted behavior is the lack of formal definition of what folding complexity is, and what properties we should expect from a quantitative measure of this complexity.

In this chapter, we propose a set of properties that we think are essential for a gyrification index and we demonstrate how spectral analysis of surfaces can be used to address the problem and define two different gyrification indices that comply to these properties. These GIs are computed directly on the cortical surface with neither of them requiring a reference surface nor a smoothing procedure. They intrinsically enable us to compute surface complexity at different spatial scales, and, in contrast to surface area-based GIs, they are able to disentangle the effect of depth on folding quantification. Finally, the issue of inconsistent analysis arising from the inter-subject brain size variability is also addressed by introducing an adaptive neighborhood. In the following, we present first a set of essential properties, then a local spectral analysis method based on the graph windowed Fourier transform of the mean curvature. This analysis is used to define two gyrification indices together with their properties. The method is then applied to some synthetic surfaces and real data in Sect. 4. Finally, we discuss the results and the specificities of our method, followed by a conclusion.

2 Essential Properties

As stated in the previous section, we propose here a number of properties that are essential to a gyrification index:

1. **Clear basis**: a GI should be defined based on a clear definition of the notion of "surface complexity" in order to be able to interpret the results produced with this GI.
2. **Physicality**: a GI should have an interpretable physical meaning. In other words, it should be proven that the proposed GI quantifies the definition of the surface complexity.
3. **Locality**: a GI should be defined locally.
4. **Multiscale**: in practice, it is an advantage to be able to quantify local gyrification in a wide range of spatial scales from few millimeters to several centimeters.

5. **Size adaptation**: gyrification quantification should be consistent across subjects. In particular it should take into account the large inter-subject brain size variability when defining the individual local scale of observation [7, 16].
6. **Geometric invariance**: a good GI should be invariant to translations, rotations and scaling, since these transformations do not change surface complexity.

In further sections we will demonstrate that the GIs that we define verify these properties.

3 Method

Spectral methods for surface analysis rely mainly on eigenvalues and/or eigenfunctions of an operator defined on the surface. For example, for a Riemannian manifold, the eigenfunctions of the Laplace–Beltrami operator serve as bases for Fourier transforms. The idea has been extended to triangular meshes modelling surfaces to analyze their structural properties; see [20] for a comprehensive survey on this topic. Here, we use such eigenfunctions to define a mesh windowed Fourier transform and provide an explicit access to the local frequency components of the mean curvature of the cortical surface that in turn are used to define two gyrification indices.

3.1 Mesh Fourier Transform

For a compact Riemannian manifold $\mathscr{S}$ as a surface in $\mathbb{R}^3$, one can consider the set of square integrable functions defined on the surface: $\mathscr{L}^2(\mathscr{S}) = \{u : \mathscr{S} \to \mathbb{R} | \int_{\mathscr{S}} u^2 < \infty\}$. The Laplace–Beltrami operator Δ, associated with the surface $\mathscr{S}$, is defined as a generalization to Riemannian manifolds of the Laplacian operator in Euclidean spaces. The eigenpair of this operator $\{(\lambda_i, \chi_i) \in \mathbb{R}^+ \times \mathscr{L}^2(\mathscr{S}), i \geq 0$ is generated by solving the differential eigenvalue problem of Δ: $\Delta\chi_i = -\lambda_i\chi_i$ in which λ_i and χ_i are called the ith eigenvalue and eigenfunction of Δ [21]. Spectral theory based on the Laplace–Beltrami spectrum can be used to obtain a new representation of the space $\mathscr{L}^2(\mathscr{S})$ in the spectral domain. In practice, the differential eigenvalue problem of Δ is usually solved on a triangulated surface by numerical methods. Formally, let $G = \{V, E\}$ be a triangular mesh modelling the surface $\mathscr{S}$ where V is the set of vertices, $V = \{P_1, P_2, ..., P_N\}$ and E is the set of edges.

By using the linear finite element method (FEM) [22], the above-mentioned differential eigenvalue problem is discretized to the following algebraic generalized eigenvalue problem

$$A\chi = \lambda B\chi, \tag{1}$$

where A and B are $N \times N$ sparse matrices with the following elements:

$$A(i,j) = \begin{cases} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} & \text{if } (i,j) \in E, \\ -\sum_{k \in \mathcal{N}(i)} A(i,k) & \text{if } i = j, \\ 0 & \text{o.w.} \end{cases} \tag{2}$$

and

$$B(i,j) = \begin{cases} \frac{|t| + |t'|}{2} & \text{if } (i,j) \in E, \\ \frac{\sum_{k \in \mathcal{N}(i)} |t_k|}{6} & \text{if } i = j, \\ 0 & \text{o.w.} \end{cases} \tag{3}$$

where α_{ij} and β_{ij} are the angles opposite to the edge $P_i P_j$ in two triangles t and t' sharing this edge, $|t_k|$ indicates the area of the triangle t_k and $\mathcal{N}(i)$ denotes the index set of all vertices of the 1-ring neighbourhood of P_i. The matrix B is positive definite and defines the so called B-inner product in $\mathbb{R}^N$:

$$\forall f, g \in \mathbb{R}^N, \quad \langle f, g \rangle_B = f^t B g. \tag{4}$$

Solutions of the discrete eigenvalue problem (1) are nonnegative real eigenvalues $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$ and a set of eigenvectors $\{\chi_j, j = 1, 2, \ldots, N\}$ in $\mathbb{R}^N$ which are orthonormal with respect to B-inner product i.e. $\langle \chi_i, \chi_j \rangle_B = \delta_{ij}$ where δ_{ij} is Kronecker delta.

The bases of the Fourier transform on a segment or a rectangle are complex exponential functions which are the eigenfunctions of the Laplace–Beltrami operator. Inspired by this fact, the eigenvectors of the discretized Laplace–Beltrami operator serve as Fourier atoms on the triangulation [23]. Given a function f defined on the vertices of triangulation, Fourier transform coefficients of f are given by the set $\{\hat{f}(l) := \langle f, \chi_l \rangle_B, l = 1, 2, \ldots, N\}$. This set is a representation of function f in the spectral domain and gives the frequency distribution of this function. In this setting, the Parseval's identity is

$$\langle f, g \rangle_B = \langle \hat{f}, \hat{g} \rangle, \tag{5}$$

where $\langle ., . \rangle$ denotes the Euclidean inner product. It yields $\| f \|_B = \| \hat{f} \|_2$ where $\| . \|_B$ is the norm induced by the B-inner product.

3.2 A Local Mesh Fourier Transform

The mesh Fourier transform described in the previous section gives a global frequency distribution of the function f. In other words, it is not able to provide information about the frequencies of f in a local spatial neighborhood. In the continuous domain, the windowed Fourier transform was introduced in order to find the local frequency distribution of a function [24]. This transform provides local information about a

function simultaneously in the spatial and frequency domains. Shuman et al. [25] have recently extended this transform to the graph settings. The general idea of this method is to localize a function defined on the vertices of a graph around a vertex by a translated window function and then, compute the graph Fourier transform of this localized function. In this paper, we extend the method to the mesh framework by using a mesh Fourier transform that takes into account the mesh geometry. Since a mesh is a graph of a specific type, one may argue that the graph spectral theory tools can be applied on a mesh without any adaptation. In general, it is true but unlike the graph Laplacian in which only the connectivity of vertices is considered, the geometric (FEM) Laplacian takes into account geometric properties of the surface. Equations (2) and (3) show how local geometry of neighboring triangles on the mesh contributes to the definition of Laplacian operator. More discussions and comparisons between graph Laplacian and geometric Laplacian can be found in [20, 22, 23, 26–28].

3.2.1 Window Function

Let $f : V \to \mathbb{R}$ be a function defined on the vertices of a triangulation (e.g. the mean curvature). To localize this function around a specific vertex, we need a window function with local support and a translation operator to move the window function to that specific vertex. Following [25], we consider the window function

$$\hat{g}(l) = C \exp(-\tau \lambda_l), \tag{6}$$

defined in the spectral domain. In this formula, τ is a parameter which determines the size of the window, λ_l is the lth Laplace–Beltrami eigenvalue and C is chosen such that $\|\hat{g}\|_2 = 1$. The window size parameter τ sets a locality tradeoff between the frequency and spatial domains [25, 29]. By increasing τ, we get a wider window in the spatial domain and the function f is localized in a larger neighborhood around each vertex, while getting a more local frequency distribution of the function in that neighborhood. The spread of the window function in spatial and frequency domains is measured by the area of the Heisenberg box ([30] Section 4.2), ([25], Section 6.6). It is proved that the Gaussian function is the unique window that minimizes the area of the Heisenberg box ([30] Theorem 2.5). Since λ_l is proportional to the square of the spatial frequency [31, 32], i.e. $\lambda_l \propto \omega_l^2$, the window function (6) corresponds to a Gaussian function in the frequency domain $\hat{g}(l) \propto \exp(-\tau \omega_l^2)$.

3.2.2 Translation Operator

The fact that there is no canonical origin and direction on a triangulated mesh makes it difficult to define a translation operator on this mesh. Inspired by the properties of the generalized Fourier transform in continuous domain and the mechanism proposed in [25] for graph settings, we propose to define a translation operator as follows.

Let $\{\psi_l, l = 1, 2, \ldots\}$ be the basis of the generalized Fourier transform in continuous domain and assume that those functions are orthonormal with respect to an inner product depending on a function ϕ i.e. $\int \psi_l \phi \psi_k = \delta_{kl}$. From the properties of the generalized Fourier transform, translation in spatial domain causes modulation in Fourier domain:

$$h(x) = f(x - x_0) \quad \Leftrightarrow \quad \hat{h}(k) = \psi_k(x_0)\phi(x_0)\hat{f}(k). \tag{7}$$

In other words, the translation of function f to point x_0 can be given by the inverse Fourier transform of the modulated Fourier coefficients $\hat{f}$:

$$(T_{x_0} f)(x) = \mathscr{F}^{-1}\{\psi_k(x_0)\phi(x_0)\hat{f}(k)\},$$

where $\mathscr{F}^{-1}$ denotes the inverse Fourier transform.

In mesh settings, the Fourier basis is the set of Laplace–Beltrami eigenvectors which are orthonormal with respect to the matrix B. If g is a function defined on the vertices of a mesh, inspired by Eq. (8), the translation of g to vertex P_i is defined as follows :

$$\begin{aligned}(T_i g)(n) &= \sqrt{N}\mathscr{F}^{-1}\{B(i,:)\chi_l \hat{g}(l)\} \\ &= \sqrt{N}\sum_{l=1}^{N}\sum_{m=1}^{N}\chi_l(n)\Big(B(i,m)\chi_l(m)\hat{g}(l)\Big),\end{aligned}$$

where $B(i,:)$ denotes the ith row of B. The translation operator T_i shifts the center of the window function to vertex P_i. By multiplying the function f by the translated window function $T_i g$, it is localized around the vertex P_i:

$$\tilde{f}_i(n) = (T_i g)(n) f(n), \quad n = 1, 2, \ldots, N. \tag{8}$$

3.2.3 Mesh Windowed Fourier Transform

The mesh windowed Fourier transform coefficients of a function $f \in \mathbb{R}^N$ are defined as the modulation of the localized function $\tilde{f}_i$ by Fourier atoms $\{\chi_k, k = 1, 2, \ldots, N\}$:

$$S_f(i,k) := \langle \tilde{f}_i, \chi_k \rangle_B, \tag{9}$$

where $i = 1, 2, \ldots, N$ is the index of vertex and k is the index of frequency. This gives us a frequency spectrum $\{|S_f(i,k)|^2,\ k = 1, 2, \ldots, N\}$ for every vertex P_i of the mesh which can be seen as the frequency distribution of the function f in a local neighborhood around the vertex P_i.

An example of such frequency spectrum is shown on Fig. 2 (top) for 4 different vertices. On Fig. 2 (bottom right), a global representation of the frequency spectrum

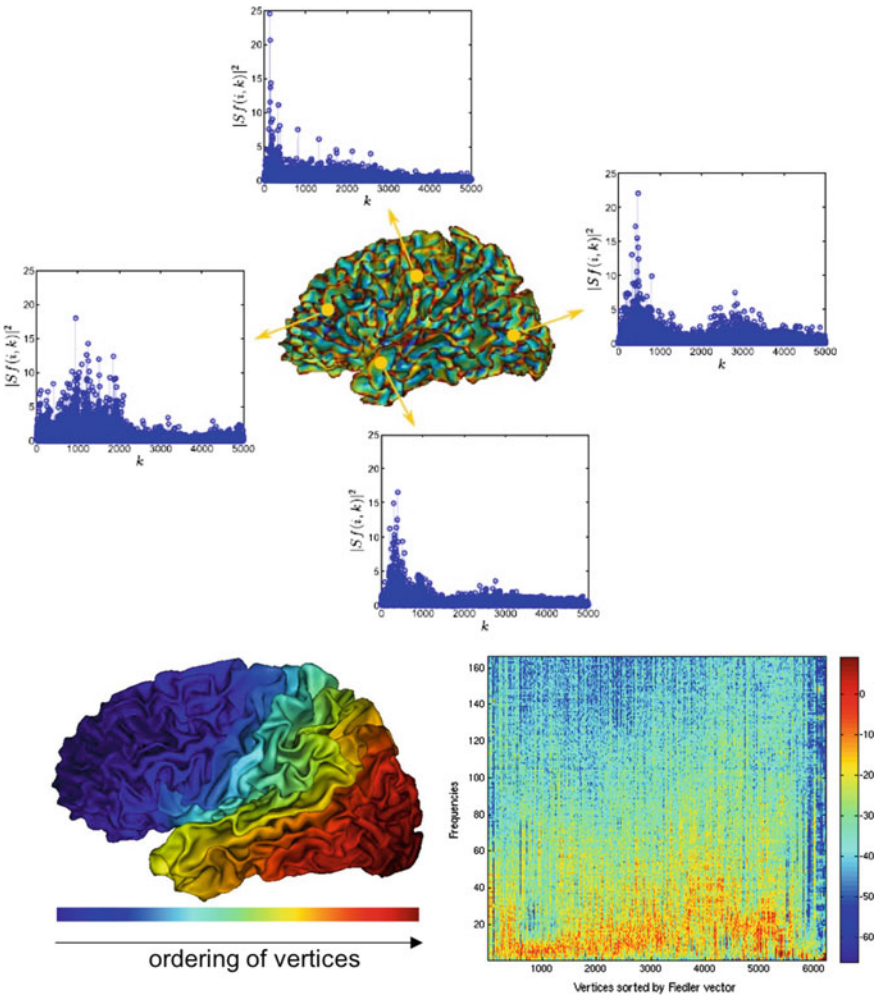


Fig. 2 Top: cortical surface with the local spectrum at 4 different points. Bottom right: color coded spectrum at all vertices. Vertices are ordered along an antero-posterior axis as shown on bottom left

at all vertices is shown (vertices on the x-axis and frequency bands on the y-axis). Vertices are ordered along a antero-posterior axis that is computed using the Fiedler vector, i.e. the second eigenfunction of the Laplace–Beltrami operator, shown on Fig. 2 (bottom left). As can be seen, the distribution of frequencies varies a lot across vertices, showing a variable degree of folding complexity on the cortical surface.

3.2.4 Adaptive Window Function

Let q be a real positive number. It is easily seen from Eqs. (1)–(3) that if the surface area is scaled by a factor q, the eigenvalues are scaled by $1/q$. In this case, due to the definition of the window function (6), the relative spread of the window function (i.e. the ratio between the area of the region covered by the window and the total surface area) is affected by the size of the surface. In other words, given a fixed window size τ for a surface and scaled versions of this surface, the window function covers a relatively larger area on a smaller version of the surface and vice versa. This leads to an inconsistent large and small scale spectral analysis for small and large surfaces, respectively. To keep the relative spread of the window function constant across surfaces, we introduce an adaptive window function in which the total surface area is incorporated:

$$\hat{g}(l) = C \exp(-\tau |\mathscr{S}| \lambda_l), \tag{10}$$

where $|\mathscr{S}|$ denotes the total area of surface $\mathscr{S}$. Note that with this definition, a dimensionless parameter is introduced inside the exponential. As we will see in Sect. 3.3, this adaptive window function also plays an important role to derive scale invariant gyrification indices.

In Fig. 3, the spread of the adaptive window function (10) with 3 different window size parameters, $\tau = 2e-4$, $1e-3$ and $5e-3$ is shown on a cortical surface around a specific vertex (yellow point). As τ increases, the spread of window in spatial domain increases as well. While the narrow window, $\tau = 2e-4$, covers a part of a gyrus and/or sulcus, the medium window, $\tau = 1e-3$, covers several folds and the wide window, $\tau = 5e-3$, covers a big portion of the cortical surface equivalent to a lobe. Tuning the window size parameter τ changes the spatial scale of the analysis.

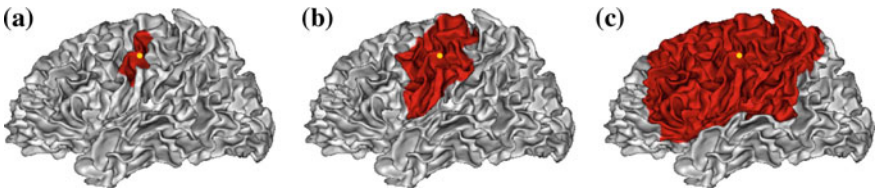


Fig. 3 Spread of the adaptive window function (Eq. (10)) with 3 different window sizes around the yellow vertex on a cortical surface, after thresholding the values of the window functions at a constant percentage of their maximal values. The red color highlights the window spread around the vertex. **a** A narrow window, with $\tau = 2e-4$, covers about 1.5% of the surface. **b** A medium window, with $\tau = 1e-3$, covers about 8% of the surface. **c** A wide window, with $\tau = 5e-3$, covers about 36% of the surface

3.3 Gyrification Indices

As mentioned above, it is important to propose an explicit interpretation of the notion of surface complexity, essential to define a gyrification index. We propose here two interpretations that rely on surface bending properties : in a neighborhood around each point of the cortical surface mesh, surface complexity is quantified by:

1. The magnitude of the surface bending
2. The spatial variations of the surface bending.

3.3.1 Definitions

A natural proxy for surface bending is the mean curvature function (that has actually been used already to define gyrification indices, see the introduction of this chapter). Let us therefore define the function f at each vertex of the cortical surface mesh as the mean curvature of the surface at this vertex. By applying the mesh windowed Fourier transform to this function, we get a frequency spectrum at each vertex P_i of the mesh which consists of the frequency powers $|S_f(i,k)|^2,\ k = 1, 2, \ldots, N$. The summation of the frequency powers is called the total power (TP) of the frequency spectrum [30]. We propose the first GI definition as follows:

Definition 1 Spectral Gyrification Index (sGI)

$$\mathrm{sGI}(i, \mathscr{S}) = \sum_{k=1}^{N} |S_f(i,k)|^2 \tag{11}$$

With i the index of the vertex.

Since fast spatial variations of a function are encoded in the high frequency band of its frequency spectrum, by giving a larger weight to higher frequency powers we can comply to the second interpretation of surface complexity presented above, which leads to the definition of a second gyrification index :

Definition 2 Weighted Spectral Gyrification Index (wGI)

$$\mathrm{wGI}(i, \mathscr{S}) = \sum_{k=1}^{N} \left(\frac{\lambda_k}{\lambda_2}\right)^2 |S_f(i,k)|^2 \tag{12}$$

In this definition, weights are the normalized eigenvalues of the Laplace–Beltrami operator, that contain information about the shape of the surface [33, 34]. The normalization by the first nonzero eigenvalue λ_2 removes the effect of the surface size on the weighting [34].

3.3.2 Verification of Essential Properties

To verify the 6 essential properties presented in Sect. 2, we need the following mathematical results

Proposition 1 *For gyrification index sGI at vertex i of subject $\mathscr{S}$ we have*

$$sGI(i, \mathscr{S}) = \|\tilde{f}_i\|_B^2$$

Proof

$$\sum_{k=1}^{N} |S_f(i,k)|^2 = \sum_{k=1}^{N} |\langle \widehat{\tilde{f}_i}, \widehat{\chi_k} \rangle|^2 \tag{13}$$

$$= \sum_{k=1}^{N} |\widehat{\tilde{f}_i}(k)|^2 \tag{14}$$

$$= \|\tilde{f}_i\|_B^2, \tag{15}$$

where (13) and (15) are derived by Parseval's identity (5) and (14) is based on the fact that $\widehat{\chi_k} = \delta_k$ (δ_k is Kronecker delta).

Lemma 1 *Let $f \in \mathbb{R}^N$ be a function defined on the vertices of a triangulation and $L \in \mathbb{R}^{N \times N}$ be the discrete Laplace–Beltrami operator i.e. $L = B^{-1}A$. Then, the Fourier coefficients of the function $y = Lf$ are $\hat{y}(l) = \lambda_l \hat{f}(l),\ l = 1, 2, \ldots, N$.*

Proof The Fourier coefficients of y are as

$$\hat{y}(l) := \langle Lf, \chi_l \rangle_B = f^t (B^{-1}A)^t B \chi_l$$

$$= f^t \lambda_l B \chi_l \tag{16}$$

$$= \lambda_l \langle f, \chi_l \rangle_B \tag{17}$$

$$= \lambda_l \hat{f}(l),$$

where Eq. (16) is given by Eq. (1) and the symmetry of matrices A and B, and the definition of B-inner product (4) gives (17).

Proposition 2 *For gyrification index wGI at vertex P_i of subject $\mathscr{S}$ we have*

$$wGI(i, \mathscr{S}) = \frac{1}{\lambda_2^2} \|L\tilde{f}_i\|_B^2$$

Proof Lemma 1 and the Parseval's identity give the first and the second equality, respectively:

$$\sum_{k=1}^{N} \left(\frac{\lambda_k}{\lambda_2}\right)^2 S_f(i,k)^2 = \frac{1}{\lambda_2^2} \| \widehat{L\tilde{f}_i} \|_2^2 = \frac{1}{\lambda_2^2} \|L\tilde{f}_i\|_B^2.$$

Proposition 3 *The gyrification indices sGI and wGI are scale invariant.*

Proof Assume that the surface $\mathscr{S}_2$ is the scaled version of the surface $\mathscr{S}_1$ by a factor q^2 i.e. $|\mathscr{S}_2| = q^2|\mathscr{S}_1|$. Then the Laplace–Beltrami eigenvalues are scaled by $1/q^2$ while the eigenvectors and the mean curvature are scaled by $1/q$.

Thanks to the adaptive window function (10), the translation operator and the mesh windowed Fourier coefficients $S_f(i,k)$ remain unchanged because

$$\begin{aligned} S_f(i,k)_{\mathscr{S}_2} &= \langle \tilde{f}_{i,\mathscr{S}_2}, \chi_{k,\mathscr{S}_2} \rangle_{B_{\mathscr{S}_2}} = q^2 \langle \frac{1}{q} \tilde{f}_{i,\mathscr{S}_1}, \frac{1}{q} \chi_{k,\mathscr{S}_1} \rangle_{B_{\mathscr{S}_1}} \\ &= S_f(i,k)_{\mathscr{S}_1}. \end{aligned}$$

It implies that sGI and wGI remain unchanged under scaling.

The two propositions make the two gyrification indices coherent with the two interpretations of surface complexity:

1. *Magnitude of the surface bending*
   Proposition 1 reflects the relationship between sGI and the magnitude of the mean curvature weighted by the window function around a vertex.
2. *Spatial variations of the surface bending*
   By nature, the Laplace–Beltrami operator estimates the local variation of a function. Proposition 2 links wGI and the local variations of the localized mean curvature $(L\tilde{f}_i)(m)$ at vertex i.

As for the properties defined in Sect. 2, for both sGI and wGI, we have provided a clear interpretation of surface complexity (property 1, clear basis), and a formal definition of a GI that is demonstrated to quantify such interpretation (property 2, physicality). Both GIs are defined locally (property 3, locality). The adaptive window function presented in Sect. 3.2.4 provides both a multiscale measure (property 4) and an independence from the overall surface size (property 5, size adaptation). Finally, the Laplace–Beltrami spectrum is invariant under isometric transformations, which makes sGI and wGI isometry invariant. Proposition 3 guarantees the scale invariance. All the requirements of property 6 (geometric invariance) are therefore satisfied.

4 Experiments and Results

In this section we demonstrate the efficiency of sGI and wGI using a set of experiments both on synthetic data, in order to control explicitly the folding of the surface, and real data. Our results will be compared to those produced using Toro's GI [5], as it is representative of the most popular class of GIs used in the neuroimaging community.

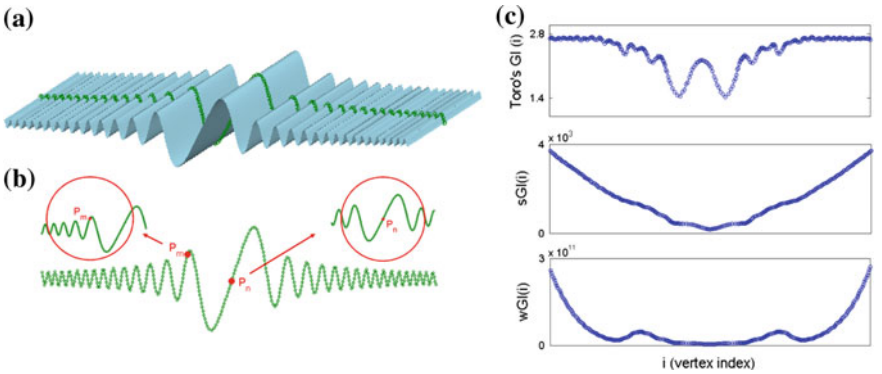


Fig. 4 **a**: synthetic wavy rectangle; **b**: middle line and two specific vertices; **c**: different GI plots along the middle line: Toro's GI (top), sGI (middle), wGI (bottom)

Table 1 Different GIs for vertices P_m and P_n of the synthetic surface depicted in Fig. 4

Vertex	sGI	wGI	Toro's GI
P_m	8.56×10^2	2.81×10^{10}	2.23
P_n	1.85×10^2	3.88×10^9	2.23

4.1 Synthetic Data

We generated a surface on a rectangular domain with controlled oscillations (wavy rectangle). It is created by the equation $z = 2\sin(60\pi x^2)/(60\pi x)$ where $-0.7 \leq x \leq 0.7$ and $0 \leq y \leq 1$ with a geodesic length equals to 4. This surface is triangulated with $N = 40,000$ equidistant vertices and is shown in Fig. 4a. The intersection of this surface with the plane $y = 0.5$ is indicated on the surface by green points and is plotted in Fig. 4b. As we move away from the center towards the left or right side of this surface, it becomes more folded i.e. the surface becomes more bended and the spatial frequency of the folds increases. The windowed Fourier transform is applied on the mean curvature of this surface and the spectrogram computed. In Fig. 4b, we focus on two vertices P_m and P_n: the surface is more folded around P_m than P_n, while the fold around P_n is deeper. The values at P_m and P_n of sGI, wGI and the surface area-based GI defined by Toro et al. [5] are given in Table 1. While sGI and wGI return appropriately higher values for P_m, Toro's GI returns equal values for both vertices, showing that the surface area-based GI is not able to discriminate between deep folds and complex folds with the same area.

In Fig. 4c, we plotted the values of the three GIs (sGI, wGI and Toro's GI) along the green middle line shown in Fig. 4a,b. Again, it is visible here that sGI and wGI provide increasing values when moving from the center towards the right and left extremities, while Toro's GI shows a peak in the center (due to the increased folding depth) and flat curves towards the left and right extremities, as depth decreases and

folding increases. This again illustrates the ability of sGI and wGI to differentiate the effect of folding and depth, as opposed to surface-based methods such as Toro's GI.

4.2 Real Data

In this section, we applied our method to a database of real cortical surfaces. We computed spatial maps of sGI, wGI, and Toro's GI for each individual as well as an average group map across all subjects. We present these maps to illustrate and quantify the properties of each method, and we also study the ability of each method to capture known neuroscientific facts such as the relationship between folding and total brain volume.

4.2.1 Data and Preprocessing

We applied the method to 124 healthy adult subjects from the Open Access Series of Imaging Studies (OASIS) database (http://www.oasis-brains.org). For each subject, three or four T1 anatomical Magnetic Resonance Images (MRIs) had been acquired at in-plane resolution of 1mm $\times$ 1mm, slice thickness = 1.25 mm, TR = 9.7 ms, TE = 4 ms, flip angle = 10u, TI = 20 ms, TD = 200 ms. Images of each subject were motion corrected and averaged to create a single image per subject with a high contrast-to-noise ratio. The resulting anatomical MR images were processed using the BrainVISA neuroimaging software platform (http://brainvisa.info), providing a segmentation of tissues, as well the cortical surface of each hemisphere modelled with a triangular mesh of spherical topology with around 50,000 nodes, depending on the subjects. The Hip-Hop algorithm [35] in BrainVISA was then applied to compute interindividual correspondence between cortical surfaces, thus allowing the averaging of any information across subjects.

4.2.2 Gyrification Maps

sGI and wGI maps of a subject in the database are shown in Fig. 5 for 3 different window sizes, $\tau = 2e-4,\ 1e-3,\ 5e-3$, associated to spatially local, medium and wide windows, respectively. On this figure, it is visible that the window size parameter τ controls the scale of observations. At $\tau = 2e-4$, the spatial scale is fine and high values are located mostly on the ridge of complex gyri, while low values are located on the walls of regular sulci. As the window size increases, a more regional effect becomes visible, with a very smooth and low variation map at value $\tau = 5e-3$, which gives a coarse scale observation of the gyrification.

Group average maps of both GIs at a medium scale (window size $\tau = 1e-3$) have been computed by using the Hip-Hop [35] cortical surface inter-subject matching

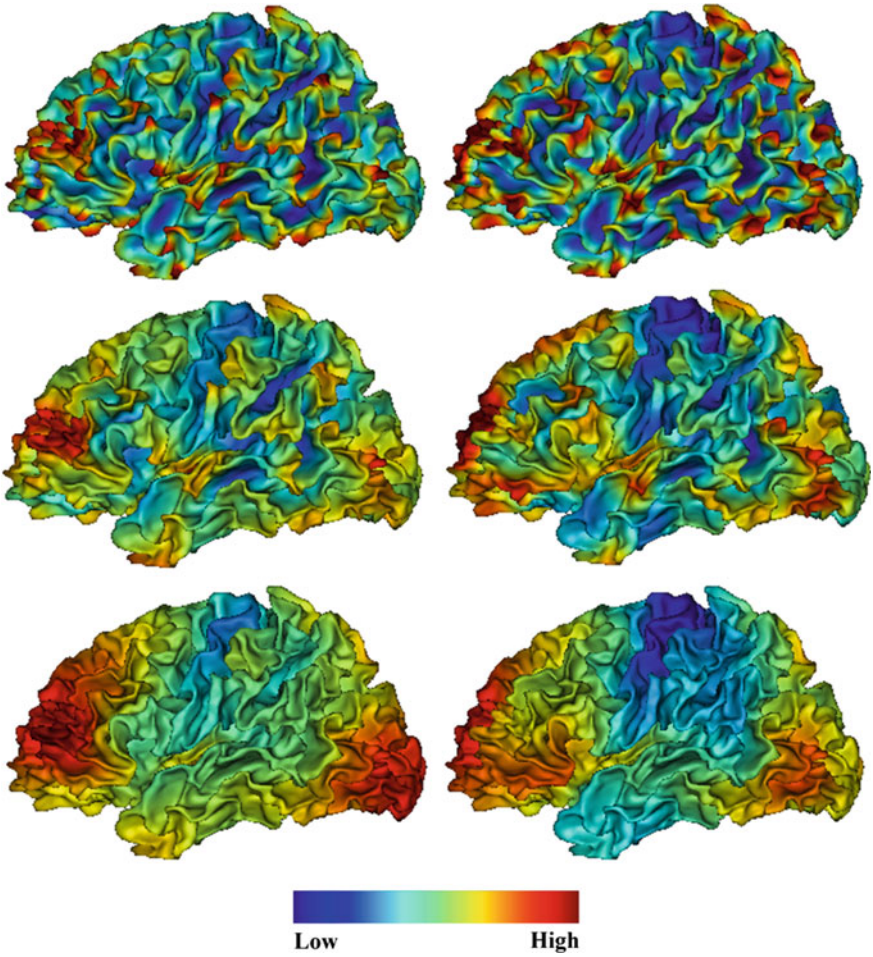


Fig. 5 Gyrification maps, (left) sGI and (right) wGI, of the left hemisphere of an individual subject from our database at 3 different scales. In the first row $\tau = 2e-4$, in the second row $\tau = 1e-3$ and in the third row $\tau = 5e-3$

method. Results are shown on the template cortical surface hiphop138 (http://mecabrain.org/software/hiphop138/) in Fig. 6 for the left hemisphere. The average patterns of sGI and wGI represented in this figure are similar to those observable for the individual subject on Fig. 5, which shows that the spatial patterns of the proposed GIs are reproducible across subjects. On top row, sGI shows higher values in the prefrontal and occipital lobes, inferior parietal lobe, inferior temporal sulcus and the medial area of the superior parietal cortex. On the middle row, wGI shows high values in the prefrontal lobe, medial part of the occipital lobe and the posterior cingulate gyrus. We also computed the average GIs of the right hemispheres and by visual

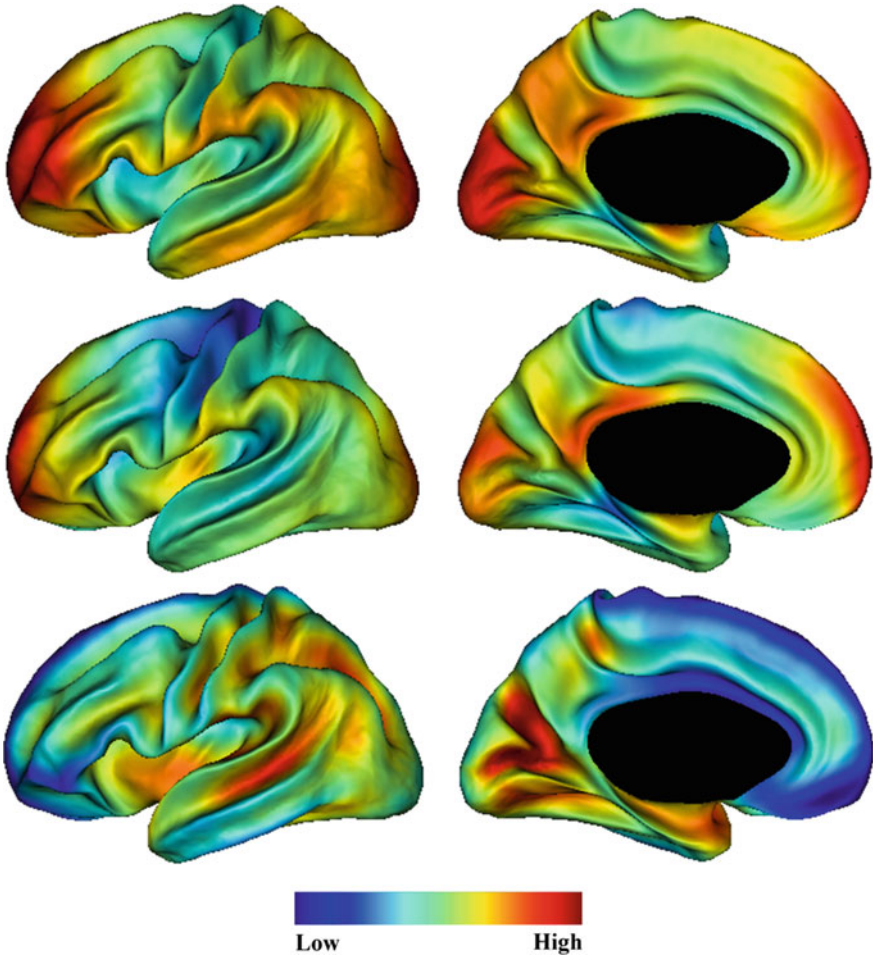


Fig. 6 Group average GI maps on the HipHop138 template surface. Top row: sGI ($\tau = 1e-3$); middle row: wGI ($\tau = 1e-3$); bottom row: Toro's GI (radius $r = 20$)

inspection, we observed no remarkable difference in gyrification patterns of the left and right hemispheres in the medium scale. For comparison, the average map of Toro's GI across all subjects is also presented on bottom row, and shows high GI values in deep folds like the central sulcus, the insula, the superior temporal sulcus, the calcarine fissure, the parieto- occipital sulcus, and the intraparietal sulcus, again showing the direct effects of depth on the gyrification estimation.

As it has been discussed in Sect. 3.3, sGI and wGI, by construction, measure complementary properties of surface folding. This is shown in Fig. 7 where a cortical surface is shown with its mean curvature. Two regions on this surface, R1 and R2, have been chosen. R1 is a sharp spike located on the postcentral gyrus and R2 is a

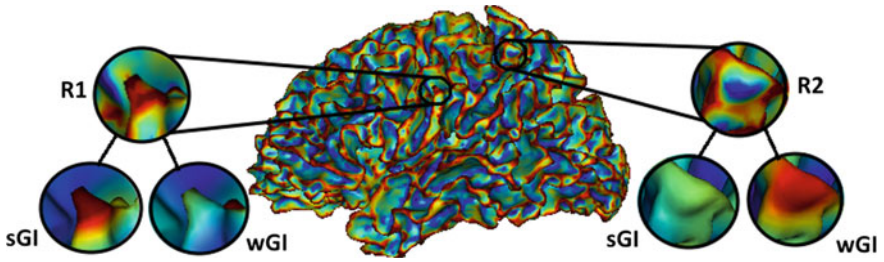


Fig. 7 Zoom on two cortical regions R1 and R2. Colormap encodes the mean curvature (from blue, negative values, to red, positive values). sGI and wGI ($\tau = 2e-4$) values are shown for R1 and R2

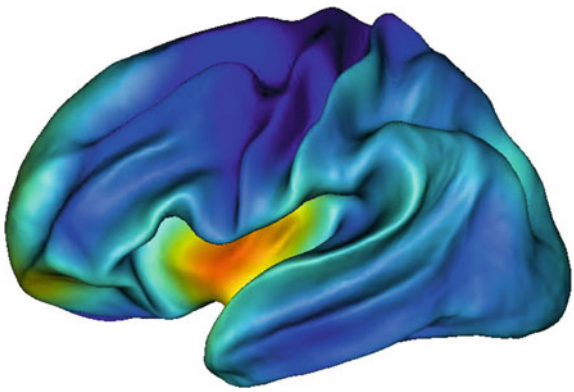


Fig. 8 Group average of the rGI ratio map displayed on the HipHop138 template surface ($\tau = 1e-3$)

very shallow fold located on the superior parietal lobe. The mean curvature of the region R1 is very high while that of the region R2 varies a lot between positive and negative values. The maps of sGI and wGI of R1 and R2 are shown in this figure ($\tau = 2e-4$). As expected theoretically from Eqs. (12) and (14), sGI gives high value to R1 while wGI assigns high value to R2.

Although not easy to interpret, the relationship between sGI and wGI is of interest and contains information. For instance, by computing the ratio rGI of wGI and sGI we get an estimate of the dominant frequency band:

$$\mathrm{rGI}(i, \mathscr{S}) := \mathrm{wGI}(i, \mathscr{S})/\mathrm{sGI}(i, \mathscr{S}) \tag{18}$$

The group-average rGI map is shown ($\tau = 1e-3$) in Fig. 8. As visible, the highest values are located in the insula, a structure with low curvature, a low sGI, and a medium wGI (see Fig. 8). These high values are due to the fact that the insula contains several small shallow folds on a rather small flat area, leading to high frequency oscillations of the surface.

4.2.3 Relationship Between Gyrification and Volume Analysis

Recent studies show that larger brains are more folded [5, 9, 36, 37]. To investigate this phenomenon, the global sGI and wGI (defined as the integration across all vertices of the local GIs) of each hemisphere are modelled by the following power law:

$$G = kV^{\alpha} \tag{19}$$

where G denotes the global gyrification index (sGI or wGI) of an hemisphere, V is the hemispheric volume and k and α are coefficients to be determined. Since gyrification indices sGI and wGI are scale invariant (property 6), the scaling exponent coefficient α of the power law (20) should be 0 under an isometric scaling of brain volume. We present in Fig. 9 the relationship between volume and GI with a fitted power law in logarithmic scale. The positive exponent coefficient reveals a positive allometric scaling of gyrification indices with volume and shows that we can capture and quantify the fact that larger brains are intrinsically more folded than smaller ones. Values of the exponent coefficient α is higher for wGI than for sGI ($0.67 > 0.37$ and $0.66 > 0.36$ for left and right hemispheres, respectively) while the proportion of the variance of sGI explained by the volume is higher than that of wGI ($0.44 > 0.36$ and $0.47 > 0.34$ for left and right hemispheres, respectively). Results also suggest that in the global hemispheric scale, the degree of folding of the left and right hemispheres increase with volume symmetrically. This experiment demonstrates the ability of our GIs to capture and quantify a biological phenomenon such as the allometric relationship between folding and volume.

5 Discussion and Conclusion

In this chapter, we proposed two clear interpretations of the notion of cortical surface complexity, and two definitions of gyrification indices that are consistent with these interpretations, based on a local spectral analysis of the surface mean curvature. The GIs were also shown to comply to essential properties defined beforehand. The methods were applied to synthetic surfaces and to a database of 124 healthy adult brains and results illustrate their very good performances. Both GIs describe a consistent pattern of gyral complexity at the individual and group levels, in which the prefrontal and occipital lobes appear to be the most gentrified areas, while deep primary folds such as the central sulcus, the superior temporal sulcus, or the insula have a lower gyrification. We also demonstrated that the GIs that we proposed are able to capture known biological phenomenon such as the allometric relationship between brain volume and folding complexity. An important feature of both GIs is the fact that they disentangle the effects of depth and geometric oscillation. Surface area-based methods such as Toro's [5] or Schaer's [6] estimate the largest gyrification values within very deep folds such as the central sulcus, the insula, the intraparietal sulcus, or the superior temporal sulcus (see Fig. 6, or ([5], Fig. 5a), or ([6], Fig. 4)),

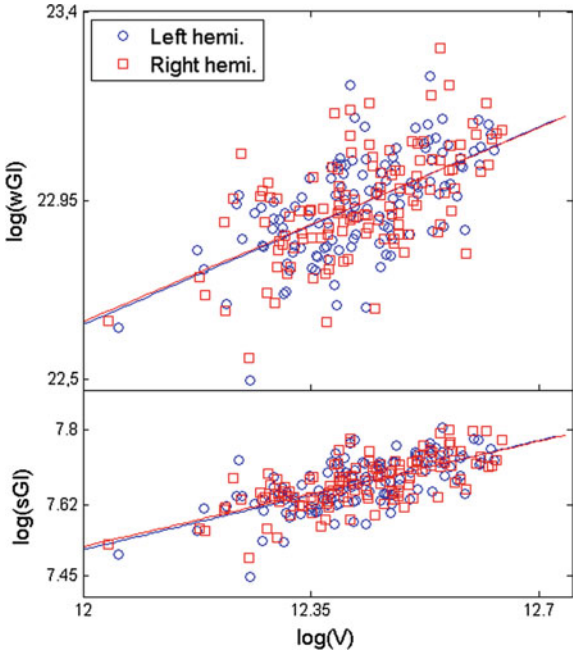


Fig. 9 Log-log relationship between brain volume and global gyrification index for wGI (top) and sGI (bottom)

because of the large depth of these folds. This in particular was evidenced with experiments on synthetic surfaces.

Another difference between our method and surface area-based methods is that the latter may be not localized enough for some applications. For example, in Fig. 4 of [6], for a small spherical neighborhood, the most folded region of the cortex is around the Sylvian Fissure and as the size of the neighborhood increases, the same pattern propagates across the cortex. Therefore, it may fail to catch other locally highly folded parts of the brain, thus affecting the reliability of findings. Our method, by tuning the neighborhood size, provides results at different spatial scales, ranging from a very local scale (in the order of a part of a sulcus/gyrus) to a more global scale (in the order of a lobar cortex); see Figs. 3 and 5.

Despite the differences we highlighted between our method and previous GI definitions, we do think that these previous gyrification indices are still valid but should be considered in the light of their specific interpretations of the notion of surface complexity. Our experiments on synthetic surfaces show how different interpretations of this notion may lead to different results. Accordingly, a message of this chapter is to point out that one should pay attention to this notion and interpret results based on it, and that spectral analysis can provide a well-defined formal measure of such notion. Another work on this topic has been done by Awate et al. [19] discussing about the different meanings of surface complexity and categorizing GIs based on their responses to different situations like surface scaling and increased spatial frequency of folds.

Finally, an important matter that has been mostly neglected so far is the effect of brain size on the relative neighborhood spread. Due to inter-subject size variability, analysis of cerebral cortices with a fixed neighborhood size may cause inconsistency. To address this issue, we introduced an adaptive neighborhood size that accounts for total surface area variability. We demonstrated in this chapter how graph signal processing tools can be adapted to triangular mesh representations of shapes in order to extract and quantify relevant information on these shapes. In particular we hope to have convinced that a local shape spectral analysis can be defined in order to advance our understanding of a biological phenomenon such as the folding of the cerebral cortex, and that the tools defined in this chapter can be used for the understanding and characterization of shapes in general.

Acknowledgements The work was supported by Labex Archimède (http://labex-archimede.univ-amu.fr/). The authors would like to thank the OASIS project for providing and sharing MRI databases of the brain. The OASIS project was supported by NIH grants P50 AG05681, P01 AG03991, R01 AG021910, P50 MH071616, U24 RR021382, and R01MH56584.

References

1. E. Armstrong, A. Schleicher, H. Omran, M. Curtis, K. Zilles, The ontogeny of human gyrification. Cereb. Cortex **5**(1), 56–63 (1995)
2. W. Welker, Why does cerebral cortex fissure and fold?, in *Cerebral Cortex*, ed. by E.G. Jones, A. Peters, no. 8B in Cerebral Cortex (Springer, Berlin, 1990), pp. 3–136
3. P. Yu, P.E. Grant, Y. Qi, X. Han, F. Segonne, R. Pienaar, E. Busa, J. Pacheco, N. Makris, R.L. Buckner, P. Golland, B. Fischl, Cortical surface shape analysis based on spherical wavelets. IEEE Trans. Med. Imaging **26**(4), 582–597 (2007)
4. J. Lefèvre, D. Germanaud, J. Dubois, F. Rousseau, I. de Macedo Santos, H. Angleys, J.-F. Mangin, P. Höppi, N. Girard, F. De Guio, Are developmental trajectories of cortical folding comparable between cross-sectional datasets of fetuses and preterm newborns? Cereb. Cortex, 1–13 (2015)
5. R. Toro, M. Perron, B. Pike, L. Richer, S. Veillette, Z. Pausova, T. Paus, Brain size and folding of the human cerebral cortex. Cereb. Cortex **18**(10), 2352–2357 (2008)
6. M. Schaer, M.B. Cuadra, L. Tamarit, F. Lazeyras, S. Eliez, J. Thiran, A surface-based approach to quantify local cortical gyrification. IEEE Trans. Med. Imaging **27**, 161–170 (2008)
7. G. Li, L. Wang, F. Shi, A.E. Lyall, W. Lin, J.H. Gilmore, D. Shen, Mapping longitudinal development of local cortical gyrification in infants from birth to 2 years of age. J. Neurosci. **34**, 4228–4238 (2014)
8. T. White, C.C. Hilgetag, Gyrification and neural connectivity in schizophrenia. Dev. Psychopathol. **23**(1), 339–352 (2011)
9. D. Germanaud, J. Lefèvre, C. Fischer, M. Bintner, A. Curie, V. des Portes S. Eliez, M. Elmaleh-Bergès, D. Lamblin, S. Passemard, G. Operto, M. Schaer, A. Verloes, R. Toro, J.F. Mangin, L. Hertz-Pannier, Simplified gyral pattern in severe developmental microcephalies? new insights from allometric modeling for spatial and spectral analysis of gyrification. NeuroImage **102**, Part 2, 317–331 (2014)
10. J.S. Shimony, C.D. Smyser, G. Wideman, D. Alexopoulos, J. Hill, J. Harwell, D. Dierker, D.C. Van Essen, T.E. Inder, J.J. Neil, Comparison of cortical folding measures for evaluation of developing human brain. NeuroImage **125**, 780–790 (2016)
11. K. Zilles, E. Armstrong, A. Schleicher, H.J. Kretschmann, The human pattern of gyrification in the cerebral cortex. Anat. Embryol. **179**, 173–9 (1988)

12. T.W.J. Moorhead, J.M. Harris, A.C. Stanfield, D.E. Job, J.J.K. Best, E.C. Johnstone, S.M. Lawrie, Automated computation of the gyrification index in prefrontal lobes: methods and comparison with manual implementation. NeuroImage **31**(4), 1560–1566 (2006)
13. E. Lebed, C. Jacova, L. Wang, M.F. Beg, Novel surface-smoothing based local gyrification index. IEEE Trans. Med. Imaging **32**(4), 660–669 (2013)
14. S. Su, T. White, M. Schmidt, C.-Y. Kao, G. Sapiro, Geometric computation of human gyrification indexes from magnetic resonance images. Hum. Brain Mapp. **34**(5), 1230–1244 (2013)
15. E. Luders, P.M. Thompson, K.L. Narr, A.W. Toga, L. Jancke, C. Gaser, A curvature-based approach to estimate local gyrification on the cortical surface. NeuroImage **29**(4), 1224–1230 (2006)
16. S.H. Kim, I. Lyu, V.S. Fonov, C. Vachet, H.C. Hazlett, R.G. Smith, J. Piven, S.R. Dager, R.C. Mckinstry, J.R. Pruett Jr., A.C. Evans, D.L. Collins, K.N. Botteron, R.T. Schultz, G. Gerig, M.A. Styner, Development of cortical shape in the human brain from 6 to 24 months of age via a novel measure of shape complexity. NeuroImage **135**, 163–176 (2016)
17. R. Shishegar, J.H. Manton, D.W. Walker, J.M. Britto, L.A. Johnston, Quantifying gyrification using Laplace Beltrami eigenfunction level-sets, in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, pp. 1272–1275, 2015
18. M. Meyer, M. Desbrun, P. Schröder, A. Barr, *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, Book section 2. Mathematics and Visualization (Springer, Berlin, 2003), pp. 35–57
19. S.P. Awate, P.A. Yushkevich, Z. Song, D.J. Licht, J.C. Gee, Cerebral cortical folding analysis with multivariate modeling and testing: studies on gender differences and neonatal development. NeuroImage **53**(2), 450–459 (2010)
20. H. Zhang, O. van Kaick, R. Dyer, Spectral methods for mesh processing and analysis, in *Proceedings of Eurographics State-of-the-art Report* (2007), pp. 1–22
21. M. Berger, *A Panoramic View of Riemannian Geometry* (Springer, Berlin, 2003)
22. M. Reuter, S. Biasotti, D. Giorgi, G. Patane, M. Spagnuolo, Discrete Laplace-Beltrami operators for shape analysis and segmentation. Comput. Graph. **33**, 381–390 (2009)
23. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**, 129–150 (2011)
24. D. Gabor, Theory of communication. J. Inst. Electr. Eng. **93**, 429–457 (1946)
25. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**(2), 260–291 (2016)
26. B. Levy, Laplace-Beltrami eigenfunctions towards an algorithm that "Understands" geometry, in *IEEE International Conference on Shape Modeling and Applications, 2006. SMI 2006* (2006), pp. 13–13
27. N. Peinecke, F.-E. Wolter, M. Reuter, Laplace spectra as fingerprints for image recognition. Comput. Aided Des. **39**(6), 460–476 (2007)
28. M. Tan, A. Qiu, Spectral Laplace-Beltrami wavelets with applications in medical images. IEEE Trans. Med. Imaging **34**(5), 1005–1017 (2015)
29. G. Kaiser, Windowed fourier transforms, in *A Friendly Guide to Wavelets*. Modern Birkhuser Classics (Birkhauser, Boston, 2011), pp. 44–59
30. S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*, 3rd edn. (Academic, New York, 2008)
31. D. Grebenkov, B. Nguyen, Geometrical structure of Laplacian eigenfunctions. SIAM Rev. **55**(4), 601–667 (2013)
32. A. Golbabai, H. Rabiei, Hybrid shape parameter strategy for the RBF approximation of vibrating systems. Int. J. Comput. Math. **89**(17), 2410–2427 (2012)
33. M. Reuter, F.-E. Wolter, N. Peinecke, Laplace Beltrami spectra as Shape-DNA of surfaces and solids. Comput. Aided Des. **38**(4), 342–366 (2006)
34. C. Wachinger, P. Golland, W. Kremen, B. Fischl, M. Reuter, BrainPrint: a discriminative characterization of brain morphology. NeuroImage **109**, 232–248 (2015)
35. G. Auzias, J. Lefèvre, A.L. Troter, C. Fischer, M. Perrot, J. Régis, O. Coulon, Model-driven harmonic parameterization of the cortical surface: HIP-HOP. IEEE Trans. Med. Imaging **32**(5), 873–887 (2013)

36. D. Germanaud, J. Lefèvre, R. Toro, C. Fischer, J. Dubois, L. Hertz-Pannier, J.F. Mangin, Larger is twistier: spectral analysis of gyrification (SPANGY) applied to adult brain size polymorphism. NeuroImage **63**, 1257–72 (2012)
37. K. Im, J.-M. Lee, O. Lyttelton, S.H. Kim, A.C. Evans, S.I. Kim, Brain size and cortical structure in the adult human brain. Cereb. Cortex **18**(9), 2181–2191 (2008)

Wavelet-Based Visual Data Exploration



Alcebiades Dal Col, Paola Valdivia, Fabiano Petronetto, Fabio Dias, Claudio T. Silva and L. Gustavo Nonato

Abstract The wavelet coefficients associated with each node of the graph encode information about the signal under analysis considering all nodes in its neighborhood. However, understanding and extracting insight out of this wealth of information can be a challenging task. In this chapter, we will briefly review how the wavelet coefficients can be interpreted and explore which visual analytics resources can be leveraged. The visual representation of wavelet coefficients is still an application-dependent open problem in visualization, but recent developments introduced alternatives to specific cases, such as geo-referenced urban data and dynamic graphs.

1 Introduction

Graphs are often used to model real world problems, because they are flexible structures able to represent relationships between entities in an irregular domain [1, 2]. When a problem is successfully modelled into a graph, several well-defined methods

A. Dal Col (✉) · P. Valdivia · L. Gustavo Nonato
ICMC - USP, São Carlos, Brazil
e-mail: alcebiades_dalcol@usp.br

P. Valdivia
e-mail: paolalv@icmc.usp.br

L. Gustavo Nonato
e-mail: gnonato@icmc.usp.br

F. Petronetto
Federal University of Espirito Santo, Vitória, Brazil
e-mail: fabiano.carmo@ufes.br

F. Dias
University of Toronto, Toronto, Canada
e-mail: fabio.dias@utoronto.ca

C. T. Silva
NYU, New York, USA
e-mail: csilva@nyu.edu

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_14

can be leveraged [3]. When the graph represents a domain with a function associated to its elements, graph signal processing (GSP) [4–6] can be used to analyze that function and extract relevant information.

Our interest here lies in a sub-domain of GSP, the graph wavelet transform (GWT), in its different definitions [7–9]. When compared to the graph Fourier transform [10, 11], the GWT provides localization information, better support for non-periodic signals, and it is more robust to noise.

The GWT is particularly useful to identify the location of patterns and outliers of a function defined over the nodes [12] or edges [13] of a graph, because the resulting coefficients express the signal differences in different scales. By examining the coefficients of a node, one can gain insight about the behavior of the signal at and around it. The coefficients can be used as feature vectors, allowing node clustering and classification, revealing broader patterns in the data.

While the information is present on the coefficients, visually representing it in a meaningful way is still an open problem. One possible solution is to represent only small portions of the data at a time, as Mohan et al. [13] used to analyze unusual traffic conditions, where magnitude plots and scaleograms are created to illustrate the coefficients for linear sub-graphs representing streets of interest.

In the general case, visualizing the coefficients involves visual graph representation [14], itself a challenging problem, but considering a multi channel signal, where each channel represents one of the scales of the transform. When urban data is considered, this problem is somewhat simplified, since each node has a determined geographic location the node positioning does not need to be computed. In both cases, temporal information might also be present, which further complicates the representation.

Moreover, the proper interpretation of the coefficients requires deep knowledge of the GWT, and can be cognitively complex. Therefore, a more popular option is to not display the coefficients, but to use them to create insights, relevant to the application, that can be more easily interpreted. For instance, Tremblay and Borgnat [15] consider the GWT to perform community mining, the result is a node-link plot with clusters of nodes with similar behavior. Valdivia et al. [16] use a graph model of Manhattan's streets to explore taxi pickups, where each node is classified through its coefficients into different classes of behavior, allowing the quick identification of outliers. The temporal behavior is depicted by a streamgraph of the quantities of each class.

While the relevant insights derived from the GWT are usually application-dependent, we explore how to visually represent them in this chapter, aiming to provide general guidelines and examples of some of the proposed solutions. In the first section, we explain what information is present in the coefficients and the effects of the topology in the results. The second section illustrates how that information was leveraged in the literature for two real world problems: geo-referenced urban data and interpersonal contact graphs.

2 Meaning of Wavelet Coefficients

The main goal of this section is to familiarize the reader with the information present in the wavelet coefficients. We start by reviewing the necessary theoretical aspects, then we explain how to extract insight from the wavelet coefficients, and how they vary in function of the topology of the graph. At the end, we explore the effects of the choice of graph wavelets on the coefficients.

2.1 Notation and Conventions

In this chapter, graphs are denote by upper case letters (G, H, ...), defined as a set of n nodes V, a set of edges E, and, optionally, a function f that associates real numbers to the nodes of the graph, which we often refer to as a *signal*. Whenever necessary, we indicate to which graph each set belongs, such as the set V_G of nodes of the graph G. Specific nodes and edges are denoted by τ and e, respectively. Eigenvectors and eigenvalues are denoted by u and λ, respectively, and are assumed to be ordered such that $\lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_n$. We adopt eight coefficients for the GWT, where coefficients from 1 to 3 are interpreted as representing low frequencies, 4 and 5 medium frequencies, and 6 to 8 to high frequencies. These coefficients are visually represented as bar plots of their magnitudes.

2.2 Brief Review of Graph Wavelet Transform

Similarly to the traditional signal processing case, the wavelets and Fourier transforms are defined as the decomposition of the signal into a combination of signals of known frequencies. However, the concept of frequency is not as easily defined in a graph domain. Indeed, while the frequencies form a continuous spectrum for the traditional case, with a defined physical meaning, in graphs they correspond to the finite set of eigenvalues of the Laplacian matrix.

While the graph Fourier transform usually considers all these frequencies independently, the GWT adopts *scales*, where several eigenvalues are aggregated, following a function called *kernel*. Therefore, the wavelet coefficients for each scale are defined by how much "energy" is present on the coefficients, weighted by the corresponding kernel function.

Formally, let f be a signal defined on the nodes of G. A GWT is a function $W : \{1, 2, \ldots, M\} \times V \to \mathbb{R}$ defined as

$$W(m, \tau) = \sum_{\ell=1}^{n} \hat{g}_m(\lambda_\ell)\hat{f}(\lambda_\ell)u_\ell(\tau), \tag{1}$$

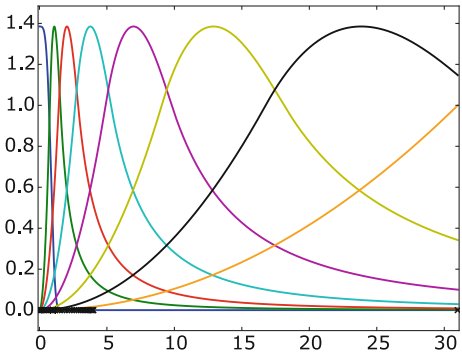


Fig. 1 Example of kernels defined over the eigenvalue space of the graph Laplacian. Each curve is the kernel for a different scale, the crosses on the horizontal axis represent eigenvalues. Adapted from [17] with permission

where $\hat{f}$ is the graph Fourier transform of f and $\{\hat{g}_m \,|\, m = 1, 2, \ldots, M\}$ is a dictionary of kernels. By varying m while keeping τ fixed, we obtain the M wavelet coefficients associated with node τ. While there are different choices for the kernels, they are usually formed by band-pass filters, as illustrated in Fig. 1. We further explore these kernel functions on Sect. 2.5.

2.3 *Decoding the Wavelet Coefficients*

Since the scales of the GWT are related to the eigenvectors of the Laplacian matrix of the graph, we adopt signals derived from those eigenvectors, leading to predictable outputs. Additionally, to avoid the potentially complicated neighborhood interactions, we start our exploration with a simple topology.

Example 2.3.1 (*Path graph*) A path graph is a graph whose nodes are adjacent to exactly two other nodes, with the exception of the two extreme ones that are connected to only one node. Consider a path graph with 64 nodes. We define a signal f_0 on the nodes of the graph as the combination of a low (u_{11}) and a high (u_{61}) frequency eigenvectors, which are illustrated in Fig. 2.

$$f_0(\tau) = \begin{cases} u_{11}(\tau) \text{ if } \tau \in \{\tau_i \,|\, 01 \leq i \leq 30\} \\ u_{61}(\tau) \text{ if } \tau \in \{\tau_i \,|\, 31 \leq i \leq 64\} \end{cases}. \tag{2}$$

The signal f_0 corresponds to u_{11} in the first thirty nodes and u_{61} in the others. Figure 3 illustrates the signal and the coefficients for two nodes, τ_{15} and τ_{45}, away from node τ_{30}, where the transition occurs. The distribution of the magnitudes of the coefficients is skewed towards low frequencies for τ_{15} and high frequencies for τ_{45}, correctly capturing the frequencies used to build each part of the signal. While the scales are defined using the eigenvalues, the only values where the frequencies are defined for the graph, they do not have a one to one relationship. Each scale extends through several eigenvalues, and each eigenvalue is associated to more than one scale,

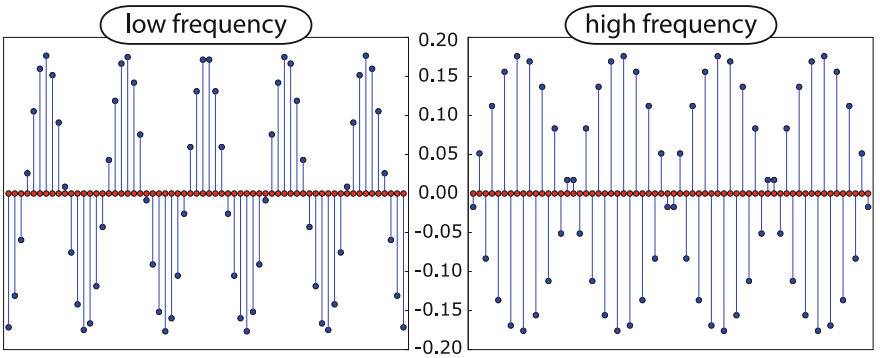


Fig. 2 Signals u_{11} and u_{61} defined on the path graph. Red dots represent the nodes, and the height of the vertical blue lines corresponds to the amplitude of the signal. Adapted from [17] with permission

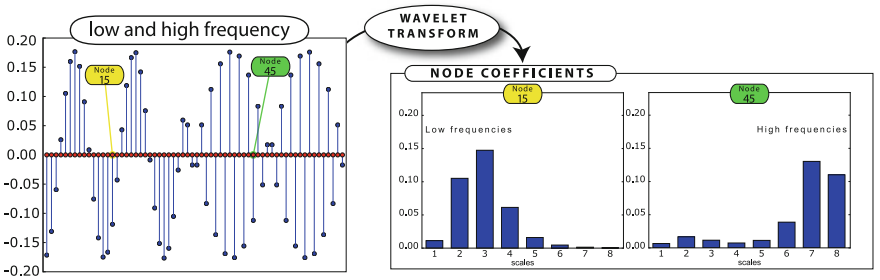


Fig. 3 The signal f_0 defined over the path graph and the corresponding wavelet coefficients of nodes τ_{15} and τ_{45}. Adapted from [17] with permission

leading to these frequency distributions. An increase in the number of scales would reduce this effect, but it would also reduce the robustness to noise. Nevertheless, the ideal number of scales is an application dependent parameter, albeit not a critical one. ■

The path graph is essentially the same as a discretization of the real line; the graph topology is fairly regular. This simple example demonstrates how the coefficients correctly identified low and high frequencies. In the next example, we consider a graph with a more irregular topology, with the signals still derived from the eigenvectors.

Example 2.3.2 (*Minnesota graph*) Consider the Minnesota road graph from [18], depicted in Fig. 4. We divide it into three different regions R_1 (blue), R_2 (red), and R_3 (green), as depicted in the left panel of Fig. 4. A signal with a different dominant frequency band is assigned to each region:

$$f := \sum_{j=1}^{3} \frac{f_j}{||f_j||_\infty} \tag{3}$$

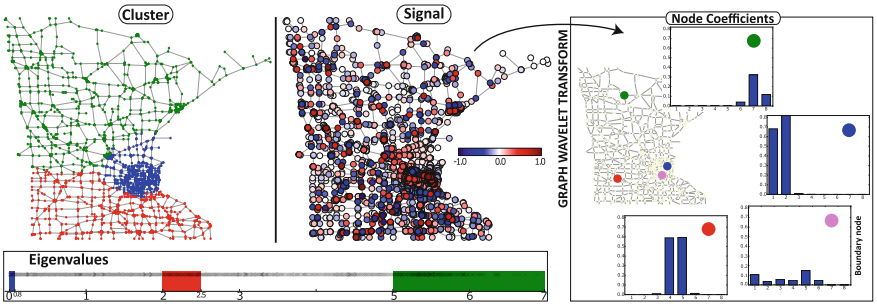


Fig. 4 Left: The Minnesota road graph divided into three different regions (cluster) and signal defined on the Minnesota road graph (signal). The signal is formed so each region has a dominant frequency band, highlighted in the eigenvalues panel. Right: The wavelet coefficients of representative nodes in the regions on the left (dot colors) and one node close to the boundary between regions (pink). Adapted from [17] with permission

where

$$f_j(\tau) = \begin{cases} \sum_{\ell \in I_j} u_\ell(\tau) & \text{if } \tau \in R_j \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

for $j = 1, 2, 3$, and I_j is the set of indexes of the eigenvalues in the intervals $[a_j, b_j]$. The intervals for I_1, I_2, and I_3 are given by $[0.0, 0.08]$, $[2.0, 2.5]$, and $[5.0, 7.0]$, respectively. The lengths of the intervals were chosen according to the distribution of the eigenvalues as illustrated in the eigenvalue panel of Fig. 4. Region R_1 (blue) has a low frequency signal, since the signal is the sum of the eigenvectors corresponding to eigenvalues in the low frequency region of the spectrum. Similarly, the signal in R_2 (red) and R_3 (green) corresponds to the sum of the eigenvectors associated with median and high frequency eigenvalues, respectively. The signal panel of Fig. 4 illustrates the signal f.

The node coefficients panel of Fig. 4 illustrates representative coefficients for each region, computed using the spectrum-adapted graph wavelets [19]. As expected, the nodes that are in the middle of their regions present clearly defined frequency distributions corresponding to the assigned signal, albeit some differences in the magnitudes. The nodes close to the boundary between regions tend to have a mixed behavior, as demonstrated for the pink node whose wavelet coefficients are illustrated in the node coefficients panel of Fig. 4. This node contains a mixture of low and medium frequencies, influenced by the low frequency blue region and the medium frequency red region. ■

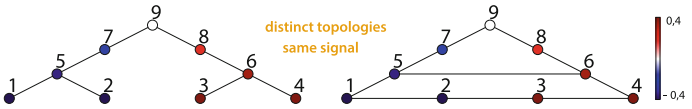


Fig. 5 Graphs G_1 and G_2, with different topology but the same signal associated to the nodes

2.4 The Interplay Between Topology and Coefficients

In the last two examples, we illustrated how the coefficients identify the frequencies present on the signal and how they can be intermixed in boundary zones. The following example illustrates how the topology of the graph impacts the wavelet coefficients, comparing the results of the GWT for the same signal defined over two graphs with different sets of edges.

Example 2.4.1 We define a graph G_1 with nine nodes, as illustrated on the left side of Fig. 5. The signal is defined using the second eigenvector of the graph Laplacian of G_1. The color of the nodes encodes the signal. Since a low frequency eigenvector was selected as signal, it varies smoothly across the graph topology. The coefficients indicate the presence of smooth signal variation on and around all nodes and they are quite similar for all nodes of G_1 varying mainly in magnitude, as illustrated at the top of the Fig. 6.

Now consider the graph G_2, illustrated on the right side of Fig. 5, which contains the same node set as G_1, but a different set of edges. A different topology means a different Laplacian matrix, and therefore a different set of eigenvectors. However, we maintain the same values associated with the nodes as the previous graph, the second eigenvector of G_1. Since the signal no longer corresponds to a low frequency eigenvector, we obtain a different distribution of frequencies, caused by abrupt changes across neighboring nodes. This effect was explored in the literature, with the objective of obtaining an edge set that would result in smooth transitions [20].

These abrupt variations in the function across the graph are reflected in the coefficients, as illustrated at the bottom of the Fig. 6, revealing the presence of signal variation at and around the nodes of graph G_2, with the exception of node 9. Indeed, this node has the same neighbors in both graphs.

Furthermore, the signal is smoother on nodes τ_1 and τ_4, which is quite evident in the wavelet coefficients (bottom of the Fig. 6), since nodes τ_1 and τ_4 have more pronounced low frequency coefficients. Nodes τ_7 and τ_8 have a pattern of wavelet coefficients similar to nodes τ_1 and τ_4, but with smaller magnitudes. The nodes with largest coefficients on high frequencies are τ_2 and τ_3, indicating a sharper signal variation on and around these nodes. ■

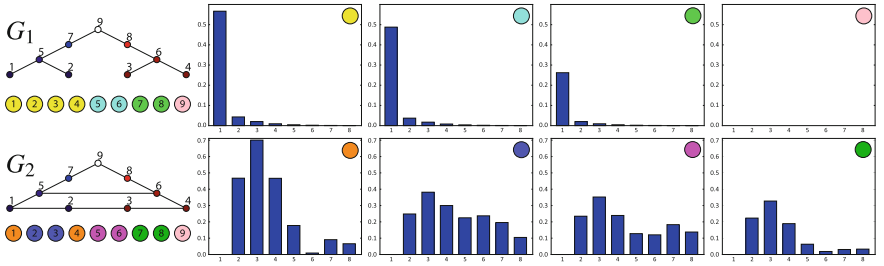


Fig. 6 Wavelet coefficients for all nodes of graphs G_1 and G_2 when the signal is defined as the second graph Laplacian eigenvector of graph G_1

2.5 *The Effects of the Choice of Graph Wavelets on the Coefficients*

While what was demonstrated in this chapter so far is relevant regardless of the dictionary adopted for the GWT, its choice can lead to dramatically different results. Spectral approaches are mostly represented by the spectral graph wavelets (SGW) [21], whose kernels uniformly cover the range of the graph spectrum, and the spectrum-adapted (tight) graph wavelets (SAGW) [19], whose kernels are adjusted to the distribution of the eigenvalues.

Hammond et al. [21] adopt the set of kernels $\{\hat{h}(\lambda), \hat{g}(s_r\lambda), \ldots, \hat{g}(s_1\lambda)\}$ as dictionary for the SGW, where $s_1, s_2, \ldots, s_r$ are logarithmically sampled between $s_1 = 2/\lambda_n$ and $s_r = 40/\lambda_n$,

$$\hat{g}(x) = \begin{cases} x^2 & \text{for } 0 \leq x < 1 \\ -5 + 11x - 6x^2 + x^3 & \text{for } 1 \leq x \leq 2 \\ 4x^{-2} & \text{for } 2 < x \end{cases}, \text{ and} \tag{5}$$

$$\hat{h}(x) = \gamma e^{-\left(\frac{10x}{0.3\lambda_n}\right)^4}. \tag{6}$$

The parameter γ is chosen such that $\hat{h}(0)$ is equal to the maximum value of $\hat{g}$. An example of this approach is illustrated on the left side of Fig. 8. Note that only the largest eigenvalue is used to define these kernels, indirectly assuming that the distribution of eigenvalues is somewhat uniform.

For the SAGW, Shuman et al. [19] propose a dictionary where the kernels are composed from functions translated accordingly to each eigenvalue of the Laplacian matrix. In more mathematical terms, there is a function $\hat{g}^U$ and a constant a such that the graph spectral filters $\hat{g}_m$ can be written as

$$\hat{g}_m(\lambda) = \hat{g}^U(\lambda - ma), \forall\lambda \in [0,\lambda_n], \tag{7}$$

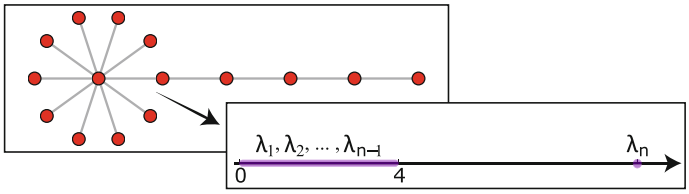


Fig. 7 Comet graph and corresponding spectral distribution. The purple interval contains $n-1$ graph Laplacian eigenvalues. The largest eigenvalue is isolated; the greater the degree of star graph, the greater the distance. Adapted from [17] with permission

for $m = 1, 2, \ldots, M$. Therefore, in this approach, the larger the number of eigenvalues in a certain region of the spectral domain, the narrower the graph spectral filters are in that region, as illustrated on the right side of Fig. 8.

We explore the differences between these approaches by analyzing another simple graph in the following example.

Example 2.5.1 (*Comet graph*) A comet graph is a graph formed by combining a path graph and a star graph, the latter centered in one of the extremities of the path graph. Figure 7 illustrates a comet graph with 15 nodes whose star extremity contains 10 nodes. The graph Laplacian eigenvalues of a comet graph present a characteristic distribution, where the difference between the two largest eigenvalues is much larger than the difference between any other sequential eigenvalues, as illustrated in the bottom right of Fig. 7. Although the second largest eigenvalue is distant from the largest eigenvalue, it corresponds to the second highest frequency in the spectrum of the comet graph.

We consider the SGW [21] and SAGW [19], both defined in the spectrum of the comet graph depicted in Fig. 7. The kernels derived from each dictionary are quite

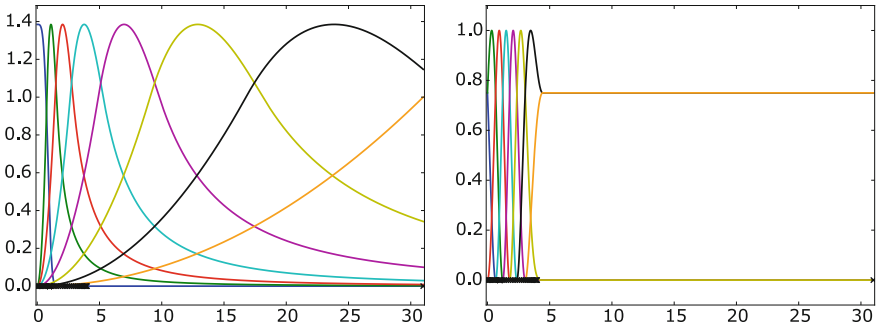


Fig. 8 Kernels forming the SGW (left) and SAGW (right) dictionaries for a comet graph with 64 nodes. The kernels of the SAGW dictionary adapt to regions with larger spectral density. The $\times$ marks on the horizontal axes represent the spectrum of the comet graph. Adapted from [17] with permission

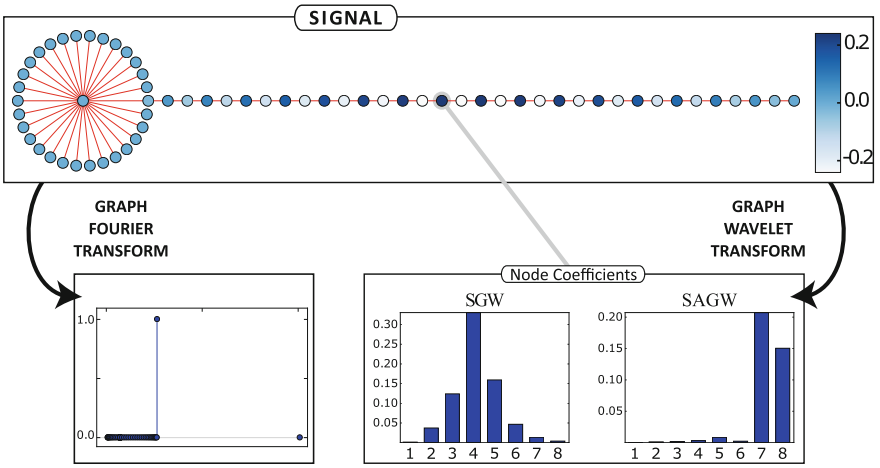


Fig. 9 Second largest graph Laplacian eigenvector defined on a comet graph with 64 nodes. The color associated with the graph nodes encodes the signal. The arrows indicate the corresponding graph Fourier transform and wavelet coefficients of the specified node. The graph Fourier transform reveals the presence of high global frequency. Adapted from [17] with permission

different, as illustrated by Fig. 8. As expected, the SGW evenly covers the spectral domain, while the SAGW adapts itself to the spectral distribution.

Consider now a comet graph with 64 nodes whose star extremity containing 30 nodes. The signal is defined as the second largest graph Laplacian eigenvector, as illustrated by the signal panel in Fig. 9. Figure 9 depicts the wavelet coefficients of the highlighted node using both the SGW and SAGW dictionaries.

While both coefficients correctly capture the composition of the signal according to their dictionaries, the interpretation can be easier for the SAGW in this extreme case. Since there is a significant gap between the two largest eigenvalues, the SGW coefficients from six to eight will never achieve larger magnitudes. Both cases correspond to one of the highest frequencies possible in this graph, but one would need to know the distribution of the eigenvalues beforehand to arrive at this conclusion using the SGW. However, the SAGW coefficients do not maintain the relative proportion between the eigenvalues; the two significantly different largest eigenvalues are represented closely in the coefficients. Therefore, the choice between SGW and SAGW is application-dependent, considering how the coefficients are further processed. ■

3 Wavelet-Based Visual Data Exploration

The main goal of this section is to demonstrate how the information present in the wavelet coefficients can be used to analyze real data. We use methods from the literature, considering geo-referenced urban data [16, 17] and dynamic graphs [12].

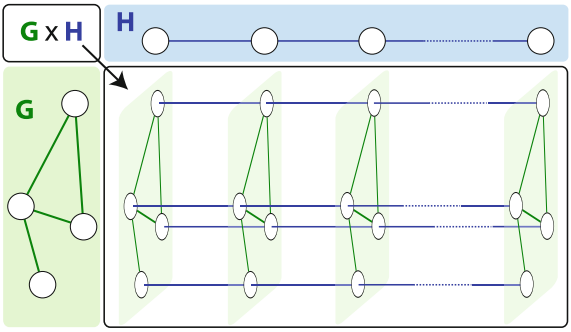


Fig. 10 Time-varying data modeled by the Cartesian product of two graphs: spatial G and temporal H. Adapted from [17] with permission

3.1 Analyzing Time Series Defined on Nodes of a Graph

The discussion in Sect. 2 assumes a scalar signal defined on the nodes of a graph. However, in many applications data varies with time, which is often represented as a time series associated with each node of the graph. One of the possible options to adapt it to the framework of GWT is through a Cartesian product of graphs.

Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two graphs. To clarify the notation, we represent the nodes of G as $\tau_i \in V_G$ and the nodes of H as $\iota_i \in V_H$. The Cartesian product of G by H is a graph $G \times H$ with node set $V_G \times V_H = \{\tau_i \iota_j \mid \tau_i \in V_G, \iota_j \in V_H\}$ and edges connecting nodes $\tau_i \iota_j$ and $\tau_k \iota_l$ if and only if $\tau_i = \tau_k$ and ι_j is adjacent to ι_l in H, or $\iota_j = \iota_l$ and τ_i is adjacent to τ_k in G. If H is a path graph, the Cartesian product graph $G \times H$ corresponds to "stacked" copies of G with edges connecting corresponding nodes in adjacent copies of G (Fig. 10). Each copy of G is interpreted as a time slice of the temporal data, enabling the use of the graph wavelet transform on $G \times H$ to analyze time-varying data. In this sense, we will refer to the edges that are derived from G as *spatial*, because that graph encodes the relationship between the nodes on a given instant, and the edges derived from H as *temporal*, because they connect the same entity across time.

In the Cartesian product graph, the wavelet coefficients capture the spatio-temporal behavior of the signal, making their interpretation more intricate than in the static case, because they indicate the behavior of the signal around each node, but cannot distinguish between spatial or temporal variation.

Example 3.1.1 Consider the graphs G and H depicted in Fig. 11, representing spatial and temporal information, respectively. Since H is a path graph with nineteen nodes, the graph $G \times H$ can be used to represent nineteen time slices, with a scalar value associated to each node. We craft four different signals on $G \times H$, denoted as f_i, $i = 1, 2, 3, 4$, defined using the nodes $\tau_a \in G$ and $\iota_{10} \in H$ as highlighted in Fig. 11.

The signal f_1 is defined as 1 on node $\tau_a \iota_{10}$ of $G \times H$ and zero elsewhere. The signal f_2 is zero at all time slices, with the exception of the time slice 10 where the values smoothly decrease from the peak centered at $\tau_a \iota_{10}$; this signal can be interpreted as a smooth spatial event that occurs at only one time interval. Similarly,

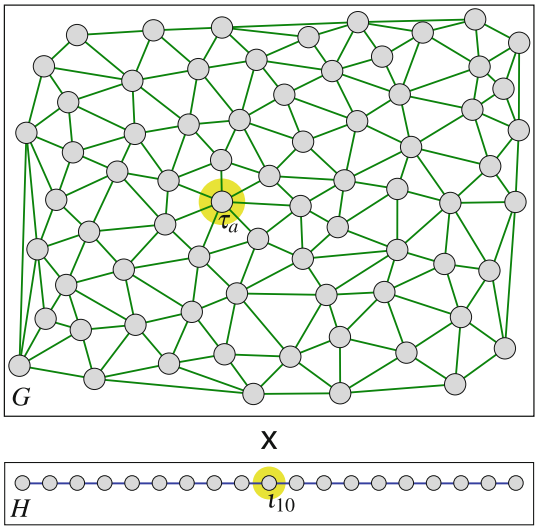


Fig. 11 Graphs: spatial G and temporal H. Adapted from [17] with permission

the signal f_3 is defined as zero on all nodes, with the exception of the nodes $\tau_a\iota_j$, $j = 1, \ldots, 19$, where the values smoothly decrease around the peak at $\tau_a\iota_{10}$; this can be interpreted as an event that is concentrated spatially, increasing and then decreasing in magnitude over time. Finally, the signal f_4 has a peak at $\tau_a\iota_{10}$, with the values smoothly decreasing around it; it can be interpreted as a spatially defined event that grows and decreases in magnitude over time.

Figure 12 depicts how the values are distributed in the Cartesian graph, for all four defined functions, along with the resulting coefficients associated with the node $\tau_a\iota_{10}$. In the bar chart corresponding to signal f_1, there is a concentration of large coefficients in high frequencies, indicating that f_1 has a high frequency on $\tau_a\iota_{10}$; a significant signal change around this node. The coefficients obtained for signal f_2 are similar, but with increased presence of lower frequency coefficients, also indicating a change around this node, but not quite as strong. Remembering that the wavelets are locally defined and that the only difference between f_1 and f_2 corresponds to the "temporal" neighbors of $\tau_a\iota_{10}$, these lower frequencies are a result of the smooth temporal transitions around the node.

The coefficients from f_3 identified a considerably lower frequency distribution, indicating smoother changes. While this node still contains two different temporal neighbors, its spatial neighborhood has similar values in all its seven nodes. Indeed, when we remove some of the spatial edges, as illustrated in Fig. 13, we obtain lower frequencies. When the transition is smooth for all of these nodes, as in f_4, we obtain a dominantly low frequency distribution. ■

To further illustrate the influence of the temporal and spatial edges, we explore real world data in the next example.

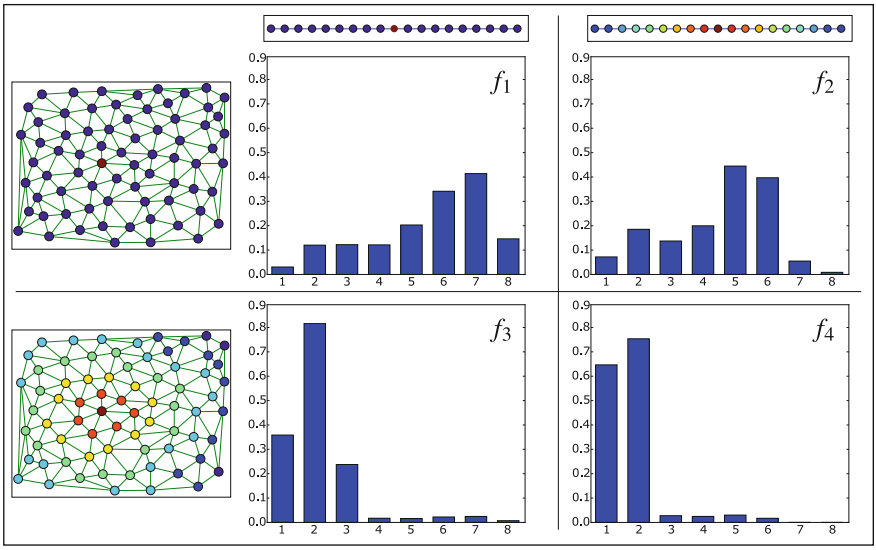


Fig. 12 Wavelet coefficients of node $\tau_a \iota_{10}$ for signals $f_i, i = 1, 2, 3, 4$. The graph wavelet transform was produced using the SAGW dictionary. Adapted from [17] with permission

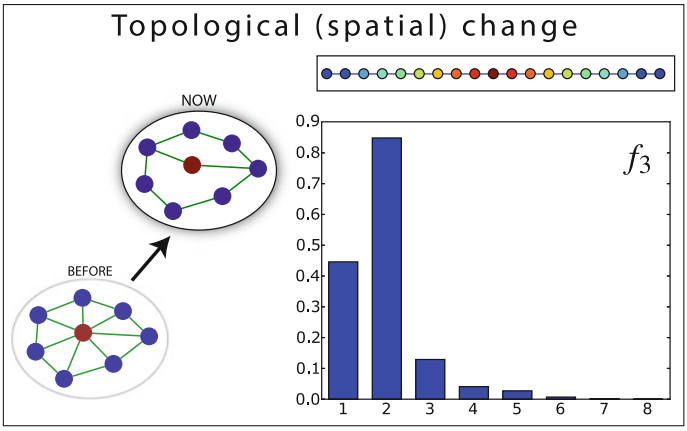


Fig. 13 Wavelet coefficients of node $\tau_a \iota_{10}$ for signal f_3 after edges removal. The graph wavelet transform was produced using the SAGW dictionary. Adapted from [17] with permission

Example 3.1.2 (*Manhattan taxi pick-ups*) This example considers real data from New York City Taxi and Limousine Commission, containing taxi pick-ups for August 12, 2013. The underlying graph is defined using the street network, with nodes and edges representing corners and streets, respectively. We assign to each node the number of taxi pick-ups that occurred near the corresponding corner in a specific time interval. For this simple example we consider three time intervals, half hour each, from 07:00–08:30.

To allow for a direct comparison of the effect of the temporal edges, we computed the GWT considering just the middle time slice separately, without any temporal edges, and considering all three time slices. For simplicity, we refer to these coefficients as *spatial* and *spatio-temporal*.

The Fig. 14 depicts downtown Manhattan graph with corners colored by the number of taxi pick-ups in August 12, 2013, 07:30–08:00. Zoom-in views in Fig. 14 show the number of taxi pick-ups at and around the corner of 8th Avenue with 41st Street, the front of the Port Authority Bus Terminal (PABT), in the consecutive time slices 07:00–07:30, 07:30–08:00, and 08:00–08:30. The corresponding spatial and spatio-temporal coefficients are illustrated in Fig. 15, considering the time slice representing the interval 07:30–08:00. The spatial coefficients indicate a higher distribution of frequencies than the spatio-temporal coefficients, because the signal under study is highly concentrated spatially, with significant temporal consistency. In other words, there are always more people entering a taxi in front of the PABT, regardless of the hour, than on the next corner of 41st Street. The temporal edges are used to represent this temporal continuity, leading to coefficients that correctly identify that, while there is change in the signal around those nodes, there is also temporal continuity. ■

While the Cartesian product graph is suitable to represent time-varying information, including methods for a faster computation of the eigenvalues and eigenvectors [22], it cannot be used when the relationships between the represented entities also change over time. This case can be represented by changing the topology of the graph, leading to a dynamic graph.

3.2 Analyzing Dynamic Graphs

Dynamic graphs are similar to the Cartesian product graph, in the sense that subgraphs are used to represent specific temporal intervals, but it does not include the constraint of these subgraphs having the same topology. Nodes representing the same entity in subsequent time slices are still connected. This concept is illustrated in Fig. 16.

Albeit simple, the difference between Cartesian product graphs and dynamic graphs means that the faster methods for the computation of the eigendecomposition of the Laplacian cannot be used. Approximation methods are often employed to allow the use of the GWT on large dynamic graphs in reasonable time [8, 12, 21].

In the node-link diagrams presented in this subsection, the nodes are positioned using the Fruchterman-Reingold force-directed algorithm [23] on a graph that

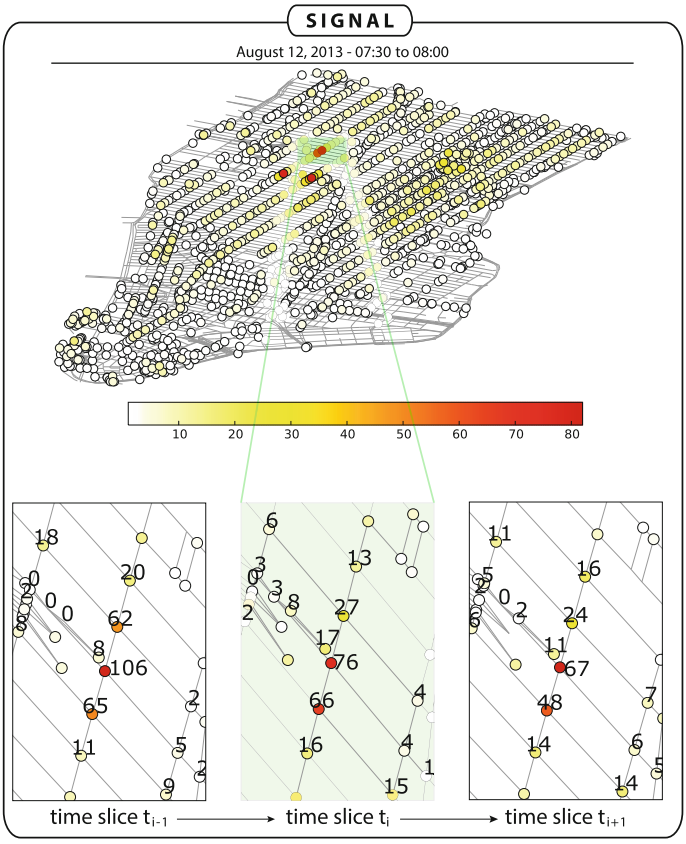


Fig. 14 Number of taxi pick-ups in downtown Manhattan on August 12, 2013, 07:30–08:00 (top) and at the front of the PABT in the consecutive time slices 07:00–07:30, 07:30–08:00, and 08:00–08:30 (bottom). Adapted from [17] with permission

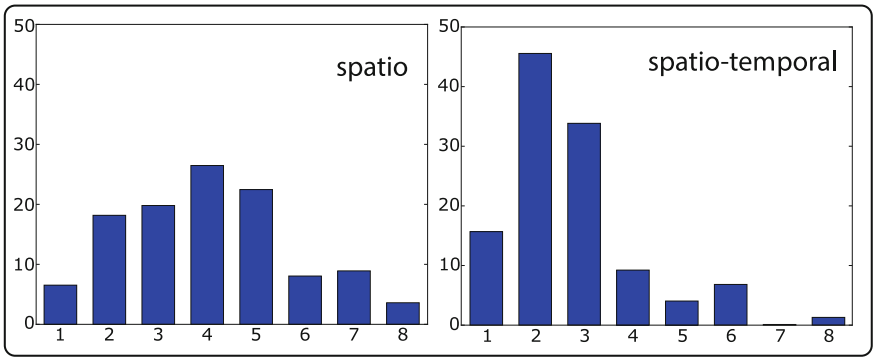


Fig. 15 Spatial and spatio-temporal wavelet coefficients (SAGW) of node in front of the PABT corresponding to August 12, 2013, 07:30–08:00. Adapted from [17] with permission

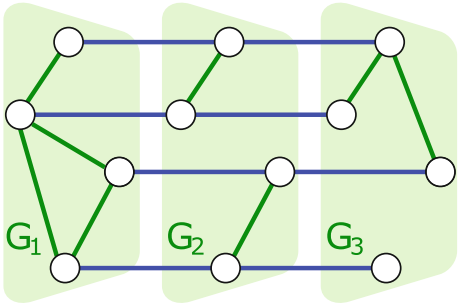


Fig. 16 Dynamic graph model. Adapted from [12] with permission

contains one node for each entity represented in the dynamic graph; edges are placed between all connected entities in the dynamic graph, regardless of the temporal slice. This can be interpreted as a "compressed" version of the dynamic graph. This ensures positioning consistency across temporal slices, reducing the cognitive load on the users.

Example 3.2.1 We generated a dynamic graph containing 250 entities and 13 time slices [12]. The signal associated with a node is the number of edges incident to the node in its time slice. There are two large spatial events in the graph, depicted in the leftmost panel of Fig. 17, corresponding to an activity peak (many edges) at time slice 4, which decrease in size until time slice 6. At time slice 7, several small spatial events (central panel in Fig. 17) appear, with some of them disappearing on the next time slices. At time slice 10, another two large spatial events (rightmost panel in Fig. 17) are created reaching a peak at time slice 11 and decreasing at time slice 12.

Figure 17 depicts the nodes of the synthetic dynamic graph colored according to the distribution of wavelet coefficients. Isolated nodes are assigned to a specific class (white). The other nodes are separated into four different classes based on their frequency distribution. Dark and light blue nodes indicate smooth signal variation as the low frequency coefficients are predominant and, consequently, a concentration of nodes in these classes suggests large and/or slow events, which is the case for time slices 4 and 10. Orange and red nodes indicate that the signal varies abruptly, revealing small/quick events; in time slice 7 there are several small events where the wavelet coefficients accurately classify the center nodes as high frequency. The yellow class includes the nodes which frequency distribution were balanced, not dominated by high nor low frequencies, usually including a mix of both.

This approach leads to a straightforward interpretation, considerably easier than interpreting the coefficients directly, where blue indicates smooth variations and red indicates abrupt variations, considering both spatial and temporal neighborhood. By leveraging this classification, we can summarize the topological evolution of the network, using a line plot, as illustrated in Fig. 18. In this plot, the horizontal axis corresponds to the time slices; the vertical axis corresponds to the sum of the function over the graph, which indicates the level of activity for this example; and the color of the marker indicates the class with proportionally more presence. This proportion

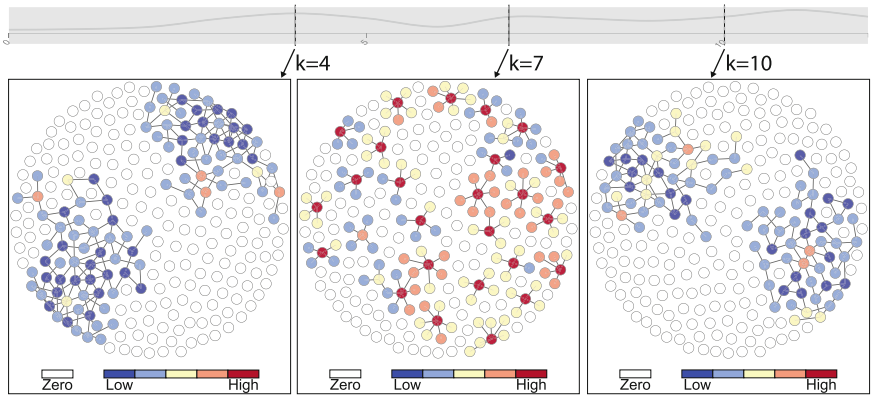


Fig. 17 Node classification for the synthetic dynamic graph. Adapted from [12] with permission

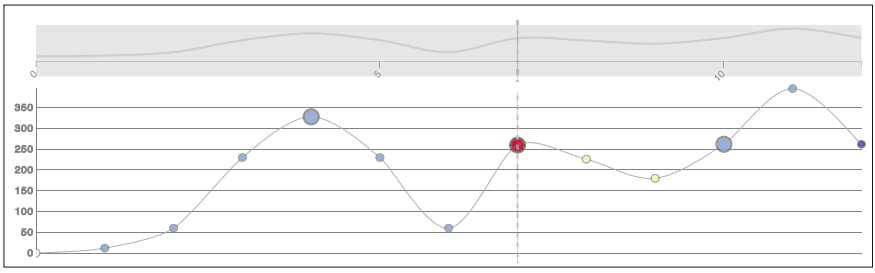


Fig. 18 Time series as a resource to bring up the information inferred by the walevet coefficients when analyzing dynamic graphs. Adapted from [12] with permission

is defined based on the maximum presence of each class at any given time, not the absolute number of nodes. For instance, considering Fig. 18, time slices 4, 7, and 10 have similar level of activities, as indicated by the vertical position, but times 4 and 10 contain large/slow events, while time 7 contains small/quick events, as corroborated by the node-link plots in Fig. 17. ■

Example 3.2.2 (*High school dynamic graph*) The high school dataset contains face-to-face contact information between 180 students from a school in France, during nine days in November of 2012 [24]. Each student belongs to one of five different classes. The original data provides contact information between students in intervals of 20 s, which we aggregated further by creating a time slice every 6 min, for a total of 2,027 time slices. The resulting dynamic graph has 14,788 edges, not counting the temporal edges. The signal associated with each node is the number of face-to-face contacts made by the corresponding child in the corresponding time slice and the weight of each edge corresponds to the number of face-to-face contacts between the two involved people.

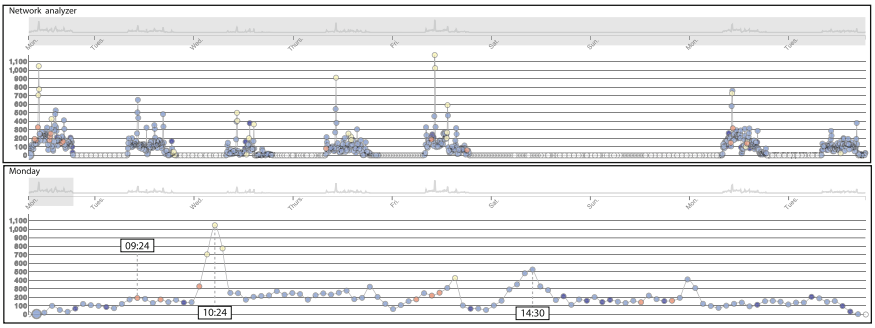


Fig. 19 Time series like representation for the high school dataset. It shows the whole dynamic graph at the top and only the first Monday (zoomed-in view) at the bottom. Adapted from [12] with permission

Figure 19 depicts a line plot with markers as proposed in the previous example. This visualization reveals the predominance of light blue nodes, that is, in most of the time slices the signal faces smooth variation, meaning that children interact primarily in balanced contact groups. In some time slices they are classified as orange, directing our attention to a different pattern. The 9:24 time slice on the first Monday is the first classified as orange on this dataset, which is illustrated in Fig. 20. By inspecting the 9:24 time slice, it is clear that this time slice mostly contains pairs of contacts, with

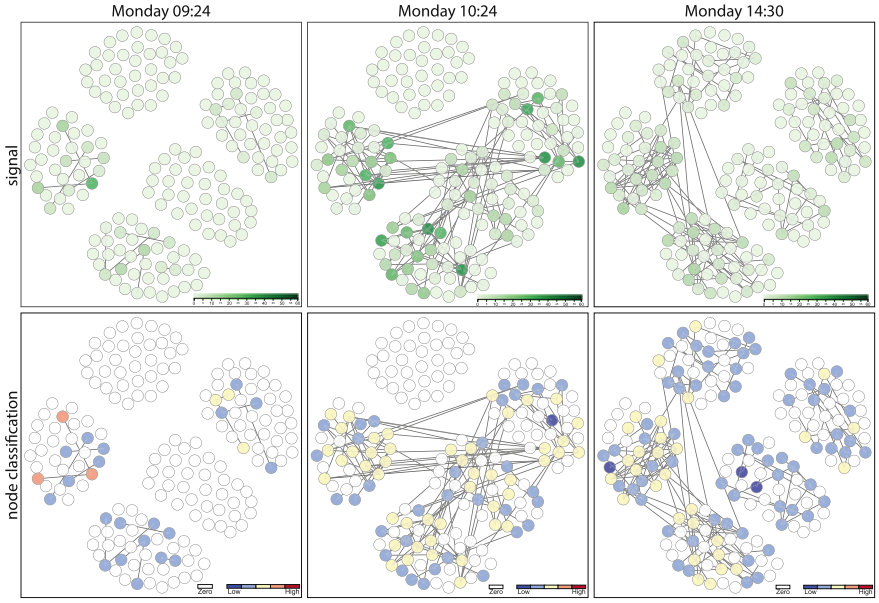


Fig. 20 Signal and node classification for the first Monday at 9:24, 10:24, and 14:30 time slices. Adapted from [12] with permission

low signal (number of contacts). The nodes are mostly light blue. However, three connected nodes on the leftmost student class present a higher signal with the signal on the center one considerably outpacing the signal on other two nodes, leading to a orange classification, which leads to the classification of this time slice as orange as well, since this is one of the highest concentration of orange nodes at a given time.

The 14:30 time slice on the first Monday contains a considerable amount of activity, indicated by the height in the time series, and the time slice is classified as light blue. The combination of increased height and low frequency classification means it contains a large event, without signal peaks. Smaller events or peaks would lead to higher frequencies. Indeed, as illustrated in the rightmost column of Fig. 20, the signal varies smoothly across a large event. ■

4 Conclusion

We have shown in this chapter how the graph wavelet transform can be employed to analyze real data. The first section leads the reader through examples that allow to acquire knowledge and some intuition about the information present in the wavelet coefficients. The wavelet coefficients reflect the variation of the signal under analysis across the topology of the graph, topology which can be quite complicated sometimes. The second section makes use of this information to explore patterns difficult to be perceived without the use of graph wavelets. For instance, the spatial relation between different time series defined on nodes of a graph; the contact pattern of a node and its neighbors in a dynamic graph.

The wavelet-based visual data exploration has room for much more developments and applications, since from our viewpoint the graph wavelet transform has a huge potential still unexplored.

Acknowledgements Grants 2011/ 22749-8, 2013/ 14089-3, 2014/ 12815-1, 2015/ 03330-7, 2016/ 04391-2, and 2016/ 04190-7 from São Paulo Research Foundation (FAPESP). The views expressed are those of the authors and do not reflect the official policy or position of the São Paulo Research Foundation. Fabio Dias was partially supported by the Urban Genome Project, an University of Toronto Connaught Global Challenge grant.

References

1. M.O. Jackson, *Social and Economic Networks* (Princeton university press, 2010)
2. C. Nowzari, V.M. Preciado, G.J. Pappas, Analysis and control of epidemics: a survey of spreading processes on complex networks. IEEE Control Syst. **36**(1), 26–46 (2016)
3. T.G. Lewis, *Network Science: Theory and Applications* (Wiley, New York, 2011)
4. Laurent Najman, Jean Cousty, A graph-based mathematical morphology reader. Pattern Recognit. Lett. **47**, 3–17 (2014)
5. A. Ortega, P. Frossard, J. Kovaevi, J.M.F. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications. Proc. IEEE **106**(5), 808–828 (2018). May

6. Aliaksei Sandryhaila, Jose M.F. Moura, Discrete signal processing on graphs: frequency analysis. IEEE Trans. Signal Process. **62**(12), 3042–3054 (2014)
7. M. Crovella, E. Kolaczyk, Graph wavelets for spatial traffic analysis, in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3 (IEEE, 2003), pp. 1848–1857
8. D.I. Shuman, P. Vandergheynst, P. Frossard, Chebyshev polynomial approximation for distributed signal processing, in *IEEE International Conference on Distributed Computing in Sensor Systems and Workshops* (Institute of Electrical and Electronics Engineers, 2011), pp. 1–8
9. A. Silva, X.H. Dang, P. Basu, A. Singh, A. Swami, Graph wavelets via sparse cuts, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2016), pp. 1175–1184
10. A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs: graph Fourier transform, in *ICASSP* (2013), pp. 6167–6170
11. D.J. Shuman, B. Ricaud, P. Vandergheynst, A windowed graph fourier transform, in *Statistical Signal Processing Workshop (SSP), 2012 IEEE* (IEEE, 2012), pp. 133–136
12. A. Dal Col, P. Valdivia, F. Petronetto, F. Dias, C.T. Silva, L.G. Nonato, Wavelet-based visual analysis of dynamic networks. IEEE Transactions on Visualization and Computer Graphics (2017)
13. D.M. Mohan, M.T. Asif, N. Mitrovic, J. Dauwels, P. Jaillet, Wavelets on graphs with application to transportation networks, in *IEEE International Conference on Intelligent Transportation Systems* (IEEE, 2014), pp. 1707–1712
14. C. Vehlow, F. Beck, D. Weiskopf, The state of the art in visualizing group structures in graphs, in *Eurographics Conference on Visualization (EuroVis)-STARs*,vol. 2 (The Eurographics Association, 2015)
15. N. Tremblay, P. Borgnat, Graph wavelets for multiscale community mining. IEEE Trans. Signal Process. **62**(20), 5227–5239 (2014)
16. P. Valdivia, F. Dias, F. Petronetto, C.T. Silva, L.G. Nonato, Wavelet-based visualization of time-varying data on graphs, in *IEEE Conference on Visual Analytics Science and Technology* (IEEE, 2015)
17. A. Dal Col, P. Valdivia, F. Petronetto, F. Dias, C.T. Silva, L.G. Nonato, Wavelet-based visual analysis for data exploration. Comput. Sci. Eng. **19**(5), 85–91 (2017)
18. D. Gleich, The matlabbgl matlab library (2015)
19. D.I. Shuman, C. Wiesmeyr, N. Holighaus, P. Vandergheynst, Spectrum-adapted tight graph wavelet and vertex-frequency frames. IEEE Trans. Signal Process. **63**(16), 4223–4235 (2015)
20. X. Dong, D. Thanou, P. Frossard, P. Vandergheynst, Learning laplacian matrix in smooth graph signal representations. IEEE Trans. Signal Process. **64**(23), 6160–6173 (2016)
21. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**(2), 129–150 (2011)
22. T. Biyikoğu, J. Leydold, P.F. Stadler, *Laplacian Eigenvectors of Graphs* (Springer, Heidelberg, 2007)
23. T.M.J. Fruchterman, E.M. Reingold, Graph drawing by force-directed placement. Softw.: Pract. Exp. **21**(11), 1129–1164 (1991)
24. Julie Fournet, Alain Barrat, Contact patterns among high school students. PLoS ONE **9**(9), e107878 (2014)

Graph-Based Wavelet Multiresolution Modeling of Multivariate Terrain Data



Teodor Cioacă, Bogdan Dumitrescu and Mihai-Sorin Stupariu

Abstract Terrain data pose modeling challenges due to their high inherent redundancy and difficulty of identifying features at different levels of detail. We propose a multiresolution analysis framework based on a graph-based wavelet construction. Our approach produces a sequence of intermediate resolution approximations of the terrain model. The details pertaining to each resolution reveal scale-specific features. Using a guiding heuristic, the proposed wavelet construction also conserves salient features. Furthermore, the proposed framework allows both geometric and attribute vertex information and can be used for modeling tasks sharing the same characteristics and constraints with terrain modeling. In particular, our graph-based wavelet framework is an option for multiresolution filtering and feature classification or clustering.

1 Introduction

The particular problematics of terrain modeling span a broad range of quantitative and qualitative analysis tasks. Such tasks include, but are not limited to geometric representation, geomorphological feature identification and analysis, vegetation coverage analysis and extraction. The continuous advancements of 3D sensor scanning technologies and the increasing popularity of civil aerial drones further facilitate gathering high density point cloud terrain representations.

In this chapter, we describe an approach for multiresolution analysis of multivariate terrain data, which is also suitable for other data organized in

T. Cioacă (✉) · B. Dumitrescu
University Politehnica of Bucharest, Bucharest, Romania
e-mail: t.cioaca@gmail.com

B. Dumitrescu
e-mail: bogdan.dumitrescu@acse.pub.ro

M.-S. Stupariu · T. Cioacă
University of Bucharest, Bucharest, Romania
e-mail: stupariu@fmi.unibuc.ro

© Springer Nature Switzerland AG 2019

L. Stanković and E. Sejdić (eds.), *Vertex-Frequency Analysis of Graph Signals*,
Signals and Communication Technology,
https://doi.org/10.1007/978-3-030-03574-7_15

similar structures. In general, the framework we propose can be easily adapted for performing multiresolution analysis on triangulated point clouds, meshes or graphs. Any of these data structures may describe an irregular domain where a multivariate signal is discretized. The samples are then concentrated in the vertices of these structures.

Given these aspects, the approach we will further describe capitalizes on recently developed methods for graph signal processing. More specifically, we explore a wavelet-based graph multiresolution construction that has a reduced approximation error and is computationally efficient. The graph wavelet interpretation facilitates tasks involving *vertex frequency analysis* or *multiresolution filtering and clustering*.

1.1 Terrain Data Representation

Typically, the geometric information describing terrain fragments is encoded in a digital elevation model (DEM) [36]. A triangulated irregular network (TIN) is a further specialization of a DEM where both point and primitive (triangle) sampling are allowed to be irregular, reducing the highly redundant information in flat regions. De Floriani and Magillo [13] provide the following definition:

> A terrain can be mathematically modeled as a function $z = f(x, y)$ mapping a point (x, y) in a domain D in the plane to its elevation value $f(x, y)$. In practice, the value of function f is known on a finite set S of points within D. A DEM provides an estimated value for function f at any point (x, y) of the domain, based on the values at the points of S. A DEM consists of a subdivision of the domain into cells and of a piece-wise interpolating function defined on such cells. A TIN is a DEM in which the domain subdivision is a triangular mesh, i.e., a set T of triangles such that:
>
> 1. the set of vertices of T is S
> 2. the interiors of any two triangles of T do not intersect
> 3. if the boundaries of two triangles intersect, then the intersection is either a common vertex, or a common edge.

We can easily conclude that the TIN definition is practically identical to that of a triangular mesh as generally understood in the field of Computer Graphics.

1.2 Challenges of Terrain Modeling From Raw Point Data

LiDAR data are usually dense and the information richness can pose a problem on its own since there is no intrinsic manner of reducing geometric or other attribute redundancy. However, model simplification is a well studied problem for which established solutions have been proposed in the field of Computer Graphics. Among the computationally efficient approaches, we mention *incremental thinning*, where geometric primitives are simply removed by following a local quality-based simplification criterion. For TIN models, Demaret et al. [14] have proposed an efficient coarsification algorithm via incremental point removal.

Suarez and Plaza [34] have introduced a coarsening algorithm applicable to right-triangulated irregular networks. Their method is especially efficient given that both the underlying geometry and triangle primitive connectivity is easier to control than in the case of general irregular networks as each vertex has a maximum of 8 directly connected neighbors. The actual coarsening is then achieved through edge midpoint split operations, which are also invertible.

Another problem posed by raw point clouds is that there is no immediate and accurate information concerning the ground or vegetation class of constituent points. While ground return information can help define a subset of soil-level points, the overall point cloud usually contains more ground-level information, as well as potentially unclassified data points. Evans and Hudak [15] have designed a *multiscale curvature classification algorithm* in order to filter and classify points present in raw LiDAR collections.

Silva et al. [2] examine four different algorithms for classifying ground from hovering points in point cloud terrain data sets. Among the algorithms discussed in their work, the progressive triangulated irregular network, implemented in the LAStools package [20], was proven to possess good performance characteristics (low RMSE and increased ground classification accuracy with respect to the ground truth).

1.3 Existing Solutions

Given that terrain modeling tasks pose similar problems to those encountered in point cloud, mesh or graph data processing, we will briefly review some of the milestone achievements from these areas.

Incremental Simplification of Point Clouds and Meshes

Historically, solutions to feature redundancy have been addressed separately from multiresolution analysis. The most common solution to this problem is incremental simplification. Modeling of terrain data is also possible with either point cloud or mesh simplification methods.

For point clouds, notable contributions are given in the work of Pauly et al. [29] where three simplification method categories are detailed: clustering, iterative simplification and particle simulation.

In the case of triangular or polygonal meshes, the additional information also facilitates the creation of reversible, continuous level-of-detail representations and mesh morphing. Schroeder et al. [32] were among the first authors to discuss triangular mesh simplification in a step-by-step manner known as *decimation*. Almost all decimation-based methods require the existence of a decimation criterion and can be summarized in two steps that are repeated until the target density is reached: *(i)* evaluate decimation criterion (usually a cost function estimating the impact of removing a set of primitives) and *(ii)* remove a primitive and re-triangulate the local geometry.

To address changes in the mesh connectivity resulting from primitive removal, Ronfard et al. [31] proposed a reversible operation called *edge collapse*. One of the most effective heuristic criteria for guiding these operations is the *quadric error metric (QEM)* formulation of Garland and Heckbert [17].

Wavelet Multiresolution Methods

Successful initiatives for adapting established Signal Processing methods for point cloud, mesh and graph analysis have been recorded.

To briefly summarizes notable adaptations of wavelet-like transforms on point clouds we mention the following works. Chen et al. [5] have introduced an SVD-based construction of so-called geometric wavelet point cloud analysis. Earlier, Choi et al. [6] applied a straightforward implementation of Swelden's *lifting scheme* [35] for denoising point sampled manifolds of low intrinsic dimension.

Among the first wavelet mesh multiresolution analysis methods is that of Lounsbery [26]. Using a subdivision operator on triangular meshes, Lounsbery introduced an analog of the refinement equation for functions defined on mesh structures instead on the real line. With the introduction of Swelden's *lifting scheme* [35], other approaches using this technique were developed. One example is the critically sampled design for irregular meshes described by Bertram [3]. His method is similar to that of Guskov et al. [18], where a geometry-based subdivision operator defines wavelet synthesis.

After mesh wavelet constructions, algorithms specifically design for more generic graph data were also introduced. We mention the method of Coifman et al. [12], which uses a diffusion operator to build a wavelet transform in the graph frequency domain. Hammond et al. [19] further proposed a method that, instead of diagonalizing the graph Laplacian matrix, a costly process for large graphs, uses approximations based on Chebyshev polynomials. In the graph spatial domain we mention Jansen's approach [21, 22], also adopted and extended by Wagner et al. [37], and by Martinez and Ortega [27].

Wavelet Multiresolution for Terrain Modeling

Among the first solutions specifically designed with terrain modeling in mind is the *wavelet triangular irregular networks* framework of Wu and Amaratunga [24]. This method is essentially a *lifting scheme* adaptation using an inverse subdivision prediction filter and resampling in the xy plane. The importance of wavelet coefficients, byproducts of multiresolution analysis, was emphasized by Beyer [4]. The author demonstrated that both terrain gradient and curvature information can be derived from these wavelet detail vectors. Kalbermatten's et al. [25] work further explores applications of wavelet transform in identifying multiscale geomorphological and geological features.

2 Terrain Modeling Using Graph-Based Wavelet Multiresolution Analysis

We now address the question of how a wavelet multiresolution framework can be defined for terrain models. From a signal processing perspective, DEMs are similar to images, which are simply two-dimensional signals defined over regular domains. When modeling data from LiDAR sources, TIN structures are better options because they allow for adaptive sampling in areas with high geometric redundancy. Building wavelets on these irregular domains would enable the analysis of terrain characteristics at different levels of resolution, which, in turn, can be perceived as information pertaining to a certain frequency band.

Our goal is threefold. First, we aim to expand previous methods and achieve multiresolution representation of multivariate terrain sets (having both geometric and attribute coordinates for each point). Second, by adopting a spatial domain graph lifting scheme construction, we maintain a balance between computational complexity and the range of potential practical applications of the method. Third, we wish to explore the semantic interpretation of detail vectors, leading to a correlation between vertices and levels of resolution. This interpretation would allow our method to be applicable to *vertex frequency analysis* tasks.

Both the construction and results presented in the following sections of this chapter have been partially described in our previous works [7, 9].

2.1 Classic Wavelet Analysis and Synthesis

Before detailing our wavelet multiresolution design for multivariate signals, we review the fundamental concepts in the one-dimensional case.

Problem Formulation

In general, wavelets are the building blocks of hierarchical function space approximations, i.e. $L^2(\mathbb{R}) \succ \ldots \succ V_n \succ V_{n-1} \succ \ldots \succ V_1 \succ V_0 \succ \ldots \succ \{0\}$, where V_j is a function space approximation at level j. We are interested in examining the connections between two consecutive approximations, V_{j+1} and V_j. Let $\Phi_j = \begin{bmatrix} \ldots & \phi_{j,k} & \ldots \end{bmatrix}$ be the row vector of basis functions that generate V_j. Then a function $f_{j+1} \in V_{j+1}$ cannot be represented by using only the basis functions in V_j, but it can be expressed as a combination of the basis functions of the V_{j+1} space, that is

$$f_{j+1} = \Phi_{j+1}\mathbf{s}_{j+1}, \tag{1}$$

where $\mathbf{s}_{j+1}$ is an infinite column vector of *scaling coefficients*. The complementary basis or *wavelet functions*, $\psi_{j,k}$, that generate the orthogonal subspace, correspond to the lost details W_j. Thus, the *direct sum decomposition* of the higher-resolution

approximation, $V_{j+1} = V_j \oplus W_j$, can be expressed in terms of basis and wavelet functions as

$$f_{j+1} = \Phi_{j+1}\mathbf{s}_{j+1} = \Phi_j \mathbf{s}_j + \Psi_j \mathbf{w}_j, \tag{2}$$

where $\Psi_j = \left[\ldots \psi_{j,k} \ldots\right]$ and $\mathbf{w}_j$ is the infinite column vector of *wavelet coefficients*.

Both the scaling and wavelet coefficients can be written in terms of internal products between f_{j+1} and the dual basis functions of their corresponding spaces, i.e.:

$$\mathbf{s}_{j+1,k} = \left\langle \tilde{\phi}_{j+1,k}, f_{j+1} \right\rangle, \tag{3}$$

$$\mathbf{s}_{j,k} = \left\langle \tilde{\phi}_{j,k}, f_{j+1} \right\rangle, \tag{4}$$

$$\mathbf{w}_{j,k} = \left\langle \tilde{\psi}_{j,k}, f_{j+1} \right\rangle, \tag{5}$$

where $\tilde{\phi}_{j,k}$ and $\tilde{\psi}_{j,k}$ denote the k-th scaling and wavelet basis functions at level j.

Second Generation Wavelets

Let $\mathbf{x} = (x_k)_{k\in\mathbb{Z}}$ be an infinite sampled signal.

The first operation involved in the lifting procedure is called a *split* or *parity assignment*. To this effect, the signal is divided into two sets, an even set, corresponding to $\mathbf{x}_e = (x_{2k})_{k\in\mathbb{Z}}$ and an odd set, corresponding to $\mathbf{x}_o = (x_{2k+1})_{k\in\mathbb{Z}}$.

The second operation of the lifting procedure is the *prediction step*. This implies approximating the odd samples using the even ones by applying a prediction operator, $\mathbf{P}$, i.e.

$$\mathbf{d} = \mathbf{x}_o - \mathbf{P}\mathbf{x}_e, \tag{6}$$

obtaining the signal $\mathbf{d}$ of *approximation errors* or *detail coefficients*.

Usually, the even and odd sample subsets are highly correlated, and, as a direct consequence, it becomes more efficient to store the signal using the information in $\mathbf{d}$ as it has lower entropy than $\mathbf{x}_o$. The prediction phase is equivalent to mapping $(\mathbf{x}_e, \mathbf{x}_o) \rightarrow (\mathbf{x}_e, \mathbf{d})$. Since $\mathbf{x}_e$ is essentially obtained through a naïve downsampling of the original signal, aliasing occurs and must be appropriately dealt with. This is performed through a third operation, the *update* or *smoothing* phase. Algebraically, this operation is implemented as

$$\mathbf{s} = \mathbf{x}_e + \mathbf{U}\mathbf{d}, \tag{7}$$

where $\mathbf{s}$ is the smoothed downsampled signal and $\mathbf{U}$ is the update operator.

Assembling the *three stages* of the lifting scheme into a sequence, we obtain the flow diagram shown in Fig. 1. The flow allows cascading the resulting filter bank by feeding the even output from one lifting pass to the input of another one, effectively creating a series of coarser approximations of the original signal.

Another immediate property of the lifting design is the straightforward invertibility, i.e.

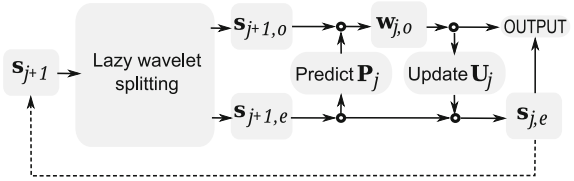


Fig. 1 The *lifting scheme* diagram: Splitting (Eq. (10)) is followed by prediction via Eq. (11) and then by the updating of the low resolution output via Eq. (12)

$$\mathbf{x}_e = \mathbf{s} - \mathbf{U}\mathbf{d} \tag{8}$$

$$\mathbf{x}_o = \mathbf{d} + \mathbf{P}\mathbf{x}_e. \tag{9}$$

Returning to the function space context, we review the mechanism behind the lifting scheme. The first operation, the *split*, when applied to the set of scaling coefficients at level $j+1$ produces a vector of scaling coefficients reindexed such that

$$\mathbf{s}_{j+1} = \begin{bmatrix} \mathbf{s}_{j+1,o} \\ \mathbf{s}_{j+1,e} \end{bmatrix}, \tag{10}$$

where the o and e subscripts stand for *odd* and *even* coefficients, respectively.

We write the *prediction* equation as

$$\mathbf{w}_j = \mathbf{s}_{j+1,o} - \mathbf{P}_j \mathbf{s}_{j+1,e}, \tag{11}$$

and the *update* as

$$\mathbf{s}_j = \mathbf{s}_{j+1,e} + \mathbf{U}_j \mathbf{w}_j, \tag{12}$$

with $\mathbf{P}_j \in \mathbb{R}^{n_{j+1,o} \times n_{j+1,e}}$ being a sparse prediction matrix, $\mathbf{U}_j \in \mathbb{R}^{n_{j+1,e} \times n_{j+1,o}}$ being a sparse update matrix and $n_{j+1,o}$ and $n_{j+1,e}$ being the number of odd and even coefficients, respectively, at level $j+1$. The prediction stage simply exploits the signal redundancy by assuming that the odd coefficients can be estimated as linear combinations of their spatial neighbors. If only the even coefficients are used to approximate the functions in V_{j+1}, then the lost details are compensated for by redistributing them among these remaining coefficients via the update matrix.

Since Eqs. (3)–(5) hold for any f_{j+1}, we can write the predict and update equations for the duals basis functions:

$$\tilde{\Psi}_j^{\mathsf{T}} = \tilde{\Phi}_{j+1,o}^{\mathsf{T}} - \mathbf{P}_j \tilde{\Phi}_{j+1,e}^{\mathsf{T}}, \tag{13}$$

$$\tilde{\Phi}_j^{\mathsf{T}} = \tilde{\Phi}_{j+1,e}^{\mathsf{T}} + \mathbf{U}_j \tilde{\Psi}_j^{\mathsf{T}} = (\mathbf{I} - \mathbf{U}_j \mathbf{P}_j) \tilde{\Phi}_{j+1,e}^{\mathsf{T}} + \mathbf{U}_j \tilde{\Phi}_{j+1,o}^{\mathsf{T}}. \tag{14}$$

To derive similar relations between the primal basis functions, we can start by rewriting Eq. (2) as

$$\left[\Phi_{j+1,o}\ \Phi_{j+1,e}\right]\begin{bmatrix}\mathbf{s}_{j+1,o}\\ \mathbf{s}_{j+1,e}\end{bmatrix}=\left[\Psi_j\ \Phi_j\right]\begin{bmatrix}\mathbf{w}_j\\ \mathbf{s}_j\end{bmatrix}, \tag{15}$$

and, expanding the scaling and wavelet coefficients on the right-hand side using Eqs. (11) and (12), we arrive to

$$\left[\Phi_{j+1,o}\ \Phi_{j+1,e}\right]\begin{bmatrix}\mathbf{s}_{j+1,o}\\ \mathbf{s}_{j+1,e}\end{bmatrix}=\left[\Psi_j\ \Phi_j\right]\begin{bmatrix}\mathbf{s}_{j+1,o}-\mathbf{P}_j\mathbf{s}_{j+1,e}\\ (\mathbf{I}-\mathbf{U}_j\mathbf{P}_j)\mathbf{s}_{j+1,e}+\mathbf{U}_j\mathbf{s}_{j+1,o}\end{bmatrix}. \tag{16}$$

Equation (16) must hold for any combination of the level $j+1$ lifting coefficients. Hence, let $\mathbf{s}_{j+1,e}=\boldsymbol{\delta}_k$, i.e. the Kronecker unit vector at index k, with $k\in 1:n_e$, i.e. $\delta_{k,i}=0, \forall k\neq i$ and $\delta_{k,k}=1$. Also, let $\mathbf{s}_{j+1,o}=\mathbf{0}$. Evaluating both sides of Eq. (16), we arrive to

$$\Phi_{j+1,e}\delta_k=-\Psi_j\mathbf{P}_j\boldsymbol{\delta}_{\boldsymbol{k}}+\Phi_j\boldsymbol{\delta}_{\boldsymbol{k}}-\Phi_j\mathbf{U}_j\mathbf{P}_j\boldsymbol{\delta}_{\boldsymbol{k}}, \tag{17}$$

or, since this equation holds for any $k\in 1:n_e$, a more direct formulation can be written as

$$\Phi_{j+1,e}=-\Psi_j\mathbf{P}_j+\Phi_j-\Phi_j\mathbf{U}_j\mathbf{P}_j. \tag{18}$$

By setting $\mathbf{s}_{j+1,e}=\mathbf{0}$ and $\mathbf{s}_{j+1,o}=\delta_k$, with $k\in 1:n_o$, we find that

$$\Phi_{j+1,o}\delta_k=\Psi_j\delta_k+\Phi_j\mathbf{U}_j\delta_k, \tag{19}$$

or

$$\Phi_{j+1,o}=\Psi_j+\Phi_j\mathbf{U}_j. \tag{20}$$

Right-multiplying both sides of Eq. (20) by $\mathbf{P}_j$ and adding the result to Eq. (18) we obtain:

$$\Phi_j=\Phi_{j+1,e}+\Phi_{j+1,o}\mathbf{P}_j, \tag{21}$$

and then

$$\Psi_j=\Phi_{j+1,o}(\mathbf{I}-\mathbf{P}_j\mathbf{U}_j)-\Phi_{j+1,e}\mathbf{U}_j. \tag{22}$$

Let $\varsigma_j=\int_{-\infty}^{\infty}\Phi_j^{\mathsf{T}}(t)dt$. Then integrating equation (21) yields:

$$\varsigma_j=\varsigma_{j+1,e}+\mathbf{P}_j^{\mathsf{T}}\varsigma_{j+1,o}. \tag{23}$$

Since the integral of the wavelet functions is zero, integrating equation (22) leads to:

$$\mathbf{0}=\varsigma_{j+1,o}-\mathbf{U}_j^{\mathsf{T}}\left(\mathbf{P}_j^{\mathsf{T}}\varsigma_{j+1,o}+\varsigma_{j+1,e}\right)=\varsigma_{j+1,o}-\mathbf{U}_j^{\mathsf{T}}\varsigma_j. \tag{24}$$

The column vectors of $\mathbf{U}_j$ can be retrieved in a one-by-one fashion from Eq. (24). If $\mathbf{u}_{j_k}$ is the kth column vector, then

$$\varsigma_{j+1,o_k} = \mathbf{u}_{j_k}^{\mathsf{T}} \varsigma_{j_k}. \tag{25}$$

It is suggested in [21] to choose the minimum norm solution of this equation as it leads to increased numerical stability, as experimentally shown in [9]. It results that

$$\mathbf{u}_{j_k} = \varsigma_{j+1,o_k} \frac{\varsigma_{j_k}}{\|\varsigma_{j_k}\|^2}. \tag{26}$$

Wavelet Construction for Multivariate Graph Signals

We examine now the changes necessary to go from the above one-dimensional construction to multivariate signals defined on graphs. In general, we are interested in analyzing a function $F(\mathbf{v})$ defined at every vertex $\mathbf{v}$ of the graph. We regard the vertices as having vector coordinates in $\mathbb{R}^n$. The purpose is to find appropriate scaling and wavelet function bases that are independent of the multivariate signal itself, but depend on the graph topology and its edge lengths. Thus, the framework construction we propose must also accommodate both univariate and multivariate functions by treating the latter as tuples of scalar-valued functions. Instead of discussing about time series indices, we now view the graph nodes as means to identify samples where the information is concentrated. The odd-even split is extended to graph vertices, denoting V_{o_l} and V_{e_l} the odd and even subsets at level l, respectively. The notion of parity is irrelevant for graphs, the odd and even denominations serving a labeling purpose. If $\mathbf{v}_{i_l} \in V_{e_l}$ denotes a vertex from the approximation set at level l, we refer to its corresponding scaling vector of coefficients by using the $\mathbf{s}_{l,\mathbf{v}_{i_l}}$ notation. At the highest level of resolution, L, we assimilate the scaling vector to the vertex coordinates, i.e. $\mathbf{s}_{L,\mathbf{v}_{i_L}} := \mathbf{v}_{i_L}^{\mathsf{T}}$. We adopt the same convention for denoting the scaling functions, i.e. $\phi_{l,\mathbf{v}_{i_l}}$, which are scalar valued. The situation is identical for the detail components, which correspond to the odd samples, $\mathbf{v}_{j_l} \in V_{o_l}$, at level l. The detail vector associated with $\mathbf{v}_{j_l}$ will be denoted by $\mathbf{w}_{l,\mathbf{v}_{j_l}}$, while for the wavelet functions the $\psi_{l,\mathbf{v}_{j_l}}$ notation will be adopted. Using these conventions, we can express the equivalent *multiresolution decomposition equation for graphs* as

$$F(\mathbf{v}) = \sum_{l \geq 0} \sum_{\mathbf{v}_o \in V_{o_l}} \mathbf{w}_{l,\mathbf{v}_o} \psi_{l,\mathbf{v}_o}(\mathbf{v}) + \sum_{\mathbf{v}_e \in V_{e_l}} \mathbf{s}_{0,\mathbf{v}_e} \phi_{0,\mathbf{v}_e}(\mathbf{v}). \tag{27}$$

By arranging the scaling and wavelet functions into row vectors, we reproduce Eqs. (21) and (22), establishing the relationship at consecutive resolution levels through the means of prediction and update filters, i.e.

$$\Phi_{l,V_{e_l}} = \Phi_{l+1,V_{e_{l+1}}} + \Phi_{l+1,V_{o_{l+1}}} \mathbf{P}_l, \tag{28}$$

$$\Psi_{l,V_{o_{l+1}}} = \Phi_{l+1,V_{o_{l+1}}} - \Phi_{l,V_{e_l}} \mathbf{U}_l, \tag{29}$$

where $\mathbf{P}_l$ is the $|V_{o_{l+1}}| \times |V_{e_{l+1}}|$ prediction filter matrix and $\mathbf{U}_l$ is the $|V_{e_{l+1}}| \times |V_{o_{l+1}}|$ update filter matrix. Using the above filter matrices, a hierarchical decomposition

of the scaling coefficients associated with the vertices of a mesh or a graph can be inferred. These vector-valued coefficients contain both the geometric and attribute coordinates of their corresponding vertices. Let us denote by $\mathbf{s}_{l,V_{e_l}}$ the $|V_{e_l}| \times n$ matrix of scaling coefficients at level l associated with the even nodes, V_{e_l}, the difference vectors computation and even node updates can be written as

$$\mathbf{w}_{l,V_{o_{l+1}}} = \mathbf{s}_{l+1,V_{o_{l+1}}} - \mathbf{P}_l \mathbf{s}_{l+1,V_{e_{l+1}}}, \tag{30}$$

$$\mathbf{s}_{l,V_{e_l}} = \mathbf{s}_{l+1,V_{e_{l+1}}} + \mathbf{U}_l \mathbf{w}_{l,V_{o_{l+1}}}. \tag{31}$$

Equations (30) and (31) describe the analysis stage of critically sampled lifting scheme. It is straightforward to invert this process. This converse operation, the synthesis stage, is translated into the following two equations:

$$\mathbf{s}_{l+1,V_{e_{l+1}}} = \mathbf{s}_{l,V_{e_l}} - \mathbf{U}_l \mathbf{w}_{l,V_{o_{l+1}}}, \tag{32}$$

$$\mathbf{s}_{l+1,V_{o_{l+1}}} = \mathbf{w}_{l,V_{o_{l+1}}} + \mathbf{P}_l \mathbf{s}_{l+1,V_{e_{l+1}}}. \tag{33}$$

Cascading the analysis stages yields a hierarchical wavelet decomposition of the initial mesh. The method stores the intermediary difference vectors $\mathbf{w}_{l,V_{o_{l+1}}}$ and the coarsest level scaling coefficients, $\mathbf{s}_{0,V_{e_0}}$, with $l \in 0 : L$. In order to recover the initial information, the intermediary filter matrices, $\mathbf{P}_l$ and $\mathbf{U}_l$, also need to be stored.

2.2 Graph-Based Lifting Scheme Operations

We now proceed to describing the necessary steps for adapting the lifting scheme principles to the irregular graph domain. In doing this, we consider a mechanism for guiding the lazy wavelet partitioning such as salient features loss is reduced during downsampling. We also aim to develop prediction and update filters that minimize the approximation error at lower levels of resolution.

Lazy Wavelet Partitioning

Our choice of a heuristic feature preservation mechanism is the *generalized quadric error metric*, detailed by Garland and Heckbert [17]. The goal of this metric is to facilitate computing the squared distances from any point to the support plane of a triangle. For the remainder of this discussion, we will refer to vertices and their attributes as elements from the $\mathbb{R}^n$ vector space, where the first three components are the geometric coordinates and the remaining $n - 3$ represent the attribute data.

Let $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{R}^n$ be column vectors corresponding to the vertices of a triangle and $\mathbf{p} \in \mathbb{R}^n$ an arbitrary point. The core idea of this approach is to algebraically express as a matrix, denoted by $\mathbf{Q}(\Delta(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3))$, the computation of the squared distance from $\mathbf{p}$ to the support plane of these 3 vertices. It is then easy to compute

the sum of squared distances from $\mathbf{p}$ to the support planes of a triangle family, $(\Delta_i(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3))_{i=1:N}$, as

$$\sum_{i=1:N} d(\mathbf{p}, \Delta_i(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3))^2 = \begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix}^{\mathsf{T}} \left\{ \sum_{i=1:N} \mathbf{Q}\left(\Delta_i(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)\right) \right\} \begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix}. \tag{34}$$

In the original incremental simplification algorithm [17], the set of faces in the one-ring neighborhood of each vertex is used to compute an associated matrix

$$\mathbf{Q}(\mathbf{v}) = \sum_{\Delta_k \in \mathcal{N}_t^1(\mathbf{v})} \mathbf{Q}(\Delta_k), \tag{35}$$

where $\mathcal{N}_t^1(\mathbf{v})$ represents the set of all triangles incident at $\mathbf{v}$.

The advantage of using the matrix notation is manifested when performing edge collapses and fusing the endpoint vertices. Whenever two vertices, $\mathbf{v}_a$ and $\mathbf{v}_b$, are replaced by a new vertex, $\mathbf{w}$, the local geometric information that was characterized by the neighborhoods of these vertices is preserved by setting $\mathbf{Q}(\mathbf{w}) \leftarrow \mathbf{Q}(\mathbf{v}_a) + \mathbf{Q}(\mathbf{v}_b)$. This way, although the mesh is coarser, the new vertex still retains the local geometric variability of the initial model. Thus, the history of collapses is added together and represented as a single quadric matrix.

The matrix terms in Eq. (35) describe quadrics in the sense that all isosurfaces obtained from varying point $\mathbf{p}$ in Eq. (34) are quadrics. The term *quadric error metric* is thus justified since these matrices offer a means of estimating an error measure from an arbitrary position $\mathbf{p}$ to a local patch around any vertex $\mathbf{v}$. As described in [7], this metric also allows for the introduction of a cost function associated to each vertex:

$$\texttt{cost}(\mathbf{v}) = \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}^{\mathsf{T}} \left(\sum_{\mathbf{v}_i \in \mathcal{N}_v^1(\mathbf{v})} \mathbf{Q}(\mathbf{v}_i) \right) \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}, \tag{36}$$

where $\mathcal{N}_v^1(\mathbf{v})$ denotes the direct neighbors of $\mathbf{v}$, or the *one-ring* vertex set, as represented in Fig. 2.

In [7] we introduced an additional saliency measurement attribute and treat the vertices of the model as points in $\mathbb{R}^{n+1}$. We opt for a discrete bending energy (or thin plate energy) estimation since it encompasses curvature and area information and it is also an isometric invariant.

In the continuous case, the bending energy is a well-defined quantity. If the principal curvatures can be computed over a surface patch A, then the amount of elastic potential energy stored in that region can be computed by evaluating the integral

$$E_b = \int_A \left(\kappa_1^2 + \kappa_2^2\right) dA, \tag{37}$$

where κ_1 and κ_2 are the principal curvature functions defined over the patch A.

Algorithm 1 Thin plate energy computation

```
INPUT: the mesh M = (V, E)
OUTPUT: the bending energy of the vertex set, E_bending(V)
for v_i ∈ V do
  COMPUTE: the Gaussian and mean curvatures of v_i, K(v_i) and H(v_i)
  COMPUTE: f(v_i) = 4H(v_i)^2 − 2K(v_i)
end for
for v_i ∈ V do
  E_bending(v_i) = 0
  for Δ ∈ N_f^1(v_i) do
    COMPUTE: E_bending(v_i) += ∫_Δ f dA
  end for
end for
```

The evaluation of the discrete Gaussian curvature, denoted as $K(\mathbf{v})$, and of the mean curvature, denoted as $H(\mathbf{v})$, can be performed as suggested by [28]. Although such estimates are known to be sensitive to noise, the data typically resulting from LiDAR sets do not exhibit irregularities that could affect the robustness. Regardless of this drawback, a curvature-based energy represents a natural measure for local geometric saliency over a one-ring neighborhood. In [9, 10] we further discuss potential alternatives for use on data heavily affected by noise.

To evaluate the discrete version of integral (37), we can use the Gaussian curvature, K, and the mean curvature, H, to compute the sum of squared principal curvatures as $\kappa_1^2 + \kappa_2^2 = 4H^2 - 2K$. The discrete counterpart of this integral then provides an estimate for the discrete bending energy concentrated at each vertex $\mathbf{v}_i$ and is computed over its one-ring neighborhood. We evaluate the discretized bending energy near a vertex $\mathbf{v}_i$ as

$$E_b(\mathbf{v}_i) = \sum_{\Delta(\mathbf{v}_i,\mathbf{v}_j,\mathbf{v}_k)\in\mathcal{N}_f^1(\mathbf{v}_i)} \frac{f(\mathbf{v}_i) + f(\mathbf{v}_j) + f(\mathbf{v}_k)}{6} \cdot \left\| (\mathbf{v}_j - \mathbf{v}_i) \times (\mathbf{v}_k - \mathbf{v}_i) \right\|, \tag{38}$$

where $f \equiv (\kappa_1^2 + \kappa_2^2)$ and $\mathcal{N}_f^1(\mathbf{v}_i)$ is the set of all incident triangles, denoted by $\Delta(\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k)$, at $\mathbf{v}_i$.

We summarize the calculation process of the discrete version of Eq. (37) in Algorithm 1.

To include the computed bending energy in the cost function from Eq. (36), one can substitute a vertex $\mathbf{v}$ with an $(n+1)$-dimensional one, $\bar{\mathbf{v}} = \begin{pmatrix} \mathbf{v} \\ E_{bending}(\mathbf{v}) \end{pmatrix}$.

The purpose of computing the per-vertex cost values is to establish an importance-based ordering of this set. The greedy strategy we employ to label all vertices as either even or odd is summarized in Algorithm 2. The labeling process is iterative and marks the vertices intuitively, according to their computed importance. During each iteration, the vertex having the lowest cost is extracted from the set of unmarked vertices. Its one-ring neighbors are then colored as even, while the extracted sample

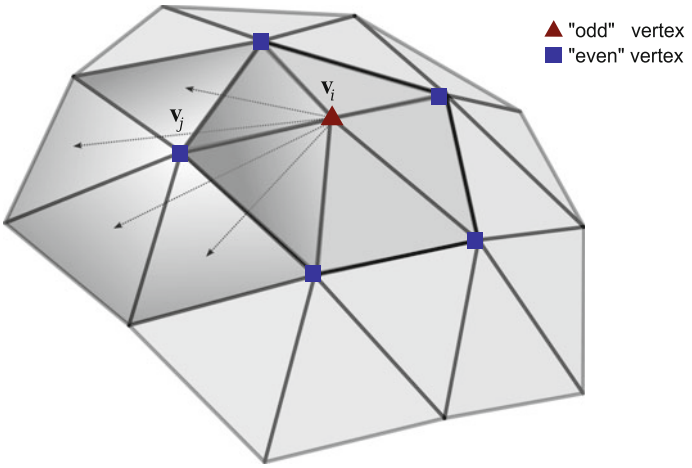


Fig. 2 The cost of removing $\mathbf{v}_i$ with respect to its one-ring neighbors. Each $\mathbf{v}_j \in \mathcal{N}_v^1(\mathbf{v}_i)$ contributes the sum of distances from $\mathbf{v}_i$ to the support planes of each incident face at $\mathbf{v}_j$

Algorithm 2 Vertex labeling and remeshing algorithm

```
INPUT: the mesh M_in = (V, E_in)
OUTPUT: an odd-even partitioning, V = V_o ∪ V_e, coarse mesh topology M_out = (V_e, E_out)
COMPUTE: for each v ∈ V, compute E_bending(v) via Algorithm 1
COMPUTE: for each v ∈ V, compute cost(v̄) as defined in (36), where v̄ = (v^T, E_bending(v))^T
SORT: V* = sort(V + in) using cost(v̄) as a key
ASSIGN: V_e = ∅, V_o = ∅, E_out = E_in
while V* ≠ ∅ do
  v = arg min_{v∈V*} (cost(v̄))
  if can_triangulate(N_v^1(v) \ {v}) then
    for v_i ∈ N_v^1(v) do
      E_out = E_out \ {v̄, v̄_i}
    end for
    E_out = E_out ∪ create_edges(N_v^1(v))
    V_o = V_o ∪ {v}
    V_e = V_e ∪ N_v^1(v)
    V* = V* \ ({v} ∪ N_v^1(v))
  else
    V_e = V_e ∪ {v}
    V* = V* \ {v}
  end if
end while
```

is marked as odd, as depicted in Fig. 2. The intuition behind the process reflects the goal of removing less relevant samples, from high redundancy areas.

Analyzing the vertex classification and remeshing performed by Algorithm 2, we notice the constraint of marking the neighbors of a removed odd node as even. This is justified by the need to consistently separate the two vertex classes, a practice

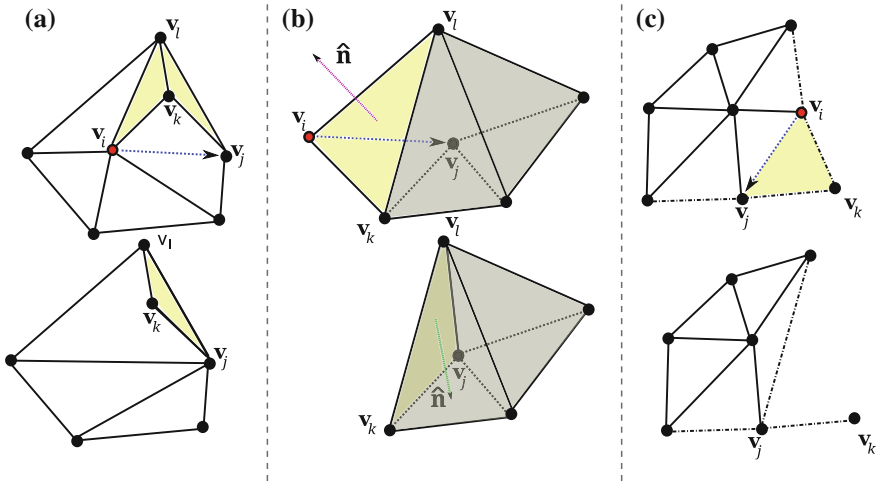


Fig. 3 Illegal collapse situations where the highlighted faces are involved in a breaching of one or several criteria of the `can_triangulate` function from Algorithm 2. In case **a**, the collapse "welds" together two faces, back-to-back, introducing an edge common to more than two triangles. In situation **b**, the highlighted face is "flipped" since the normal unit vectors before and after the collapse point in opposite directions. "Cutting a corner" is the operation depicted in situation **c**, where the corner vertex is no longer adjacent to at least two edges

also employed by [18]. Another key property of this algorithm is the partial graph connectivity alteration incurred by the removal of the odd samples. Together with these nodes, their adjacent edges are also removed. Whether or not the one-ring hole bounded by $\mathscr{N}_v^1(\mathbf{v}) \setminus \{\mathbf{v}\}$ can be triangulated is also a key factor in deciding the label of $\mathbf{v}$. The `can_triangulate`$(\cdot)$ function is implemented as a set of geometrical and topological consistency criteria that must be satisfied by a valid triangulation. This problem has numerous solutions, one example being the *splitting plane* method of [32] known to produce more regular aspect ratio triangulations. It is possible for a vertex having the lowest importance score to produce invalid triangulations, and, in this case, the vertex is marked as even. In our implementation, the triangulation of the one-ring hole is easily performed using half-edge collapses. A valid edge is selected from the set of all edges adjacent to the removed odd vertex such that the quadric error measured using the $\mathbf{Q}(\mathbf{v}_e)$ matrix of its even endpoint is the smallest of all other collapsible pairs. We refer to a vertex pair to be collapsible if it is connected through an edge and if by translating the odd vertex over the even one the discrete manifold property of the mesh is not affected and no newly created triangular face is flipped with respect to its original normal vector (refer to Fig. 3 for several examples of illegal collapses). Additionally, to prevent boundary shrinkage, the odd boundary vertices cannot be translated over interior even ones. Boundary vertices can be part of a collapse if the edge connecting them is a boundary one as well. We also allow for merging an interior vertex with a boundary one when the collapse direction is from the interior towards the border.

Prediction Filter Construction

The goal of the prediction operation that follows the even-odd splitting is to compute estimates for the odd nodes of the graph from their even neighboring nodes. Intuitively, if the predicted values are closer to the actual odd samples, it is possible to recover a faithful representation of the entire graph by storing only a subset of the initial data. The resulting estimates are usually obtained by applying a low-pass filter to the even subset, while computing the difference between the estimates and the actual odd samples resembles the behavior of a high-pass filter. As a graph low-pass filter, we propose using a discrete Laplacian operator, given its smoothing and averaging effects. In many applications, Laplacian filters are also used for suppressing local irregularities (both features and noise). Thus, a similar effect could be achieved by removing the ensuing details.

To understand how the prediction filter weights are computed, we recall the concept of a graph Laplacian operator. For a weighted graph, the Laplacian matrix is obtained as the difference between the weighted diagonal degree matrix, $\mathbf{D} = (d_{i,i}) = \sum_j \omega_{i,j}$, and the cost matrix, $\boldsymbol{\Omega} = (\omega_{i,j})$, where $\omega_{i,j}$ are the weights associated with the $(\mathbf{v}_i, \mathbf{v}_j)$ edges. Thus, if $\mathbf{L} = \mathbf{D} - \boldsymbol{\Omega}$, the random-walk Laplacian operator is defined as

$$\mathscr{L}_{rw} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\boldsymbol{\Omega}. \tag{39}$$

The geometric interpretation for the action this operation has on a mesh is a smoothing effect, also achievable through the use of a generalized umbrella operator. An in-depth comparative discussion of various Laplacian discretizations is offered by [38]. Concretely, the extraction of the difference vectors $\mathbf{w}_{l,V_{o_{l+1}}}$ from Eq. (30) is similar to a Laplacian smoothing where the difference between the smoothed vertex and its actual position is stored for later reference. For a single vertex, $\mathbf{v}_n \in V_{o_{l+1}}$, this equation can be rewritten as

$$\mathbf{w}_{l,\mathbf{v}_n} = \mathbf{s}_{l+1,\mathbf{v}_n} - \sum_{\mathbf{v}_m \in \mathscr{N}_v^1(\mathbf{v}_n)} p_{l,\mathbf{v}_n}(\mathbf{v}_m)\mathbf{s}_{l+1,\mathbf{v}_m}, \tag{40}$$

where $p_{l,\mathbf{v}_n}(\mathbf{v}_m)$ is the prediction weight coefficient at level l associated with vertex $\mathbf{v}_n$ and contributed by one of its even, one-ring neighbors, $\mathbf{v}_m$. Depending on the Laplacian discretization, several prediction weight choices are possible. In general, the prediction weights in Eq. (40) are computed as

$$p_{l,\mathbf{v}_n}(\mathbf{v}_m) = \frac{\omega_{n,m}}{\sum_{\mathbf{v}_k \in \mathscr{N}_v^1(\mathbf{v}_n)} \omega_{n,k}}. \tag{41}$$

One of the more popular Laplacian design choices is the *cotangent weights* Laplacian, computed as described by Meyer et al. [28]. Abdul Rahman et al. [1] recommend this Laplacian as a prediction filter for the analysis of free-form surfaces with additional attributes. Although this filter is suitable for smoothing tasks, its weights depend

strictly on the geometric information and not also on any additional attributes. A simple alternative for this operator, that introduces movements in the tangential plane of a vertex, is the Fujiwara or geometric Laplacian [16], with weights defined as

$$\omega_{i,j} = {l_{i,j}}^{-1}, \tag{42}$$

where $l_{i,j}$ is the length of the $(\mathbf{v}_i, \mathbf{v}_j)$ edge. This operator is scale dependent and it preserves the distribution of triangle sizes. The main advantages of this design are: its dependency on all geometry and range attributes, its smoothing effect being closer to the cotangent weight formulation than that of the umbrella operator, and the predicted point being shifted towards its closest neighbors, thus inherently providing a better approximation.

An alternative to the Laplacian prediction filter is to employ a least squares fitting of the weights in order to minimize the approximation error. This approach, adopted in several works [27, 37], proved to be numerically unstable when directly applied to terrain sets. To overcome this issue, Wagner et al. [37] did not include the boundary vertices in the downsampling process. In regions where the one-ring neighborhood does not have a convex (x, y) projection, negative weights can appear. Furthermore, the large magnitude of the weights may lead to numerical instability during the subsequent update stage. To counteract these effects, we propose a non-negative least squares (NNLS) fitting of the weights in Eq. (40), such that the magnitude of the $\mathbf{w}_{l,\mathbf{v}_n}$ vector is minimized. To achieve a similar Laplacian smoothing effect, the sum $\sum_{\mathbf{v}_m \in \mathcal{N}_v^1(\mathbf{v}_n)} p_{l,\mathbf{v}_n}(\mathbf{v}_m)$ should be equal to 1. This constraint can be directly added to the NNLS solver. By design, this modification improves the root mean square error throughout the hierarchical downsampling, as it will be later discussed in the results section. We note that positive and convex weights allow for the removal of odd boundary vertices. Nevertheless, computing these weights incurs a computational penalty, so we only recommend this choice for scenarios where minimizing the approximation error is crucial.

In terms of storage complexity, the Laplacian matrix is very sparse. As a consequence, the prediction matrix at level l, $\mathbf{P}_l$, is also sparse, each of its rows being populated with the weights used to predict the same odd vertex, $\mathbf{v}_n$, from its even neighbors.

Update Filter Construction

The heuristically guided lazy wavelet removal of details minimizes feature loss, but does not propagate detail loss to inferior levels. Wavelet transforms manage this problem by redistributing the extracted detail among the coarser scales. This information is effectively contained in the difference vectors. Without compensating for these losses, the algorithm would mostly resemble incremental simplification. The update filter of the lifting scheme is responsible for distributing the lost details among the even vertices in a way that preserves an aggregate signal property such as signal average. We opt for this choice because it helps maintain both overall shape and decreases the approximation error, as we will later experimentally observe.

Because the prediction filter determines the amount of detail loss, the update filter should express a dependency on the prediction weights. In [27], the authors suggest the following expression for the update filter vectors:

$$\mathbf{u}_{l,\mathbf{v}_u} = \frac{0.5}{\sum\limits_{\mathbf{v}_{p,i} \in \mathcal{N}_v^1(\mathbf{v}_u) \cap V_{o_{l+1}}} p_{l,\mathbf{v}_{p,i}}} \left[p_{l,\mathbf{v}_{p,1}}, p_{l,\mathbf{v}_{p,2}}, \ldots, p_{l,\mathbf{v}_{p,k}} \right], \tag{43}$$

where $\mathbf{v}_u$ represents an even vertex, $\mathbf{v}_{p,i}$ denotes an odd one-ring neighbor of this vertex, and $p_{l,\mathbf{v}_{p,i}} \equiv p_{l,\mathbf{v}_{p,i}}(\mathbf{v}_u)$ is the prediction weight $\mathbf{v}_u$ contributed in estimating its $\mathbf{v}_{p,i}$ odd neighbor. By using this design, there is no guarantee the signal average will be preserved, unless applied to unweighted graphs, hence when using the umbrella operator Laplacian.

Abdul-Rahman et al. [1] proposed a similar approach where the update filter construction aims to directly preserve the average value of the one-ring neighborhood of an odd vertex before and after its removal. Using the same update vector for all even neighbors, the following equation ensues:

$$\frac{1}{N+1} \left(\mathbf{w}_{l,\mathbf{v}_n} + \sum_{\mathbf{v}_m \in \mathcal{N}_v^1(\mathbf{v}_n)} (p_{l,\mathbf{v}_n}(\mathbf{v}_m) + 1) \mathbf{s}_{l+1,\mathbf{v}_m} \right) = \tag{44}$$

$$\frac{1}{N} \sum_{\mathbf{v}_m \in \mathcal{N}_v^1(\mathbf{v}_n)} (\mathbf{s}_{l+1,\mathbf{v}_m} + \mathbf{u}_{l,\mathbf{v}_m}), \tag{45}$$

with $N = |\mathcal{N}_v^1(\mathbf{v}_n)|$, and the update vectors $\mathbf{u}_{l,\mathbf{v}_i} = \mathbf{u}_{l,\mathbf{v}_j} = \mathbf{u}_{\mathcal{N}_v^1(\mathbf{v}_n)}$ for any $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{N}_v^1(\mathbf{v}_n)$. Thus, the uniform update vector is determined as

$$\mathbf{u}_{\mathcal{N}_v^1(\mathbf{v}_n)} = \frac{1}{N+1} \mathbf{w}_{l,\mathbf{v}_n} + \sum_{\mathbf{v}_m \in \mathcal{N}_v^1(\mathbf{v}_n)} \frac{N p_{l,\mathbf{v}_n}(\mathbf{v}_m) - 1}{N(N+1)} \mathbf{s}_{l+1,\mathbf{v}_m}. \tag{46}$$

In case the prediction filter weights correspond to the umbrella operator type of Laplacian, the second term in Eq. (46) vanishes. Generally, this design requires storing update weights for both difference vectors and even nodes from the previous level, thus becoming a more memory consuming approach. By aiming to directly preserve the one-ring average of the scaling coefficients, it becomes more difficult to find update weights that depend only on the difference vectors. This is due to the fact that determining such weights implies solving a sparse system involving all scaling coefficients. In the multivariate scenario, this system becomes overdetermined and an exact solution may not exist. In this situation, an approximate solution must be searched for. Overall, this process is more complex than the entire lifting pipeline.

In [9] we also considered the solution proposed by Jansen et al. [22]. In principle, it also exploits the average preserving requirement. This prerequisite provides a mathematical constraint that can be expressed in terms of the integrals of the scaling functions, as described for the one-dimensional context discussion. Let $\varsigma_{l,\mathbf{v}_u}$ be the

integral of the scaling function $\phi_{l,\mathbf{v}_u}$ (i.e. an element of $\Phi_{l,V_{e_{l+1}}}$ corresponding to the vertex $\mathbf{v}_u \in E_l$). Rewriting Eq. (23), we obtain

$$\varsigma_{l,\mathbf{v}_u} = \varsigma_{l+1,\mathbf{v}_u} + \sum_{\mathbf{v}_k \in \mathcal{N}_v^1(\mathbf{v}_u) \cap V_{o_{l+1}}} p_{l,\mathbf{v}_k}(\mathbf{v}_u)\varsigma_{l+1,\mathbf{v}_k}. \tag{47}$$

The wavelet biorthogonality condition requires for each wavelet function to have zero integral. In consequence, when integrating equation (29) the left hand side term vanishes, i.e.,

$$0 = \varsigma_{l+1,V_{o_{l+1}}} - \mathbf{U}_l \varsigma_{l,V_{e_l}}. \tag{48}$$

Further rewriting Eq. (25) for each predicted vertex, $\mathbf{v}_p \in V_{o_{l+1}}$, leads to

$$\varsigma_{l+1,\mathbf{v}_p} = \mathbf{u}_{l,\mathbf{v}_p}^{\mathsf{T}} \varsigma_{l,\mathcal{N}_v^1(\mathbf{v}_p)}, \tag{49}$$

where $\mathbf{u}_{l,\mathbf{v}_p}$ is the vector containing the update weights this vertex contributes with for each of its one-ring, even vertices. Finding $\mathbf{u}_{l,\mathbf{v}_p}$ requires solving an overdetermined equation of the form $\alpha = \mathbf{u}^{\mathsf{T}}\mathbf{v}$, where $\mathbf{u}$ is the unknown vector. From all possible solutions, both Jansen et al. [22] and Wagner et al. [37] choose the minimum update norm solution due to its stabilizing effect. This translates into setting $\mathbf{u} = \frac{\alpha \mathbf{v}}{\|\mathbf{v}\|^2}$. Finally, Eq. (26), which gives the expression of the update vector of coefficients for $\mathbf{v}_p$, becomes

$$\mathbf{u}_{l,\mathbf{v}_p} = \frac{\varsigma_{l+1,\mathbf{v}_p}}{\sum_{\mathbf{v}_k \in \mathcal{N}_v^1(\mathbf{v}_p)} \varsigma_{l,\mathbf{v}_k}^2} \varsigma_{l,\mathcal{N}_v^1(\mathbf{v}_p)}. \tag{50}$$

With these results, the update Eq. (31) of an even scaling coefficient for a vertex $\mathbf{v}_u$ is written as

$$\mathbf{s}_{l,\mathbf{v}_u} = \mathbf{s}_{l+1,\mathbf{v}_u} + \sum_{\mathbf{v}_n \in \mathcal{N}_v^1(\mathbf{v}_u) \cap V_{o_{l+1}}} u_{l,\mathbf{v}_u}(\mathbf{v}_n)\mathbf{w}_{l,\mathbf{v}_n}. \tag{51}$$

The choice of the scaling functions does not affect the so far described transform. Since the initial integrals $\varsigma_{l+1,E_{l+1}}$ need to be computed, one option is to assume a choice of the scaling functions such that these integrals are all equal to 1.

Given the fact that a terrain mesh has sparse connectivity, the filter matrices are also sparse. Each odd node is surrounded by even nodes, ensuring a 25% average reduction factor for each decimation stage. Asymptotically, almost 80% of the initial edge density will still be required to store the filter matrices. Nevertheless, the information is still very sparse and lends itself for specific algebraic manipulations after the cascaded analysis stages.

Algorithm 3 Downsampling algorithm

INPUT: a d-dimensional point cloud consisting of vertex-attribute pairs.
OUTPUT: the lowest resolution vertex set, $\mathbf{s}_{0,V_0}$, the chain of prediction and update matrices, $(\mathbf{P}_l)_l$ and $(\mathbf{U}_l)_l$, and the string of difference vectors, $(\mathbf{w}_{l,V_{o_{l+1}}})_l$, where l ranges from 0 to the total number of refinement levels, L.
TRIANGULATE: the input set, V_L.
for l := L - 1 **downto** 0 **do**
  COMPUTE: the removal cost of each vertex using algorithm 1
  PARTITION: the vertex set $V_{l+1} = V_{e_{l+1}} \cup V_{o_{l+1}}$ using Algorithm 2.
  COMPUTE: $\mathbf{w}_{l,V_{o_{l+1}}}$ and $\mathbf{s}_{l,V_{e_l}}$ from $\mathbf{s}_{l+1,V_{e_{l+1}}}$ using Eqs. (30) and (31)
  STORE: $\mathbf{P}_l$, $\mathbf{U}_l$ and $\mathbf{w}_{l,V_{o_{l+1}}}$
end for
STORE: $\mathbf{s}_{0,V_0}$

2.3 Algorithm Overview

The lifting scheme flow depicted in Fig. 1 is at the core of the iterative downsampling process, illustrated in Fig. 4 and summarized in Algorithm 3. In overview, we can identify three main stages: the cost computation, the labeling or *lazy wavelet* decomposition and the analysis itself. Together, these stages resemble the structure of a classical filter bank.

This algorithm has $O(N \log N)$ complexity (where N is the total number of vertices). This higher complexity is due to the nodes being sorted according to their removal cost. While an $O(N)$ complexity is achievable, the salient features will not

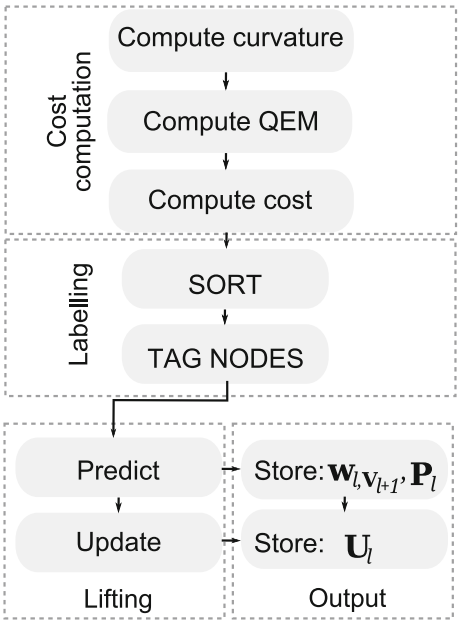


Fig. 4 An overview of our hybrid algorithm for downsampling an input set

be preserved as faithfully (see Fig. 8). The initial triangulation can be performed only once, during the pre-processing stage. During the downsampling steps, the one-rings of the odd vertices can be re-triangulated on the fly, keeping a consistent manifold connectivity.

The spatial complexity of this method is linear with respect to the size of the input data set. More precisely, the lowest resolution data and the difference vectors require the same amount of storage as the initial point cloud. Empirically, the triangulated meshes will not be far from semi-regular, thus the prediction weights will require $6N$ scalars in total. For the update weights, sparse structures corresponding to even vertices being surrounded by 3 odd vertices on average have to be stored. Assuming a decimation rate of 25%, the spatial requirements for storing the update weights asymptotically approach $15N$.

Quadric Error Matrix Update Procedure

The quadric error metric matrix fusion of collapsed vertex pairs is a property that we adapt locally, in the odd vertex neighborhood. We achieve this by distributing the matrix corresponding to an odd sample, $\mathbf{Q}(\mathbf{v}_{o_{j+1}})$, among its even neighbors. Thus, each even matrix is updated by

$$\mathbf{Q}(\mathbf{v}_{e_j}) = \mathbf{Q}(\mathbf{v}_{e_{j+1}}) + \sum_{\mathbf{v}_{o_{j+1}} \in \mathcal{N}_v^1(\mathbf{v}_{e_{j+1}}) \cap V_{o_{j+1}}} \rho_{j,\mathbf{v}_{e_{j+1}}}(\mathbf{v}_{o_{j+1}}) \mathbf{Q}(\mathbf{v}_{o_{j+1}}), \tag{52}$$

where $\rho_{j,\mathbf{v}_{e_{j+1}}}(\mathbf{v}_{o_{j+1}})$ is the redistribution weight describing the influence of $\mathbf{v}_{o_{j+1}}$ on its even neighbor, $\mathbf{v}_{e_{j+1}}$. We propose using the same weights that the prediction filter relies on estimating the lost information, i.e.,

$$\rho_{j,\mathbf{v}_{e_{j+1}}}(\mathbf{v}_{o_{j+1}}) = \frac{p_{j,\mathbf{v}_{o_{j+1}}}(\mathbf{v}_{e_{j+1}})}{\sum_{\mathbf{v}_{u_{j+1}} \in \mathcal{N}_v^1(\mathbf{v}_{o_{j+1}})} p_{j,\mathbf{v}_{o_{j+1}}}(\mathbf{v}_{u_{j+1}})}. \tag{53}$$

This choice is natural since the even vertex that contributes more to the prediction of an odd neighbor receives a larger fraction of its quadric error matrix. We justify this to be a more natural choice given that most prediction weights automatically encode a node similarity magnitude. Other quadric error matrix redistribution weights choices and strategies are possible and were analyzed in [7, 8]. Experimentally, the choice of weights presented through Eq. (53) was determined to achieve, on average, slightly more accurate approximations than the alternatives presented in [8].

3 Results and Discussion

We assessed the validity and efficiency of our method through a series of experiments involving three different LiDAR sets. The first model is a scan fragment of the Great Smoky Mountains (available through the www.opentopography.org/ portal) with a

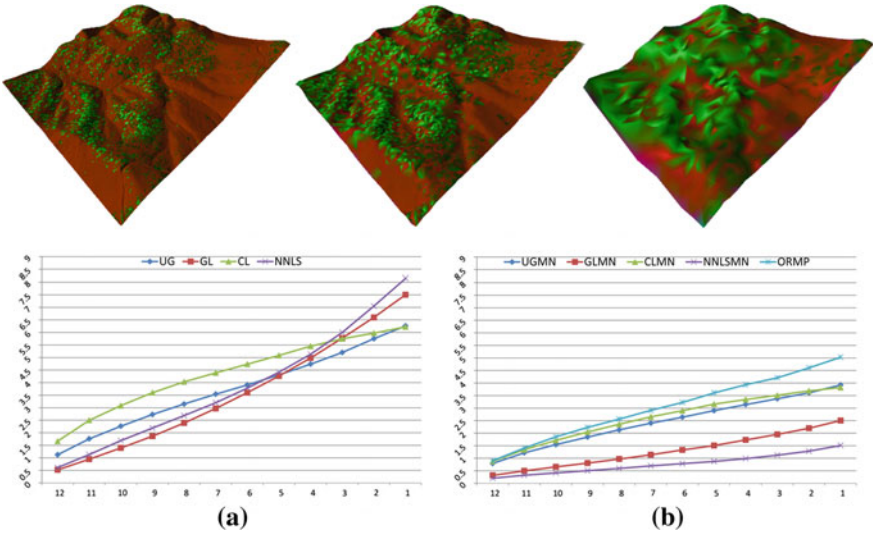


Fig. 5 Smoky Mountains fragment. Top row, left to right: shape and attribute evolution at levels 0, 12 and 24 using the UG filters. Bottom row: comparative charts for the RMSE evolution across 12 analysis sequences

density of approximately 2.23 points per square meter and a size of 280,000 points (Fig. 5). The second set is larger and much denser at approximately 20 points per square meter and a count of 9 million (Fig. 6). The third set is larger but less denser, consisting of 11.5 million points, at a density of 5 points per square meter (Fig. 7). These larger and denser LiDAR samples were acquired through custom aerial scans conducted over two regions of the Romanian Carpathians. All terrain samples contain both geometry (point coordinates) and attribute information (i.e., vegetation type, as a scalar between 0 and 20, and height above ground, as a scalar between 0 and 30). All coordinates are translated and scaled to fit within a zero-centered unit bounding box dividing them by their range width. This way we significantly alleviate the effect of the diverse scales on the final outcome. However, the attributes could be scaled differently if one desires to diminish or enhance their contribution.

3.1 Root Mean Squared Error Measurements

As a measure of quality and accuracy, we have chosen the root mean square error (RMSE). Although other error measuring mechanisms exist, the RMSE or L_2 norm is one of the simplest and most efficient indicators of shape and attribute quality (see [30] for a more in-depth discussion of the properties and applications of this error). In our case, the RMSE computation concerns both geometry and attribute vertex coordinates, except for the artificially added bending energy. Thus, if $\mathbf{s}_{l,\mathbf{v}_i}$ is

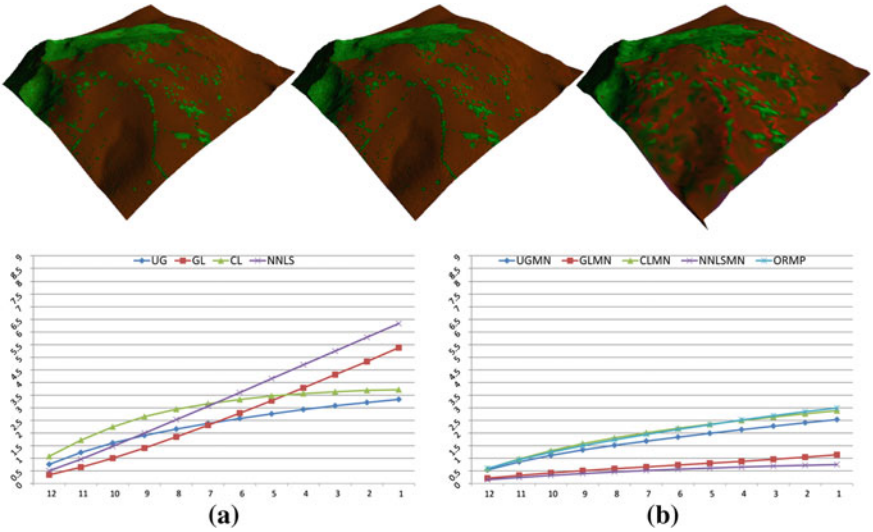


Fig. 6 High-density Carpathian Mountains fragment. Top row, left to right: shape and attribute evolution at levels 0, 12 and 24 the UG filters. Bottom row: comparative charts for the RMSE evolution across 12 analysis sequences

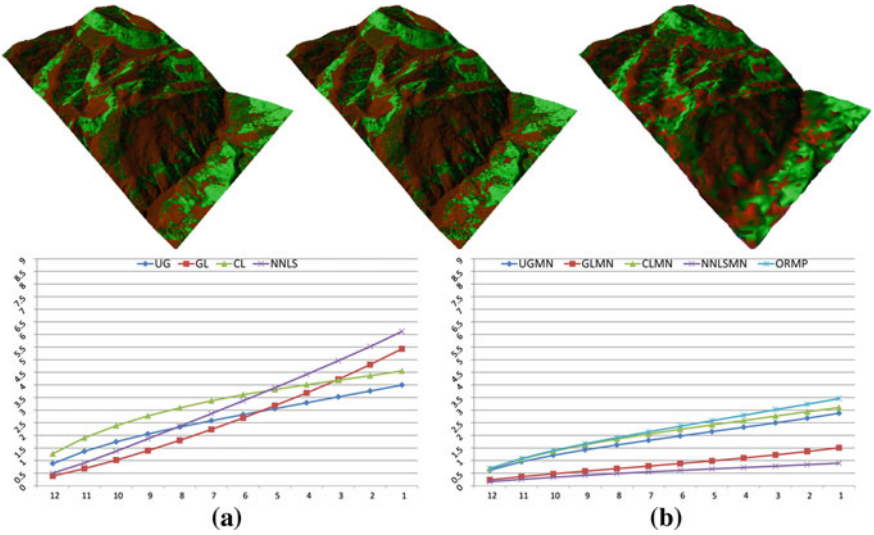


Fig. 7 Low-density Carpathian Mountains fragment. Top row, left to right: shape and attribute evolution at levels 0, 12 and 24 the UG filters. Bottom row: comparative charts for the RMSE evolution across 12 analysis sequences

the scaling coefficient at level l of a node $\mathbf{v}_i$, and $\mathbf{s}_{L,\mathbf{v}_i}$ is the corresponding coefficient in the initial input set, the RMSE is evaluated as

$$\mathtt{RMSE}_l = \sqrt{\sum_{\mathbf{v}_i \in E_l} \|\mathbf{s}_{L,\mathbf{v}_i} - \mathbf{s}_{l,\mathbf{v}_i}\|^2}. \tag{54}$$

For the multiresolution representation experiments, we have also considered several prediction and update filter designs. In this respect, we labeled the result sets according to the type of prediction and update filters as follows:

- *prediction filters*: uniform weights obtained by using the unweighted graph Laplacian (UG), present in the design of Martinez and Ortega [27], the cotangent Laplacian weights (CL), chosen by Abdul-Rahman et al. [1], the Fujiwara or geometric Laplacian (GL), which is our proposal for a Laplacian design for multivariate data and a constrained non-negative least squares weights design (abbreviated as NNLS), which we propose in order to minimize the detail vector norms.
- *update filters*: the filter design proposed by Martinez and Ortega [27] compensates for the detail loss incurred by the removal of the odd vertices, but it does not preserve the graph signal average for weighted graphs, unless all weights are uniform. On the other hand, the design used in [22, 37], achieves this goal while minimizing the norm of the update coefficient vectors. To distinguish between the first option and the minimum norm update weights, we have added the MN suffix to the prediction filter abbreviation. The third and final design option, proposed in [1], preserves the local, one-ring mean of the data samples and is abbreviated as ORMP.

3.2 Feature Selectivity and Robustness to Noise

For a proof of concept, we subjected the Smoky Mountains terrain fragment to a sequence of 8 analysis steps. The same set was again analyzed using the lazy-wavelet partitioning strategy employed in [27, 37]. Without our proposed feature preservation heuristic, the lowest resolution representation will lose a bigger fraction of the sharp features, as depicted in Fig. 8.

Computing the discrete curvatures using the method of [28] is recommended for meshes with low to no noise. In [9, 10], we have experimentally shown that the proposed heuristic partitioning of the data points into odd and even categories is robust to noise.

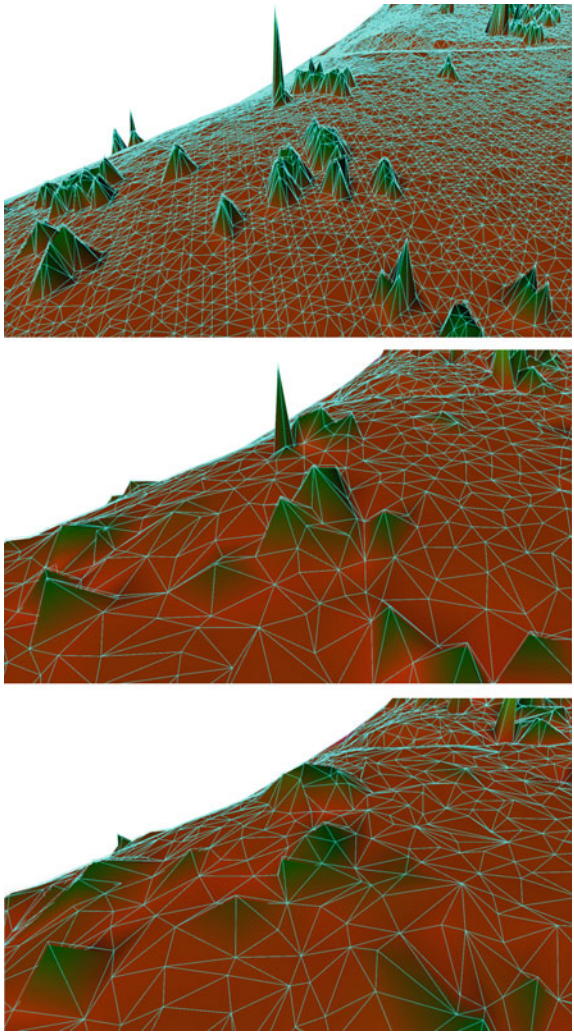


Fig. 8 Preservation of salient features after 8 analysis steps. Top image: high-density, high-detail input (1.6 M faces), middle image (160 K triangles): the combined QEM and thin plate energy cost, bottom image (158 K triangles): the graph coloring strategy used in [27, 37]

3.3 Vertex Frequency Analysis Interpretation

Vertex frequency analysis is a more recently developed subfield of Graph Signal Processing. This subfield discusses solutions for a graph equivalent of time and frequency domain localized transforms such as the windowed fast Fourier transform, the short-time Fourier transform or the windowed fast wavelet transform. Shuman et al. [33] described a solution based on a new definition for graph-based translation operators. Jestrović et al. [23] improved this solution by designing an $O(N^3)$ implementation, as opposed to the $O(N^4)$ complexity of the original. While these developments are sound mathematical generalizations (although some of the

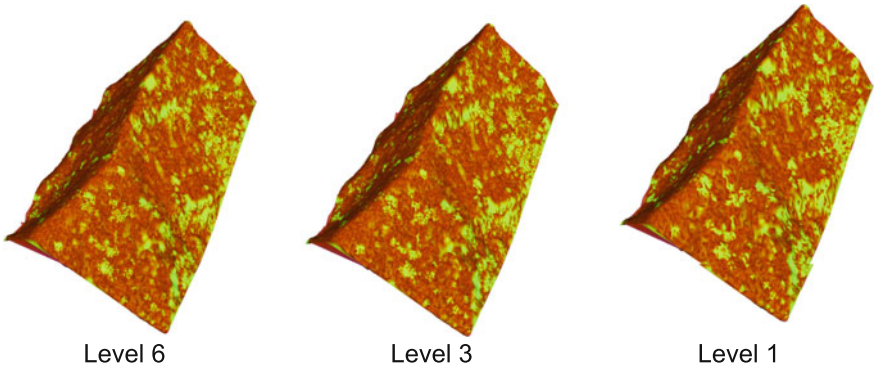


Fig. 9 Kauai Mountain fragment consisting of 79,000 point samples. Represented are level 6, 3 and 1 corresponding to the initial set, the middle resolution and the lowest resolution in a cascade of 5 analysis passes

translation operator properties on graphs are no longer preserved with respect to the one-dimensional case), their computational complexity may not lend them useful for processing very large sets that are common in most applied problems.

Using the wavelet coefficients resulted from a cascade of analysis steps, we can offer an intuitive interpretation that can partially cover the goal of vertex frequency analysis. Given the nature of lifting scheme wavelet constructions, it is not directly possible to discuss spectral features of the signal. However, the detail vectors associated to the odd samples at each scale offer an intuition in this direction. Typical graph vertex frequency analysis algorithms employ an adapted form of the windowed fast Fourier transform or simply restrict the Laplacian eigendecomposition to a neighborhood of a certain size around each vertex. Detail vectors, on the other hand, are directly associated with a single vertex and a single scale. Thus, the information encoded in these entities is, by definition, localized in space and frequency domain. As opposed to classical vertex frequency interpretations, it is not directly possible to examine more than one spectral component for a specific odd vertex. Nevertheless, the magnitude of the wavelet coefficients is a direct indicator of the strength of the local graph signal with respect to the characteristic frequency band of a level of resolution.

We propose an experiment where we subject a terrain fragment of the Kauai Mountain in Hawaii (extracted from the www.opentopography.org portal), to a sequence of five cascading analysis steps (see Fig. 9). The resulting difference vector amplitudes are then plotted in Fig. 10.

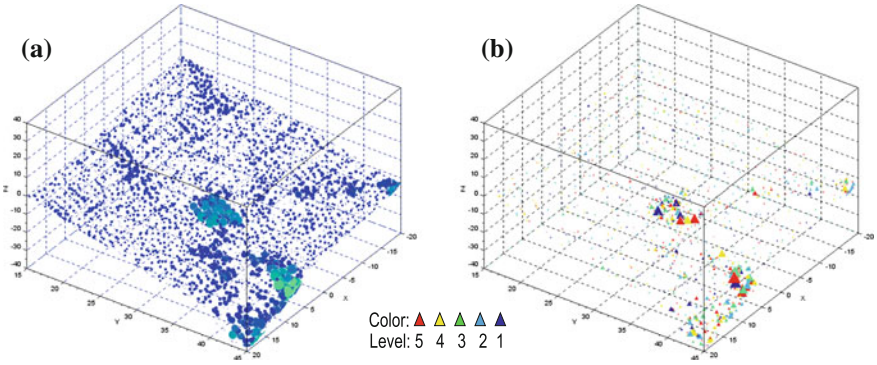


Fig. 10 Difference vector magnitude across 5 consecutive analysis passes of the Kauai fragment Fig. 9. The **a** subplot reveals the height above ground of the initial terrain points, while the **b** subplot contains markers indicating the strength (size) and frequency band of the resulting difference vectors at each of the odd vertices present in the **a** subplot. The regions in **a** with abrupt variations register a high frequency footprint, while similar points clustered together register lower frequency signatures as well

4 Conclusions

Several conclusions can be drawn by analyzing the results obtained using the different lifting designs. First, by using the update designs that are not guaranteed to preserve the signal average for general graphs (e.g. [27]), we observe the RMSE levels have the highest values, regardless of the prediction filter design. As such, the charts Figs. 5a, 6a and 7a reveal that over a progression of 12 analysis steps, the uniform weighing prediction design (UG) is, on average, the better choice. While the geometric Laplacian (GL) and the non-negative least squares weights (NNLS) yield better quality results, after 6 or 7 decomposition steps they become unstable. The use of the cotangent Laplacian weights (CL) is not justified either, since this design takes into account only the geometric structure of the data, regardless of the attribute variability. Overall, both uniform and cotangent Laplacian weights produce more stable results, but the uniform design is to be preferred due to its consistently lower RMSE levels. Next, directly preserving the average of the scale coefficients in a one-ring neighborhood (ORMP) contributes to both stability and error-minimizing properties of the analysis sequence for all sets (charts Figs. 5b, 6b and 7b). The best results are, however, achieved through the use of the minimum norm, mean preserving update weights (proposed by Jansen et al. [22]). This design ensures the highest stability while sensibly decreasing the mean square error. More specifically, for all terrain sets the ORMP design is surpassed by the combined use of Laplacian prediction weights and the minimum norm update vector coefficients. Both of our proposals, the geometric Laplacian (GLMN) and the non-negative least squares (NNLSMN) attain an almost twofold accuracy over the cotangent (CLMN) and uniform weights Laplacian (UGMN). While the RMSE could be reduced even further by lifting the

constraints on the least squares prediction filter, as proposed in [27, 37], the scheme becomes numerically unstable due to overfitting of the prediction weights values and the prediction of the boundary vertices. While we can fix the boundary vertices and alleviate the stability issue, our design does not require such vertex selection constraints to be applied.

The multiresolution experiments conducted with the three samples also confirm the 25% reduction ratio of the number of vertices after each downsampling step. More specifically, the average reduction rate attained for the Smoky set (Fig. 5) is 27%, for the high-density Carpathian set (Fig. 6) the average ratio is 28%, and for the low-density Carpathian set (Fig. 7) this average ratio is 27%.

As immediate applications for this graph-wavelet multiresolution framework, we suggest filtering [11] and, as experimentally examined in Fig. 10, we also suggest considering the potential of using the detail vector information to offer an intuitive classification similar to that of vertex frequency analysis.

References

1. H.S. Abdul-Rahman, X. Jane Jiang, P.J. Scott, Freeform surface filtering using the lifting wavelet transform. Precis. Eng. **37**(1):187–202 (2013)
2. C. Alberto Silva, C. Klauberg, A.M. Klein Hentz, A.P. Dalla Corte, U. Ribeiro, V. Liesenberg, Comparing the performance of ground filtering algorithms for terrain modeling in a forest environment using airborne LiDAR data. Floresta e Ambiente **25** 00 (2018)
3. M. Bertram, Wavelet analysis for progressive meshes, in *Proceedings of the 23rd Spring Conference on Computer Graphics, SCCG '07* (ACM, New York, 2007), pp. 161–167
4. G. Beyer, Terrain inclination and curvature from wavelet coefficients. J. Geodesy **76**(9), 557–568 (2003)
5. G. Chen, M. Maggioni, Multiscale geometric wavelets for the analysis of point clouds, in , *2010 44th Annual Conference on Information Sciences and Systems (CISS)* (2010), pp. 1–6
6. H. Choi, R. Baraniuk, Multiscale manifold representation and modeling, in *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05)*, vol. 4 (2005), pp. iv/569–iv/572
7. T. Cioaca, B. Dumitrescu, M.-S. Stupariu et al., Heuristic-driven graph wavelet modeling of complex terrain, in *Sixth International Conference on Graphic and Image Processing (ICGIP 2014)*, ed. by Y. Wang, X. Jiang, D. Zhang, vol. 9443 (SPIE Proceedings, 2015)
8. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, Combined quadric error metric and lifting scheme multivariate model simplification. U.P.B. Sci. Bull. Ser. C (2016)
9. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, Graph-based wavelet representation of multi-variate terrain data. Comput. Graph. Forum **35**(1), 44–58 (2016)
10. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, *Lazy wavelet simplification using scale-dependent dense geometric variability descriptors* (J. Control Eng. Appl, Inf, 2016)
11. T. Cioaca, B. Dumitrescu, M.-S. Stupariu, Riemannian filters for multi-variate mesh signals, in *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2017) - Volume 1: GRAPP, Porto, Portugal, February 27–March 1, 2017* (2017), pp. 228–235
12. R.R. Coifman, M. Maggioni, Diffusion wavelets. Appl. Comput. Harmon. Anal. 21(1):53 – 94, 2006. Special Issue: Diffusion Maps and Wavelets
13. L. De Floriani, P. Magillo, *Triangulated Irregular Network* (Springer, New York, 2016), pp. 1–2

14. L. Demaret, N. Dyn, M.S. Floater, A. Iske, Adaptive thinning for terrain modelling and image compression, in *Advances in Multiresolution for Geometric Modelling*, ed. by N.A. Dodgson, M.S. Floater, M.A. Sabin (Springer, Berlin, 2005), pp. 319–338
15. J.S. Evans, A.T. Hudak, A multiscale curvature algorithm for classifying discrete return lidar in forested environments. IEEE Trans. Geosci. Remote Sens. **45**(4), 1029–1038 (2007)
16. K Fujiwara, Eigenvalues of laplacians on a closed riemannian manifold and its nets, in *Proceedings of AMS*, vol. 123 (1995)
17. M. Garland, P.S. Heckbert, Simplifying surfaces with color and texture using quadric error metrics, in *Proceedings of the Conference on Visualization '98, VIS '98* (IEEE Computer Society Press, Los Alamitos, 1998), pp. 263–269
18. I. Guskov, W. Sweldens, P. Schröder, Multiresolution signal processing for meshes, in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99* (ACM Press/Addison-Wesley Publishing Co, New York, 1999), pp. 325–334
19. D.K. Hammond, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Analys. **30**(2), 129–150 (2011)
20. M. Isenburg, Lastools–efficient tools for lidar processing, version 150304 (2015)
21. M. Jansen, Multiscale local polynomial smoothing in a lifted pyramid for non-equispaced data. IEEE Trans. Signal Process. **61**(3), 545–555 (2013)
22. M.H. Jansen, G.P. Nason, B.W. Silverman, *Scattered Data Smoothing by Empirical Bayesian Shrinkage of Second Generation Wavelet Coefficients*, ed. by M. Unser, A. Aldroubi, vol. 4478 (2001)
23. I. Jestrović, J.L. Coyle, E. Sejdić, A fast algorithm for vertex-frequency representations of signals on graphs. Signal Process. **131**, 483–491 (2017)
24. W. Jingsong, K. Amaratunga, Wavelet triangulated irregular networks. Int. J. Geograph. Inf. Sci. **17**(3), 273–289 (2003)
25. M. Kalbermatten, D. Van De Ville, P. Turberg, D. Tuia, S. Joost, Multiscale analysis of geomorphological and geological features in high resolution digital elevation models using the wavelet transform. Geomorphology **138**(1), 352–363 (2012)
26. M. Lounsbery, Multiresolution analysis for surfaces of arbitrary topological type. Ph.D. thesis, Department of Computer Science and Engineering, U. of Washington (1994)
27. E. Martinez-Enriquez, A. Ortega, Lifting transforms on graphs for video coding. Data Compression Conference (DCC) **2011**, 73–82 (2011)
28. M. Meyer, M. Desbrun, P. Schrder, A.H. Barr, Discrete differential-geometry operators for triangulated 2-manifolds, in *Visualization and Mathematics III. Mathematics and Visualization*, ed. by H.-C. Hege, K. Polthier (Springer, Berlin Heidelberg, 2003), pp. 35–57
29. M. Pauly, M. Gross, L.P. Kobbelt, Efficient simplification of point-sampled surfaces, in *Proceedings of the Conference on Visualization '02, VIS '02* (IEEE Computer Society, Washington, DC, 2002), pp. 163–170
30. F. Payan, M. Antonini, Mean square error approximation for wavelet-based semiregular mesh compression. IEEE Trans. Visual. Comput. Graph. **12**(4), 649–657 (2006). July
31. R. Ronfard, J. Rossignac, Full-range approximation of triangulated polyhedra. Comput. Graph. Forum **15**(3), 67–76 (1996)
32. W.J. Schroeder, J.A. Zarge, W.E. Lorensen, Decimation of triangle meshes. SIGGRAPH Comput. Graph. **26**(2), 65–70 (1992)
33. D.I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**(2), 260–291 (2016)
34. J.P. Surez, A. Plaza, Four-triangles adaptive algorithms for rtin terrain meshes. Math. Comput. Model. **49**(5), 1012–1020 (2009)
35. W. Sweldens, The lifting scheme: a custom-design construction of biorthogonal wavelets. Appl. Comput. Harmon. Anal. **3**(2), 186–200 (1996)
36. M. van Kreveld, *Digital elevation models and tin algorithms, in Algorithmic Foundations of Geographic Information Systems* (Springer, Berlin, 1997), pp. 37–78

37. R. Wagner, H. Choi, R. Baraniuk, V. Delouille, Distributed wavelet transform for irregular sensor network grids, in *2005 IEEE/SP 13th Workshop on Statistical Signal Processing* (2005), pp. 1196–1201
38. M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, Discrete laplace operators: No free lunch, in *Proceedings of the Fifth Eurographics Symposium on Geometry Processing, SGP '07* (Eurographics Association, Aire-la-Ville, Switzerland, 2007), pp. 1196–1201