Data Science with Graphs:
A Signal Processing Perspective

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering.

Siheng Chen

Carnegie Mellon University
Pittsburgh, PA

Nov. 2016

Keywords: Graph signal processing, representations, sampling & recovery, detection & localization

"This duality can be pursued further and is related to a duality between past and future and the notions of control and knowledge. Thus we may have knowledge of the past but cannot control it; we may control the future but have no knowledge of it."

Claude Elwood Shannon

Abstract

A massive amount of data is being generated at an unprecedented level from a diversity of sources, including social media, internet services, biological studies, physical infrastructure monitoring and many others. The necessity of analyzing such complex data has led to the birth of an emerging framework, graph signal processing. This framework offers an unified and mathematically rigorous paradigm for the analysis of high-dimensional data with complex and irregular structure. It extends fundamental signal processing concepts such as signals, Fourier transform, frequency response and filtering, from signals residing on regular lattices, which have been studied by the classical signal processing theory, to data residing on general graphs, which are called graph signals.

In this thesis, we consider five fundamental tasks on graphs from the perspective of graph signal processing: representation, sampling, recovery, detection and localization. Representation, aiming to concisely model shapes of graph signals, is at the heart of the proposed techniques. Sampling followed by recovery, aiming to reconstruct an original graph signal from a few selected samples, is applicable in semi-supervised learning and user profiling in online social networks. Detection followed by localization, aiming to identify and localize targeted patterns in noisy graph signals, is related to many real-world applications, such as localizing virus attacks in cyber-physical systems, localizing stimuli in brain connectivity networks, and mining traffic events in city street networks, to name just a few. We illustrate the power of the proposed tools on two real-world problems: fast resampling of 3D point clouds and mining of urban traffic data.

Acknowledgments

It has been a long and hard fight. During the past five years, I was very lucky to have guidance, inspiration, help, support and love from so many people. This is a tribute for some of you.

I would like to express my deepest gratitude to my academic mother, Prof. Jelena Kovačević. Jelena, you are a perfect role model for me. If I could achieve something in my career, the glory and honor belongs to you. Thank you to the rest of my thesis committee, Prof. Aarti Singh, Prof. José Moura and Prof. Antonio Ortega. Aarti, you have taught me so much and contributed to many pieces of this work. I really appreciate your enthusiasm and meticulous. I am very proud that I have co-authored seven papers with you; José, your vision, hard-working and humor always inspire me. I am very proud that I have co-authored five papers with you; Antonio, your papers educated me a lot when I started to study graph signal processing. The discussions with you sharpened my work and encouraged me extending my thinking directions.

I would like to thank Dr. Aliaksei Sandryhaila. Aliaksei, thank you for introducing graph signal processing to me. It was a great experience to work with you. Thanks to all those brilliant people who contribute to graph signal processing. When I started my Ph.D, graph signal processing is a very small field, it is amazing to witness the growth of this field. Especially to Prof. Yue Lu, Prof. David Shuman and Prof. Pierre Vandergheynst, your work inspired me.

I would like to thank my data analysis project adviser, Prof. Christos Faloutsos. Christos, you are probably the nicest person I have ever seen. Your passion, hard-working and friendliness is absolutely amazing. I would like to thank my collaborators at the department of civil and environmental engineering, Dr. George Lederman, Prof. Fernando Cerda, Prof. Jacobo Bielak, Prof. James H. Garrett and Prof. Hae Young Noh. My research started from a project of bridge structural health monitoring. Thanks so much for all your support and help. I would like to thank my host and collaborators at Mitsubishi Electric Research Laboratories during my internship, Dr. Dong Tian, Dr. Chen Feng and Dr. Anthony Vetro. Especially to Dr. Dong Tian, I deeply appreciate your generous help.

Thanks to my labmates, Dr. Anupama Kuruvilla, Dr. Michael McCann, Dr. Jackie Chen, Dr. Filipe Condessa, Rohan Varma, Anuva Kulkarni, Chaojing Duan and the rest of CBI. Especially to Mike, thanks for your supervision in my first year. Thanks to those students that I advised: Yu Zhou, Sean Bittner, Tianxi Ji, Shi Zong, Chao Li, Chen Liang, Akshay Varun, Xiao Ma, Yuru Wu and Chao Pan; this is a much richer thesis because of your help.

Many others contributed to this thesis indirectly, in person, over phone, emails across time zones. Many thanks to my parents for your endless love! Many thanks to my friends, especially Dr. Pin-Yu Chen, Luyang Chen, Pinzhi Chen, Yilun Chen, Dr. Qin Gao, Jialiang Gu, Jia Guo, Zichun Guo, Dr. Jianjun He, Jian Li, Jiadong Ji, Yifan Jiang, Dr. Baoan Liu, Chi Liu, Dr. Tong Lv, Dan Lv, Sufeng Niu, Dr. Yajie Miao, Lingbing Peng, Jing Qu, Jinwei Shen, Qing Shi, Guowei Sun, Yifeng Tao,

Dr. Sui Tang, Tianhao Tang, Dr. Xiaohan Wang, Jian Wang, Xin Wang, Prof. Yang Weng, Dr. Weitao Wu, Jingling Xu, Dr. Feng Yang, Yaoqing Yang, Zhiding Yu, Shanghang Zhang, Xiaoyue Zhao.

Thanks to all the CMU staffs who have helped me. Christina Cowan, thank you for your cheer and I will always remember my promise to you. I also gratefully acknowledge support from the NSF through award 1421919, 1563918 and 1513936, and the CMU CIT Infrastructure Award.

Contents

List of Figures

List of Tables

Part I

Introduction and Background

Chapter 1

Introduction

1.1 Motivation

Data Science. Datasets collected from physical and engineering applications in fields like social activity, cell biology, economy and security are becoming larger and more complex. In many cases, the data is analyzed manually or by ad-hoc methods that extract only superficial information and can possibly lead to subjective or non-reproducible conclusions. There is thus an urgent need to develop methodologies that formalize complex data analysis.

Graph signal processing. The emerging framework of *graph signal processing* has offered a new paradigm for the analysis of high-dimensional data with complex and irregular structure [1,2]. The theory extends fundamental signal processing concepts including signals, Fourier transform, frequency response, low- and high-pass filtering, from signals residing on regular lattices, which have been studied by the classical signal processing theory, to data residing on general graphs. Furthermore, signal processing on graphs formulates and offers solutions to a number of tasks in data science, such as data compression, sampling, recovery, detection, localization, etc..

GOAL: Develop the fundamentals of data representation, sampling, recovery, detection and localization on graphs to solve graph structure related real-world problems.

In this thesis, we pursue our above-stated goal by answering, in a mathematically rigorous manner, the following fundamental questions:

Aim 1. How to properly represent graph signals?
We will extend relevant transforms and representation techniques for time signals and images such as Fourier and wavelet bases and dictionaries to those for general graph signals. The generalized representations will be consistent with the classical signal processing theory to the extent possible and will address additional challenges presented by graph structures and properties.

Aim 2. How to properly sample and recover graph signals?
We will extend relevant sampling and recovery strategies for time signals and images such as uniform, experimentally designed, and active (feedback-driven) sampling to those for general graph signals. The generalized sampling and recovery strategies will be consistent with the classical

signal processing theory and will address additional challenges presented by graph structures and properties.

Aim 3. How to properly detect and localize graph signals?
We will extend relevant detection and localization algorithms for time signals and images. The generalized detection and localization algorithms will be consistent with the classical signal processing theory and will address additional challenges presented by graph structures and properties.

Broader Impact. The proposed tools are based on a combination of signal processing, machine learning, data mining and network science. The goal is to provide solutions for graph structure related real-world applications. For example,

- designing optimized routes for autonomous vehicles in a traffic network
- labeling a huge number of unlabeled facial images with few queries
- identifying the most representative individuals in Facebook or Twitter
- distinguishing the key features in a LinkedIn user profiles that lead to a connection
- understanding the brain activity in the brain connectivity network

As researchers in numerous fields are collecting and studying unprecedented amounts of data, analysis and processing of that data is a task that requires enormous resources and may unveil only the superficial information. A rigorous mathematical framework is thus needed that will allow researchers to formulate data analysis problems in a principled way and provide a set of methodologies that are applicable to datasets of different origin, nature, and contents. Classical signal processing provides a unified way to process time signals and images. Similarly, graph signal processing aims to provide a unified way to process data with complex structure and to offer a set of standard signal processing tools—among others filtering, frequency analysis and sampling—that are instantiated and applied to data from different domains.

The Need for Signal Representations on Graphs. Signal representations provide a succinct way to capture valuable hidden information in the data and lay a foundation for solving numerous problems in data analysis and processing. Compared to time signals and images, signals supported on complex, irregular graph structures have richer hidden information and require more advanced representation techniques. A good representation lays a foundation to uncover hidden patterns and structure within data and is beneficial for related tasks including approximation, compression, sampling, recovery, denoising, detection, localization, and many others.

The Need for Signal Sampling and Recovery on Graphs. As one of the most fundamental tasks, sampling followed by recovery considers selecting few representative data samples from a large dataset that capture most of the original information from those samples and reconstructing the original data distribution based on those samples. In contrast to the sampling of time signals and images, where Nyquist sampling is the norm, we aim to design more efficient sampling strategies than the existing ones for time signals and images by taking advantage of the underlying complex, irregular graph structures. These sampling strategies can be considered as extensions of the Nyquist sampling for regular domains. A good sampling strategy is beneficial for active learning and dimensionality reduction. For example, in semi-supervised/active learning, we want to label some data samples as the training data. A sampling strategy helps choose

the most representative training data and lead to better learning performance. A good recovery strategy is beneficial for prediction, completion and denoising. For example, we want to monitor the health status of bridges based on the noisy measurements from sensor networks. Similarly, in the Netflix problem, users typically rate only a few movies. To recommend movies based on a users preferences, we want to predict the users' preference for unrated items. This is equivalent to complete a matrix of graph signals from a few random, noisy samples.

The Need for Signal Detection and Localization on Graphs. Detection followed by localization has been intensely studied in classical signal/image processing from various aspects over the past few decades. For example, impulse detection localizes impulses in a noisy signal [3]; change-point detection identifies times when the probability distribution of a stochastic process or time series changes [4]; support recovery of sparse signals localizes sparse activations with a limited number of samples [5, 6]; foreground detection localizes foreground in a video sequence [7]; cell detection and segmentation localize cells in microscopy images [8] and matched filtering localizes radar signals in the presence of additive stochastic noise [9, 10]. In a big data era, a massive amount of data are generated, which makes pattern discovery more challenging and meaningful. For example, do users from Carnegie Mellon University form social communities on Facebook? do researchers from signal processing community tightly cooperate with each other? do Chinese restaurants in Manhattan cluster together? These seemingly different problems share common structure: some attribute may be localized on a large-scale graph; in other words, a graph signal may exhibit some structure-related property. Graph signal detection aims to distinguish which graph signal is structure-related and graph signal localization aims to find the precise supports of targeted patterns. A good detection and localization strategy is beneficial for many important applications, such as detecting virus attacks in cyber-physical systems, localizing stimulus in brain connectivity networks, and mining traffic events in city street networks.

1.2 Thesis Contribution

The core contribution of this thesis is to deal with five tasks on graphs, including representation, sampling, recovery, detection and localization, from a perspective of signal processing.

- representation: we consider three graph signal models: smooth, piecewise-constant and piecewise-smooth graph signals. For each graph signal model, we propose a corresponding graph dictionary to achieve efficient and provably effective approximation;
- sampling: we design provably effective sampling strategies to select the most informative samples from smooth graph signals, which also presents a comprehensive explanation for when and why sampling for semi-supervised learning with graphs works;
- recovery: we formulate a general optimization problem to recover one or multiple smooth graph signals from noisy, corrupted, or incomplete measurements. We further provide corresponding solutions and theoretical analysis;
- detection: we formulate a statistical hypothesis test to decide whether a given graph signal is localized or not. We propose two statistics, including graph wavelet statistic and graph scan statistic, which are provably effective to detect localized graph signals;

- localization: we formulate a general optimization problem to find the supports of localized graph signals. We further propose a computationally efficient solver, which is parameter-free, scalable and robust to noise.

1.3 Thesis Outline

This thesis is divided in six parts: introduction and required background on data science with graphs; representations of graph signals; sampling and recovery of graph signals; detection and localization of graph signals; two case studies with applications to resampling of 3D point clouds and mining of urbafilters n traffic data; and concluding remarks and further work.

In Part I, we introduce graph theory, graph mining, network science and graph signal processing. In Part II, we present our work on representations of smooth graph signals (Chapter 3), piecewise-constant graph signals (Chapter 4) and piecewise-smooth graph signals (Chapter 5). In Part III, we present our work on sampling of bandlimited graph signals (Chapter 6), sampling of approximated-bandlimited graph signals (Chapter 7), and recovery of smooth graph signals (Chapter 8). In Part IV, we present our work on detection of localized graph signals (Chapter 9), localization of localized graph signals (Chapter 10). In Part V, we present our work on fast resampling of 3D point clouds (Chapter 11), pattern mining in urban traffic data (Chapter 12). We conclude in Part IV.

Chapter 2

Background

Graphs are generic forms to represent the inter-dependencies by the introduction of edges between the related objects. It provide a powerful machinery for effectively capturing both short- and long-range correlations among objects. There is a long history about graphs. The earliest known work in this field is the famous seven bridges of Königsberg written by Leonhard Euler in 1736. Euler's mathematical description of vertices and edges was the foundation of graph theory. Nowadays, graphs are broadly interested in a wide range of disciplines, such as physics, biology, social sciences, information systems and many others.

In this chapter, we first briefly introduce graph-related topics and then introduce graph signal processing, which lays a foundation of this thesis.

2.1 Graph Theory

In mathematics, graph theory is the study of graphs, which are structures used to model pairwise relations between objects. A graph is made up of nodes, which are connected by edges. A graph may be either directed (each edge has a direction), or undirected (there is no distinction between the two nodes associated with each edge). A graph may be either weighted (edges have weights or values) or unweighted (edge weights are binary).

Graph theory introduces basic concepts in graphs, such as degree, path, walk, cycle, tree, geodesic distance, complement graph, clique, bipartite graph, subgraph and many others [11,12]. Graph theorists have been studying various types of graphs, such as planer graphs (each node has a coordinate in the plane and there is no cross edges in the plane), interval graphs (arising in scheduling), symmetric graphs (hypercubes and those from group theory), routing networks (from communications), random graphs, and computational graphs that are used in designing algorithms or simulations [13]. Some prevailing topics in graph theory includes subgraph matching problem [14], graph coloring [15], shortest/longest path algorithms [16], minimum graph cuts [17], maximum flow [18], graph drawing [19].

2.2 Spectral Graph Theory

Spectral graph theory studies the properties and structure of a graph from its graph spectrum; that is, it is about the eigenvalues and eigenvectors of matrices associated with graphs, and their applications [20]. Graph theory emphasizes the analysis in the graph vertex domain and spectral graph theory emphasizes the analysis in the graph spectral domain, which corresponds to the properties of eigenvalues and eigenvectors of the graph Laplacian matrix.

Spectral graph theory is related to the physics in resistor networks and spring networks [21]. A resistor network is a combination of several resistors that are configured into a pattern. Corresponding edge weights are determined by the resistance. The relation between voltage and current can be linked by a graph Laplacian matrix induced by effective resistance. A spring network is a combination of several springs. Each edge corresponds to a spring. When we nail down some nodes and minimize the total potential energy, the solution is also related to the graph Laplacian matrix. Spectral graph theory is also widely used in spectral clustering [22], spectral layout [23], graph sparsification [24] and Laplacian-related linear programming [25]. Some fundamental ideas in spectral graph theory includes spectral decomposition, Rayleigh quotient, Cheegers inequality, random walks and heat kernels.

2.3 Network Science

From a perspective of physics, network science studies complex networks as a new class of objects and phenomena in networks as a new class of universe behaves [26]. In terms of terminology, graphs are the mathematical abstraction of networks and networks are practical implementation of graphs.

Several types of networks are intensely studied recently, including the Internet, telephone network, power grids, transportation network, online social network, citation network, biological network and economic network. Empirical study of those networks summarizes their structure and physical behaviors, such as the six degrees of separation and the power-law phenomenon. Various network models are proposed to mathematically model the network structure, such as Erdős-Rényi random model, Watts-Strogatz small work model and preferential attachment model. To quantify the network property, network-related measures and metrics are also proposed, such as degree centrality, betweenness centrality and closeness centrality. Some prevailing topics include epidemics process on networks [27], community detection [28] and dynamics on networks [29].

2.4 Graph Mining

From a perspective of computer science, structure mining or structured data mining is the process of finding and extracting useful information from semi-structured data sets. Graph mining, sequential pattern mining and molecule mining are special cases of structured data mining. Graph mining investigates the modeling, managing, and mining of large-scale graphs and networks in bioinformatics, social networks, and computer systems [30]. Its applications range from commu-

nity detection in social networks, malicious program analysis in computer security, to searches for functional modules in biological pathways and structural analysis in chemical compounds. Some prevailing topics include anomaly detection [31], graph visualization, graph summarization [32], fraud user detection [33], ego-network analysis [34] and user behavior modeling [35].

2.5 Graph Signal Processing

Graph signal processing [1, 36] is both a generalization of and an axiomatic framework to classical discrete signal processing. Graph signal processing deals with signals with an underlying complex and irregular structure. The framework models that underlying structure by a graph and signals by graph signals, generalizing concepts and tools from classical discrete signal processing to graph signal processing.

Recent additions to the toolbox include graph-based transforms [37–40], representations for graph signals [41–43], uncertainty principles on graphs [44–46], graph filter bank design [47–51], graph clustering [52–54], stationary graph signal processing [55–57], graph signal denoising [47, 58, 59], sampling of graph signals [60–64], recovery of graph signals [65–69], graph topology learning [70–74], and many others.

Graphs

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathrm{W})$ be a directed, weighted graph, where $\mathcal{V} = \{v_i\}_{i=1}^N$ is the set of nodes, $\mathcal{E} = \{e_i\}_{i=1}^E$ is the set of weighted edges, and $\mathrm{W} \in \mathbb{R}^{N \times N}$ is the weighted adjacency matrix. When there is an edge $e = (i, j)$ connecting the ith node and the jth node, the entry $\mathrm{W}_{i,j}$ is the edge weight, measuring the inter-dependency between the ith and the jth nodes. The connectivities and edge weights are either dictated by the physics of the problem at hand or inferred from the data. In most tasks in graph signal processing, we consider a fixed and given graph.

Graph signals

A *graph signal* is a map from a set of nodes $\mathcal{V}$ into a set of real numbers $\mathbb{R}$. Given a fixed ordering of nodes, we assign a signal coefficient to each node, we write a graph signal as a vector,

$$\mathbf{x} = [x_1, x_2, \cdots, x_N]^T \in \mathbb{R}^N,$$

with x_i the signal coefficient corresponding to the node v_i.

The examples of graph signals exist in numerous engineering and science fields. Time series, such as acoustic data, vibration data and financial data, are indexed by time stamps and are signals supported on a directed cyclic graph. Each node corresponds to a time stamp, and each value is related to the value at the previous time stamp, reflecting the causality of a time series. This relation is asymmetric, hence all edges are directed and have the same weight. Images are indexed by pixels and are signals supported on a lattice graph. Each node corresponds to a pixel, and each pixel value (intensity) is related to the values of the four adjacent pixels. This relation is symmetric, hence all edges are undirected and have the same weight, with possible exceptions

of boundary nodes that may have directed edges and/or different edge weights, depending on boundary conditions. In online social networks and recommender systems, we may be interested in analyzing attributes describing the users. Those attributes are signals supported on relational graphs, where nodes correspond to users and edges correspond to the relationship between users. In city street networks, we may be interested in analyzing traffic data and census data describing human mobility patterns. Those data are signals supported on a city street graph, where nodes correspond to intersections and edges correspond to the road segments connecting intersections.

Graph elementary operators

Graph elementary operators are elementary tools to process graph signals with priors on graph structure. Graph elementary operators build basic interactions between graph signals and graphs.

Graph shift operator. In discrete signal processing, a time shift (delay) is a basic nontrivial operation performed on a signal. Let $\mathbf{x} \in \mathbb{R}^N$ be a time series. The shifted signal is

$$\widetilde{\mathbf{x}} \;=\; \mathrm{C}\,\mathbf{x}, \tag{2.1}$$

where time shift operator C is the $N \times N$ cyclic permutation matrix

$$\mathrm{C} \;=\; \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \tag{2.2}$$

The cyclic permutation matrix C is exactly the adjacency matrix of the directed cyclic graph.

Graph signal processing extends the concept of time shift to general graphs by defining *graph shift* [36]. A graph shift operator $\mathrm{A} \in \mathbb{R}^{N \times N}$ is a local operation that replaces a signal coefficient at a node with a weighted linear combination of coefficients at its neighboring nodes. Let $\mathbf{x} \in \mathbb{R}^N$ be a graph signal. The shifted graph signal is

$$\widetilde{\mathbf{x}} \;=\; \mathrm{A}\,\mathbf{x}, \tag{2.3}$$

where the ith element $\widetilde{x}_i = \sum_j \mathrm{A}_{i,j}\, x_j$ is a weighted linear combination of the signal coefficients at the neighbors of the ith node. The graph shift (2.3) is a natural generalization of the time shift (2.1). It can be interpreted as a first-order interpolation, weighted averaging, or regression on graphs, which is a widely used in graph regression, distributed consensus, telecommunications, Markov processes and other approaches.

A graph shift operator is chosen according to practical needs. Some common choices of a graph shift are weighted adjacency matrix W, normalized adjacency matrix $\mathrm{W}_{\mathrm{norm}} = \mathrm{D}^{-\frac{1}{2}}\,\mathrm{W}\,\mathrm{D}^{-\frac{1}{2}}$, transition matrix $\mathrm{P} = \mathrm{D}^{-1}\,\mathrm{W}$, where the degree matrix D is a diagonal matrix whose ith diagonal element d_i is equal to the sum of the weights of all the edges incident to the ith node.

Graph Laplacian operator. A graph Laplacian operator, defined as $\mathrm{L} = \mathrm{D} - \mathrm{W} \in \mathbb{R}^{N \times N}$, is a local difference operation that replaces a signal coefficient at a node with a weighted difference [37]. Let $\mathbf{x} \in \mathbb{R}^N$ be a graph signal. The graph signal operated by the graph Laplacian operator is

$$\widetilde{\mathbf{x}} \;=\; \mathrm{L}\,\mathbf{x}, \tag{2.4}$$

where the ith element $\widetilde{x}_i = \sum_j \mathrm{A}_{i,j}\,(x_i - x_j)$ is a weighted difference between the signal coefficient at the ith node and signal coefficients at the neighbors of the ith node. The graph Laplacian operator is valid only when all the edge weights are nonnegative.

Graph difference operator. A graph incidence matrix is a matrix that shows the relationship between nodes and edges. Let $\Delta \in \mathbb{R}^{E\times N}$ be a *graph incidence matrix*, or a *graph difference operator* [1], where E is the number of edges. Each row of Δ corresponds to a directed edge associated with two nodes. For example, if e_i is a directed edge that starts from the jth node to the kth node, the elements of the ith row of Δ are

$$\Delta_{i,\ell} = \begin{cases} \sqrt{|\,\mathrm{W}_{j,k}\,|}, & \ell = j; \\ -\sqrt{|\,\mathrm{W}_{j,k}\,|}, & \ell = k; \\ 0, & \text{otherwise}, \end{cases} \tag{2.5}$$

where W is the weighted adjacency matrix. For a unweighted graph, in the row of edge e, there is one 1 in the column corresponding to one node of e and one -1 in the column corresponding to the other node of e, and all other columns have 0. The graph incidence operator is unique up to negation of any of the rows, since negating the entries of a row corresponds to reversing the direction of an edge.

The graph difference operator compares the differences between the signal coefficients at all the adjacent nodes and the output $\Delta\mathbf{x}$ is an *edge signal* recording those differences. The ith element of $\Delta\mathbf{x}$,

$$(\Delta\mathbf{x})_i \;=\; \sqrt{|\,\mathrm{W}_{j,k}\,|}\,(x_j - x_k)\,,$$

is the difference between two adjacent signal coefficients associated with the ith edge, where the ith edge connects the jth node to the kth node.

Graph Fourier transform

Mathematically, a Fourier transform with respect to a set of operators is the expansion of a signal into a basis of the operators' eigenfunctions. Recall that the ith Fourier coefficient of a finite time series of length N is

$$\widehat{x}_k \;=\; \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} x_i e^{-j\frac{2\pi}{N}ik},$$

and the time signal's discrete Fourier transform is written in vector form as

$$\widehat{\mathbf{x}} \;=\; \mathbf{DFT}_N\,\mathbf{x},$$

where $\mathbf{DFT}_N$ is the $N\times N$ discrete Fourier transform matrix with the (i,k)th entry $1/N\exp(-j2\pi ik/N)$, which also corresponds to the eigendecomposition of the time shift operator (2.2)

$$\mathrm{C} \;=\; \mathbf{DFT}_N^{-1} \begin{bmatrix} e^{-j\frac{2\pi\cdot 0}{N}} & & \\ & \ddots & \\ & & e^{-j\frac{2\pi\cdot(N-1)}{N}} \end{bmatrix} \mathbf{DFT}_N\,.$$

The graph Fourier transform generalizes the discrete Fourier transform and is used to represent the graph signal in the graph spectral domain. Let a graph shift operator A be diagonalizable. We consider its eigendecomposition as

$$\mathrm{A} \;=\; \mathrm{V}\,\Lambda\,\mathrm{V}^{-1}, \tag{2.6}$$

where the columns $\mathbf{v}_i$ of $\mathrm{V} \in \mathbb{C}^{N\times N}$ are the eigenvectors of A, and $\Lambda \in \mathbb{C}^{N\times N}$ is the diagonal matrix of corresponding eigenvalues $\lambda_1, \cdots, \lambda_N$ of A. If A is not diagonalizable, the Jordan decomposition is used [75].

The *graph Fourier transform* of $\mathbf{x} \in \mathbb{R}^N$ is

$$\widehat{\mathbf{x}} \;=\; \begin{bmatrix}\widehat{x}_1 & \cdots & \widehat{x}_N\end{bmatrix} \;=\; \mathrm{V}^{-1}\,\mathbf{x} \;=\; \mathrm{F}\,\mathbf{x}, \tag{2.7}$$

where $\mathrm{F} = \mathrm{V}^{-1}$ is the graph Fourier transform matrix.

The values $\widehat{x}_i$ in (2.7) are the signal's expansion in the eigenvector basis and represent the graph frequency content of the signal $\mathbf{x}$. The eigenvalues λ_i of the graph shift operator A represent graph frequencies, and the eigenvectors $\mathbf{v}_i$ represent the corresponding graph frequency components.

The *inverse graph Fourier transform* reconstructs the graph signal from its frequency content by combining graph frequency components weighted by the coefficients of the signal's graph Fourier transform:

$$\mathbf{x} \;=\; \mathrm{F}^{-1}\,\widehat{\mathbf{x}} \;=\; \mathrm{V}\,\widehat{\mathbf{x}}. \tag{2.8}$$

Graph spectral analysis

In discrete signal processing, frequency contents of time series and digital images are described by complex or real sinusoids that oscillate at different rates. These rates provide an intuitive, physical interpretation of low and high frequencies: low-frequency components oscillate less and high-frequency ones oscillate more.

In analogy to discrete signal processing, frequency components on graphs can also be characterized as low and high frequencies. In particular, this is achieved by ordering the graph frequency components according to how much they change across the graph; that is, how much the signal coefficients of a frequency component differ at connected nodes. The amount of change is calculated using the graph total variation [76]. The ℓ_p-norm based graph total variation defined as,

$$\mathrm{S}_p(\mathbf{x}) = \left\| \mathbf{x} - \frac{1}{|\lambda_{\max}(\mathrm{A})|}\,\mathrm{A}\,\mathbf{x} \right\|_p^p, \tag{2.9}$$

where $p \geq 1$ and $\lambda_{\max}(\mathrm{A})$ is the eigenvalue of A with the largest magnitude. We can show that when the eigenvalues of the graph shift A are sorted in a nonincreasing order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$, the variations of the corresponding eigenvectors follow a nondecreasing order $\mathrm{S}_p(\mathbf{v}_1) \leq \mathrm{S}_p(\mathbf{v}_2) \leq \ldots \leq \mathrm{S}_p(\mathbf{v}_N)$. The variations of graph Fourier basis vectors thus allow us to provide the ordering: the Fourier basis vectors with small variations are considered as *low-frequency* components while the vectors with large variations are considered as *high-frequency* components [76].

Based on the above discussion, the eigenvectors associated with large eigenvalues of the graph shift represent low-frequency components and the eigenvectors associated with small eigenvalues of the graph shift represent high-frequency components. In the following discussion, we assume all graph Fourier bases are ordered from low frequencies to high frequencies.

Graph filters

In signal processing, a filter is a system $\mathrm{H}(\cdot)$ that takes a graph signal $\mathbf{x}$ as an input and outputs

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

Among the most widely used filters are linear shift-invariant ones. A filter is linear, if for a linear combination of inputs, it produces the same combination of outputs: $\mathrm{H}(\alpha\mathbf{x}_1 + \beta\mathbf{x}_2) = \alpha\,\mathrm{H}(\mathbf{x}_1) + \beta\,\mathrm{H}(\mathbf{x}_2)$. Filters $H_1(\cdot)$ and $H_2(\cdot)$ are commutative, or shift-invariant, if the order of their application to a signal does not change the output: $H_1(H_2(\cdot)) = H_2(H_1(\cdot))$. Every linear, shift-invariant filter is a polynomial in the time shift [2]; that is, the output signal is given by the product

$$\mathbf{y} \;=\; h(\mathrm{C})\mathbf{x} \;\in\; \mathbb{R}^N.$$

of the input signal $\mathbf{x}$ and the matrix

$$\begin{aligned} h(\mathrm{C}) \;&=\; \sum_{i=0}^{N-1} h_i\,\mathrm{C}^i \\ &=\; \begin{bmatrix} h_0 & h_{N-1} & \cdots & h_1 \\ h_0 & \ddots & \ddots & \vdots \\ h_0 & \ddots & \ddots & h_{N-1} \\ h_{N-1} & \cdots & h_1 & h_0 \end{bmatrix} \end{aligned}$$

Graph signal processing extends the concept of filters to general graphs by defining graph filters. A linear, shift-invariant graph filter has the form

$$h(\mathrm{A}) \;=\; \sum_{\ell=0}^{L-1} h_\ell\,\mathrm{A}^\ell = h_0\,\mathbf{I} + h_1\,\mathrm{A} + \ldots + h_{L-1}\,\mathrm{A}^{L-1}, \tag{2.10}$$

where h_i are filter coefficients and L is the length of this graph filter. Its output is given by the matrix-vector product

$$\mathbf{y} \;=\; h(\mathrm{A})\mathbf{x} \;\in\; \mathbb{R}^N. \tag{2.11}$$

Graph filters have a number of important properties. An inverse of a graph filter, if it exists, is also a graph filter that can be found by solving a system of at most N linear equations. Also, the number of taps in a graph filter is not larger than the degree of the minimal polynomial of A, which provides an upper bound on the complexity of their computation.

To express the frequency content of graph signals, the graph Fourier transform also characterizes the effect of filters on the frequency content of signals. The filtering operation (2.11) can be written as

$$\mathbf{y} \;=\; h(\mathrm{A})\mathbf{x} \;=\; h(\mathrm{F}^{-1}\,\Lambda\,\mathrm{F})\mathbf{x} \;=\; \mathrm{F}^{-1}\,h(\Lambda)\,\mathrm{F}\,\mathbf{x}, \tag{2.12}$$

where $h(\Lambda)$ is a diagonal matrix with values $h(\lambda_k) = \sum_{\ell=0}^{L-1} h_\ell \lambda_k^\ell$ on the diagonal. As follows from (2.12),

$$\mathbf{y} \;=\; h(\mathrm{A})\mathbf{x} \iff \widehat{\mathbf{y}} \;=\; h(\Lambda)\widehat{\mathbf{x}}, \tag{2.13}$$

where $\widehat{\mathbf{x}} = \mathrm{F}\,\mathbf{x}$ and $\widehat{\mathbf{y}} = \mathrm{F}\,\mathbf{y}$. That is, the frequency content of a filtered signal is modified by multiplying its frequency content elementwise by $h(\lambda_k)$. These values represent the graph frequency response of the graph filter (2.10). The relation (2.13) is a generalization of the classical convolution theorem to graphs: filtering a graph signal in the graph domain is equivalent in the frequency domain to multiplying the signal's spectrum by the frequency response of the graph filter.

Part II

Representations of Graph Signals

Overview of Representations

The task of representations has always been at the heart of most signal processing techniques. It considers describing similar signals by using a mathematical model. In classical signal processing, space-time-frequency representations lay the foundation for analyzing time-series signals and images. In this chapter, we discuss the representations of graph signals. This topic has been studied in the previous literature for a while [37, 42, 77–79]; however, most proposed representations do not target on any specific graph signal model. Here we rigorously define graph signal models and design the corresponding graph dictionaries.

There are mainly two approaches to design a representation for graph signals: one is based on the graph Fourier domain and the other one is based on the graph vertex domain. The representations based on the graph Fourier domain are based on the spectral properties of the graph. The most fundamental representation based on the graph Fourier domain is the graph Fourier basis, which is the eigenvectors of a matrix that represents a graph structure [1, 2]. Based on the graph Fourier basis, people propose various versions of multiresolution transforms on graphs, including diffusion wavelets [77], spectral graph wavelets [37], graph quadrature mirror filter banks [47], windowed graph Fourier transform [79], polynomial graph dictionary [42]. The representations based on the graph vertex domain are based on the connectivity properties of the graph. Some examples include multiscale wavelets on trees [80], graph wavelets for spatial analysis [81], spanning tree wavelet basis [78].

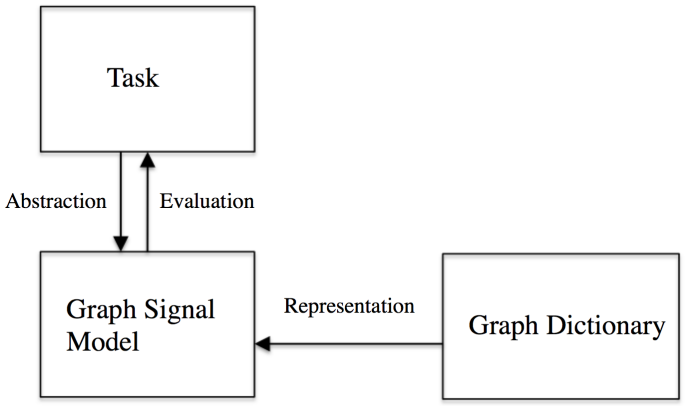


Figure 2.1: The central concept here is the graph signal model, which is abstracted from given data and is represented by some dictionary.

We consider a representation-based framework with three components, including graph signal model, representative dictionary and related tasks. As shown in Figure 2.1, when studying a task with a graph, we first model the given data with some graph signal model. The model describes data by capturing its important properties. Those properties can be obtained from observations, domain knowledge, or statistical learning algorithms. We then use a representative dictionary to represent the graph signal model. This lays a core idea of the thesis, which are essentially special cases that follow this general framework. In the following discussion, we go through each component one by one.

Graph Signal Models

In classical signal processing, people often work with signals with some specific properties, instead of arbitrary signals. For example, smooth signals have been studied extensively over decades; sparse signals are intensively studied recently. Here we also need a graph signal model to describe a class of graph signals with specific properties. In general, there are two approaches to mathematically model a graph signal, including a descriptive approach and a generative approach.

For the descriptive approach, we describe the properties of a graph signal by bounding the output of some operator. Let $\mathbf{x} \in \mathbb{R}^N$ be a graph signal, $f(\cdot)$ be a function operating on $\mathbf{x}$, we define a class of graph signals by restricting

$$f(\mathbf{x}) \leq C, \tag{2.14}$$

where C is some constant. For example, we define smooth graph signals by restricting $\mathbf{x}^T \, \mathrm{L} \, \mathbf{x}$ be small [82].

For the generative approach, we describe the properties of a graph signal by using a graph dictionary. Let $\mathrm{D} \in \mathbb{R}^{N \times S}$ be a graph dictionary whose columns represent elementary structure-related patterns. We define a class of graph signals by restricting

$$\mathbf{x} \; = \; \mathrm{D} \, \mathbf{a} \in \mathbb{R}^N,$$

where $\mathbf{a} \in \mathbb{R}^S$ is a vector of expansion coefficients. For the descriptive approach, we do not need to know everything about a graph signal, instead, we just need to its output of some operator; for the generative approach, we need to reconstruct a graph signal, which requires to know everything about a graph signal.

Graph Dictionary

For a certain graph signal model, we aim to find some dictionary to provide accurate and concise representations.

Design. In general, there are two approaches to design a graph dictionary, including a passive approach and a active one.

For the passive approach, the graph dictionary is designed only based on the graph structure; that is,

$$\mathrm{D} = g(\mathrm{A}) \in \mathbb{R}^{N \times S},$$

where $g(\cdot)$ is some function on the graph shift operator A. For example, D can be the eigenvector matrix of A, which is the graph Fourier basis. In classical signal processing, the Fourier basis, wavelet bases, wavelet frames, and Gabor frames are all constructed using this approach, where the graph is a line graph or a lattice graph [83].

For the active approach, the graph dictionary is designed based on both graph structure and a set of given graph signals; that is,

$$\mathrm{D} = g(\mathrm{A}, \mathrm{X}) \in \mathbb{R}^{N \times S},$$

where X is a matrix representation of a set of given graph signals. We can fit to those given graph signals and provide a specialized dictionary. Some related works see [42, 70].

Properties. The same class of graph signals can be modeled by various dictionaries. For example, whenever D is an identity matrix, it can represent arbitrary graph signals, but may not be appealing to represent a non-sparse signals. Depending on the application, we may have different requirements for the constructed graph dictionary. Here are some standard properties of a graph dictionary D, we aim to study.

- Frame bounds. For any $\mathbf{x}$ in a certain graph signal model,

$$\alpha_1 \left\|\mathbf{x}\right\|_2 \leq \left\|\mathrm{D}\,\mathbf{x}\right\|_2 \leq \alpha_2 \left\|\mathbf{x}\right\|_2,$$

  where α_1, α_2 are some constants;

- Sparse representations. For any $\mathbf{x} \in \mathbb{R}^N$ in a certain graph signal model, there exists a sparse coefficient $\mathbf{a} \in \mathbb{R}^M$ with $\left\|\mathbf{a}\right\|_0 \leq C$, which satisfies

$$\left\|\mathbf{x} - \mathrm{D}\,\mathbf{a}\right\|_2^2 \leq \epsilon,$$

  where C, ϵ are some constants;

- Uncertainty principles. For any $\mathbf{x}$ in a certain graph signal model, the following is satisfied

$$\left\|\mathbf{a}_1\right\|_0 + \left\|\mathbf{a}_2\right\|_0 \geq C,$$

  where $\left\|\mathbf{x} - \mathrm{D}_1\,\mathbf{a}_1\right\|_2^2 \leq \epsilon$ and $\left\|\mathbf{x} - \mathrm{D}_2\,\mathbf{a}_2\right\|_2^2 \leq \epsilon$, and $\mathrm{D} = \begin{bmatrix}\mathrm{D}_1 & \mathrm{D}_2\end{bmatrix}$.

Tasks

We mainly consider three standard tasks in signal processing, approximation, sampling followed with recovery and detection followed with localization. In this chapter, we will consider approximation. The latter two will be the focus of Parts III and IV.

Approximation. Approximation is a standard task to evaluate a representation. The goal is to use a few expansion coefficients to approximate a graph signal. We consider approximating a graph signal $\mathbf{x}$ by using a linear combination of a few atoms from D and solving the following sparse coding problem,

$$\begin{aligned} \mathbf{a}^* = \quad & \arg\min_{\mathbf{a}} \quad && d(\mathbf{x}', \mathbf{x}), \\ & \text{subject to :} && \mathbf{x}' = \mathrm{D}\,\mathbf{a}, \\ & && \left\|\mathbf{a}\right\|_0 \leq K, \end{aligned} \tag{2.15}$$

where $d(\cdot, \cdot)$ is some evaluation metric and K is a predefined sparsity level. The objective function measures the difference between the original signal and the approximated one, which evaluates how well the designed graph dictionary represents a given graph signal. The same formulation can also be used for denoising graph signals.

Sampling & Recovery. The goal is to recover an original graph signal from a few samples. We consider a general sampling and recovery setting. We consider any decrease in dimension via a linear operator as sampling, and, conversely, any increase in dimension via a linear operator

as recovery [84]. Let $\mathrm{F} \in \mathbb{R}^{N \times N}$ be a sampling pattern matrix, which is constrained by a given application and the sampling operator is

$$\Psi = \mathrm{C}\,\mathrm{F} \in \mathbb{R}^{M \times N},$$

where $\mathrm{C} \in \mathbb{R}^{M \times N}$ selects rows from F. For example, when we choose the kth row of F as the ith sample, the ith row of C is

$$\mathrm{C}_{i,j} = \begin{cases} 1, & j = k; \\ 0, & \text{otherwise.} \end{cases}$$

There are three sampling strategies: (1) uniform sampling when designing C, , where row indices are chosen from from $\{0, 1, \cdots, N-1\}$ independently and uniformly; and *experimentally design sampling*, where row indices can be chosen beforehand; and *active sampling*, where we will use feedback as samples are sequentially collected to decide the next row to be sampled. Each sampling strategy can be implemented by two approaches, including a random approach and a deterministic one. The sampling pattern matrix F constraints the following sampling patterns: when F is an identity matrix, Ψ is a subsampling operator; when F is a Gaussian random matrix, Ψ is a compressed sampling operator.

In the sampling phase, we take samples with the sampling operator Ψ,

$$\mathbf{y} = \Psi\mathbf{x} + \epsilon,$$

is a vector of samples and ϵ is noise. In the recovery phase, we reconstruct the graph signal by using a recovery operator Φ,

$$\mathbf{x}' = \Phi\mathbf{y},$$

where $\mathbf{x}'$ recovers $\mathbf{x}$ either exactly or approximately. The evaluation metric can be the mean square error or other metrics. Without any property of $\mathbf{x}$, it is hopeless to design an efficient sampling and recovery strategy. Here we focus on a special graph signal model, which can be described by a graph dictionary. The prototype of designing sampling and recovery strategies is

$$\begin{aligned} \Psi^*(\mathrm{D}), \Phi^*(\mathrm{D}) = \arg\min_{\Psi,\Phi} \max_{\mathbf{a}} \quad & d(\mathbf{x}', \mathbf{x}), \\ \text{subject to} \quad & \mathbf{x}' = \Phi\mathbf{y}, \\ & \mathbf{x} = \mathrm{D}\,\mathbf{a}, \end{aligned}$$

where $d(\cdot, \cdot)$ is some evaluation metric. The optimization problem shows that for any graph signal $\mathbf{x}$ generated based on the graph dictioanry D, we want to find a pair of sampling and recovery strategies to minimize the reconstruction error. The optimal sampling and recovery strategies Ψ^*, Φ^* are influenced by the given graph dictionary D. We often consider fixing either the sampling strategy or the recovery strategy and optimizing over the other one.

Detection & Localization. The goal is to identify targeted patterns in a graph signal. We consider targeted patterns are structure-related; that is, those structure-related patterns are generated from a graph dictionary,

$$\mathcal{X}_K = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \mathrm{D}\,\mathbf{a}, \mathbf{a} \in \mathbb{R}^S, \|\mathbf{a}\|_0 \leq K\},$$

A graph dictionary D may promote various structure-related patterns. For example, when D is an identity matrix, we detect impulses on graphs; when D is the graph Fourier basis, we detect activated graph frequencies.

Given a noisy observation of graph signal $\mathbf{y} \in \mathbb{R}^N$, graph signal detection aims to test the null against the alternative hypotheses:

$$\begin{aligned} H_0 &: \mathbf{y} \sim f(0, \epsilon), \\ H_1 &: \mathbf{y} \sim f(\mathbf{x}, \epsilon) \text{ with } \mathbf{x} \in \mathcal{X}_K. \end{aligned}$$

where ϵ is noise and $f(\cdot, \cdot)$ is a link function. For example, $f(\mathbf{x}, \epsilon) = \mathbf{x} + \mathbf{w} \in \mathbb{R}^N$ corresponds to a Gaussian noise model, where $\mathbf{w} \sim \mathcal{N}(0, \epsilon)$ is a Gaussian random variable; $f(\mathbf{x}, \epsilon) = \mathrm{Bernoulli}(\mathbf{x} + \epsilon) \in \mathbb{R}^N$ corresponds to a Bernoulli noise model, where each element $f(\mathbf{x}, \epsilon)_i$ is a vector of Bernoulli random variables with mean $(\mathbf{x} + \epsilon)_i$. Let this test be a mapping $T(\mathbf{y}) = \{0, 1\}$, where 1 indicates that we reject the null.

Given a noisy observation of graph signal $\mathbf{y} = f(\mathbf{x}, \epsilon) \in \mathbb{R}^N$, where $\mathbf{x} = \sum_{i=1}^{K} a_{\Omega_i} \mathbf{d}_{\Omega_i} \in \mathcal{X}_K$, graph signal detection aims to recover each activated atom, $\mathbf{d}_{\Omega_i}$. The prototype of designing localization strategies is to solve the following optimization problem

$$\begin{aligned} \mathbf{a}^* = \arg \min_{\|\mathbf{a}\|_0 \leq K} \quad & d(\mathbf{x}', \mathbf{y}), \\ \text{subject to} \quad & \mathbf{x}' = \mathrm{D}\,\mathbf{a}, \end{aligned}$$

where $d(\cdot, \cdot)$ is some evaluation metric. This formulation is similar to graph signal approximation; however, the goal is slightly different: localization aims to identify which atoms are activated (recover $\mathbf{a}$), and approximation aims to reconstruct the original graph signal (recover $\mathrm{D}\,\mathbf{a}$). When the size of graph dictionary grows exponentially with the size of graph, the graph dictionary cannot be explicitly expressed, which makes precise localization challenging.

2.5.1 Roadmap

	Smooth graph signal	Piecewise-constant graph signal	Piecewise-smooth graph signal
Representation	Section 3	Section 4	Section 5
Sampling	Sections 6 & 7		
Recovery	Section 8		
Detection		Section 9	
Localization		Section 10	

Table 2.1: Roadmap.

Table 2.1 presents a roadmap. We are going to study the following five tasks (rows): representation, sampling, recovery, detection and localization, for each of three types of graph signals (columns): smooth graph signals, piecewise-constant graph signals and piecewise-smooth graph signals. In Sections 3, 4 and 5, we consider the representations of smooth, piecewise-constant

and piecewise-smooth graph signals, respectively. In Sections 6 and 7, we consider the sampling strategies for smooth graph signals. In Section 8, we consider the recovery strategies for smooth graph signals. In Section 9, we consider the detection strategies for piecewise-constant graph signals. In Section 10, we consider the localization strategies for piecewise-constant graph signals.

Chapter 3

Representations of Smooth Graph Signals

3.1 Introduction

In signal processing, smoothness means adjacent signal coefficients have similar values. This concept is widely used in numerous applications. For example, in the task of denoising, we smooth a signal/image to create an approximating function that captures important patterns and leave out noise.

Mathematically, various smoothness criteria are defined, typically with respect to some norm. For example, the Sobolev space is a vector space of functions equipped with a norm that is a combination of L_p-norms of the function itself and its derivatives up to a given order, the Lipschitz space is a vector space of functions equipped with the Lipschitz norm, and the Besov space a vector space of functions equipped with the Besov norm [85]. A relaxed smoothness criterion is bounded variation. A continuous function has bounded variation if the length of its graph on any finite interval is finite [84]. Certain natural representations from harmonic analysis are the optimal representation for objects in the corresponding functional classes. For example, Fourier series are optimal representations for L_2 Sobolev classes, wavelets are optimal representations for L_p Sobolev, Hölder, Triebel and Besov classes [86].

In this chapter, we introduce smoothness criteria for graph signals in a similar vein. We then discuss the corresponding representation for each smoothness criterion. Similar concepts are proposed or hinted in many previous works [1, 80]. The goal here is to provide a summary and comparison.

3.2 Graph Signal Models

We introduce four smoothness criteria for graph signals. We start with the pairwise Lipschitz smooth criterion.

Definition 1. A graph signal $\mathbf{x}$ with unit norm is *pairwise Lipschitz smooth* with parameter C when it satisfies

$$|x_i - x_j| \;\leq\; C\, d(v_i, v_j), \text{ for all } i, j = 0, 1, \ldots, N-1,$$

with $d(v_i, v_j)$ the distance between the ith and the jth nodes.

We can choose the geodesic distance, the diffusion distance [87], or some other distance metric for $d(\cdot,\cdot)$. Similarly to the traditional Lipschitz criterion [88], the pairwise Lipschitz smoothness criterion emphasizes pairwise smoothness, which zooms into the difference between each pair of adjacent nodes.

Definition 2. A graph signal $\mathbf{x}$ with unit norm is *total Lipschitz smooth* with parameter C when it satisfies

$$\sum_{(i,j)\in\mathcal{E}} \mathrm{W}_{i,j}(x_i - x_j)^2 \;\leq\; C,$$

where $\mathrm{W} \in \mathbb{R}^{N\times N}$ is the adjacency matrix.

The total Lipschitz smoothness criterion generalizes the pairwise Lipschitz smoothness criterion while still emphasizing pairwise smoothness, but in a less restricted manner; it is also known as the Laplacian smoothness criterion [89].

Definition 3. A graph signal $\mathbf{x}$ with unit norm is *local normalized neighboring smooth* with parameter C when it satisfies

$$\sum_i \left(x_i - \frac{1}{\sum_{j:(i,j)\in\mathcal{E}} \mathrm{W}_{i,j}} \sum_{j:(i,j)\in\mathcal{E}} \mathrm{W}_{i,j}\, x_j \right)^2 \;\leq\; C.$$

The local normalized neighboring smoothness criterion compares each node to the local normalized average of its immediate neighbors.

Definition 4. A graph signal $\mathbf{x}$ with unit norm is *global normalized neighboring smooth* with parameter C when it satisfies

$$\sum_i \left(x_i - \frac{1}{|\lambda_{\max}^{(\mathrm{W})}|} \sum_{j:(i,j)\in\mathcal{E}} \mathrm{W}_{i,j}\, x_j \right)^2 \;\leq\; C.$$

The global normalized neighboring smoothness criterion compares each node to the global normalized average of its immediate neighbors. The difference between the local normalized neighboring smoothness criterion and the global normalized neighboring smoothness criterion is the normalization factor. For the local normalized neighboring smoothness criterion, each node has its own normalization factor; for the global normalized neighboring smoothness criterion, all nodes have the same normalization factor.

The four criteria quantify smoothness in different ways: the pairwise and the total Lipschitz ones focus on the variation of two signal coefficients connected by an edge with the pairwise Lipschitz one more restricted, while the local and global neighboring smoothness criterion focus on comparing a node to the average of its neighbors.

3.3 Graph Dictionary Construction

As shown in (2.14), The graph signal models in Definitions 1, 2, 3, 4 are introduced in a descriptive approach. Following these, we are going to translate the descriptive approach into a generative approach; that is, we represent the corresponding signal classes satisfying each of the four criteria by some representative graph dictionary.

3.3.1 Design

We first construct polynomial graph signals that satisfy the Lipschitz smoothness criterion.

Definition 5. A graph signal $\mathbf{x}$ is polynomial with degree K when

$$\mathbf{x} \;=\; \mathrm{D}_{\mathrm{poly(K)}}\,\mathbf{a} \;=\; \begin{bmatrix}\mathbf{1} & \mathrm{D}^{(1)} & \mathrm{D}^{(2)} & \dots & \mathrm{D}^{(K)}\end{bmatrix}\mathbf{a} \in \mathbb{R}^N,$$

where $\mathbf{a} \in \mathbb{R}^{KN+1}$ and $\mathrm{D}_{\mathrm{poly(K)}}$ is a graph polynomial dictionary with $\mathrm{D}^{(k)}_{i,j} = d^k(v_i, v_j)$. Denote this class by $\mathrm{PL}(K)$.

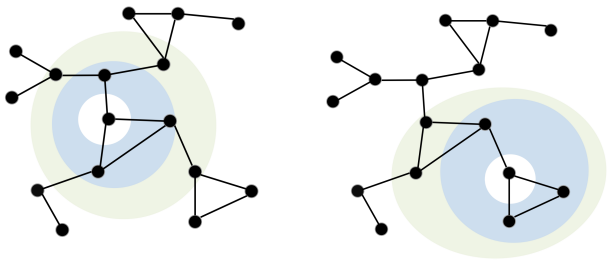


Figure 3.1: Different origins lead to different coordinate systems; white, blue, and green denote the origin, nodes with geodesic distance 1 from the origin, and nodes with geodesic distance 2 from the origin, respectively.

In classical signal processing, polynomial time signals can be expressed as $x_n = \sum_{k=0}^{K} a_k n^k$, $n = 1, \dots, N$; we can rewrite this as in the above definition as $\mathbf{x} = \mathrm{D}_K\,\mathbf{a}$, with $(D_K)_{n,k} = n^k$. The columns of D_K are denoted as $\mathrm{D}^{(k)}$, $k = 0, \dots, K$, and called *atoms*; the elements of each atom $\mathrm{D}^{(k)}$ are n^k. Since polynomial time signals are shift-invariant, we can set any time point as the origin; such signals are thus characterized by $K+1$ degrees of freedom a_k, $k = 0, \dots, K$. This is not true for graph signals, however; they are not shift-invariant and any node can serve as the origin (see Figure 3.1). In the above definition, $\mathrm{D}^{(k)}$ are now matrices with the number of atoms equal to the number of nodes N (with each atom corresponding to the node serving as the origin). The dictionary D_K thus contains $KN+1$ atoms.

We now show that graph polynomial dictionary represents the pairwise Lipschiz smooth signals.

Theorem 1. $\mathrm{PL}(1)$ is a subset of the pairwise Lipschitz smooth with some parameter C.

Proof. Let $\mathbf{x} \in \mathrm{PL}(1)$, that is,

$$\mathbf{x} = \begin{bmatrix}\mathbf{1} & \mathrm{D}^{(1)}\end{bmatrix}\mathbf{a},$$

Then, we write the pairwise Lipschitz smooth criterion as

$$\begin{aligned} |x_i - x_j| \;&=\; |\sum_k \left(d(v_k, v_i) - d(v_k, v_j)\right) a_k| \\ &\le\; \sum_k |d(v_k, v_i) - d(v_k, v_j)||a_k| \\ &\le\; \sum_k |a_k||d(v_i, v_j)| \;=\; \|\mathbf{a}\|_1\,|d(v_i, v_j)|. \end{aligned}$$

The parameter $C = \|\mathbf{a}\|_1$, which corresponds to the energy of the original graph signal. □

We now construct bandlimited signals that satisfy the total Lipschitz, local normalized neighboring and global normalized neighboring smoothness smoothness criteria.

Definition 6. A graph signal $\mathbf{x}$ is bandlimited with respect to a graph Fourier basis V with bandwidth K when

$$\mathbf{x} = \mathrm{V}_{(K)}\, \mathbf{a},$$

where $\mathbf{a} \in \mathbb{R}^K$ and $\mathrm{V}_{(K)}$ is a submatrix containing the first K columns of V. Denote this class by $\mathrm{BL}_{\mathrm{V}}(K)$ [90].

When V is the eigenvector matrix of the unnormalized graph Laplacian matrix, we denote it as V_{L} and can show that signals in $\mathrm{BL}_{\mathrm{V}_{\mathrm{L}}}(K)$ are total Lipschitz smooth; when V is the eigenvector matrix of the transition matrix, we denote it as V_{P} and can show that signals in $\mathrm{BL}_{\mathrm{V}_{\mathrm{P}}}(K)$ are local normalized neighboring smooth; when V is the eigenvector matrix of the weighted adjacency matrix, we denote it as V_{W} and can show that signals in $\mathrm{BL}_{\mathrm{V}_{\mathrm{W}}}(K)$ are global normalized neighboring smooth.

We now show that the graph Fourier basis of the graph Laplacian represents the total Lipschiz smooth signals.

Theorem 2. For any $K \in \{1, \cdots, N\}$, $\mathrm{BL}_{\mathrm{V}_{\mathrm{L}}}(K)$ is a subset of the total Lipschitz smooth with parameter C, when $C \geq \lambda_k^{(\mathrm{L})}$.

Proof. Let $\mathbf{x}$ be a graph signal with bandwidth K, that is,

$$\mathbf{x} = \sum_{k=1}^{K} \widehat{x}_k \mathbf{v}_k^{(\mathrm{L})},$$

Then, we write the total Lipschitz smooth criterion as

$$\begin{aligned}
&\sum_{(i,j)\in\mathcal{E}} \mathrm{W}_{i,j}(x_i - x_j)^2 \;=\; \mathbf{x}^T\, \mathrm{L}\, \mathbf{x} \\
=\;& (\sum_{k=1}^{K} \widehat{x}_k \mathbf{v}_k^{(\mathrm{L})})^T (\sum_{k=0}^{K-1} \widehat{x}_k \lambda_k \mathbf{v}_k^{(\mathrm{L})}) \;=\; \sum_{k=1}^{K} \lambda_k^{(\mathrm{L})} \widehat{x}_k^2 \\
\leq\;& \lambda_K^{(\mathrm{L})} \sum_{k=1}^{K} \widehat{x}_k^2 \;=\; \lambda_K^{(\mathrm{L})}.
\end{aligned}$$

□

We now show that the graph Fourier basis of the transition matrix represents the local normalized neighboring smooth signals.

Theorem 3. For any $K \in \{1, \cdots, N\}$, $\mathrm{BL}_{\mathrm{V}_{\mathrm{P}}}(K)$ is a subset of the local normalized neighboring smooth with parameter C, when $C \geq (1 - \lambda_K^{(\mathrm{P})})^2$.

Proof. Let $\mathbf{x}$ be a graph signal with bandwidth K, that is,

$$\mathbf{x} = \sum_{k=1}^{K} \widehat{x}_k \mathbf{v}_k^{(\mathrm{P})},$$

Then, we write the local normalized neighboring smooth criterion as

$$\begin{aligned}
& \left| x_i - \frac{1}{\sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}} \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\, x_j \right| \\
= & \left| \left(\sum_{k=1}^{K} \widehat{x}_k \mathbf{v}_k^{(\mathrm{P})} \right)_i - \sum_{j\in\mathcal{N}_i} \mathrm{P}_{i,j} \left(\sum_{k=1}^{K} \widehat{x}_k \mathbf{v}_k^{(\mathrm{P})} \right)_j \right| \\
= & \left| \sum_{k=1}^{K} \widehat{x}_k \left((\mathbf{v}_k^{(\mathrm{P})})_i - \sum_{j\in\mathcal{N}_i} \mathrm{P}_{i,j} (\mathbf{v}_k^{(\mathrm{P})})_j \right) \right| \\
= & \left| \sum_{k=1}^{K} \widehat{x}_k (1-\lambda_k) (\mathbf{v}_k^{(\mathrm{P})})_i \right| \\
\leq & \ (1-\lambda_K^{(\mathrm{P})}) \left| \sum_{k=0}^{K-1} \widehat{x}_k (\mathbf{v}_k^{(\mathrm{P})})_i \right| \ = \ (1-\lambda_K^{(\mathrm{P})}) |x_i|.
\end{aligned}$$

The last equality follows from the fact that $\mathbf{v}_k^{(\mathrm{P})}$ and $\lambda_k^{(\mathrm{P})}$ are eigenvectors and eigenvalues of P.

$$\begin{aligned}
& \sum_i \left(x_i - \frac{1}{\sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}} \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\, x_j \right)^2 \\
= & \sum_{i=1}^{N} |x_i - \frac{1}{\sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}} \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\, x_j|^2 \\
\leq & \sum_{i=0}^{N-1} (1-\lambda_K^{(\mathrm{P})})^2 |x_i|^2 \ = \ (1-\lambda_K^{(\mathrm{P})})^2.
\end{aligned}$$

□

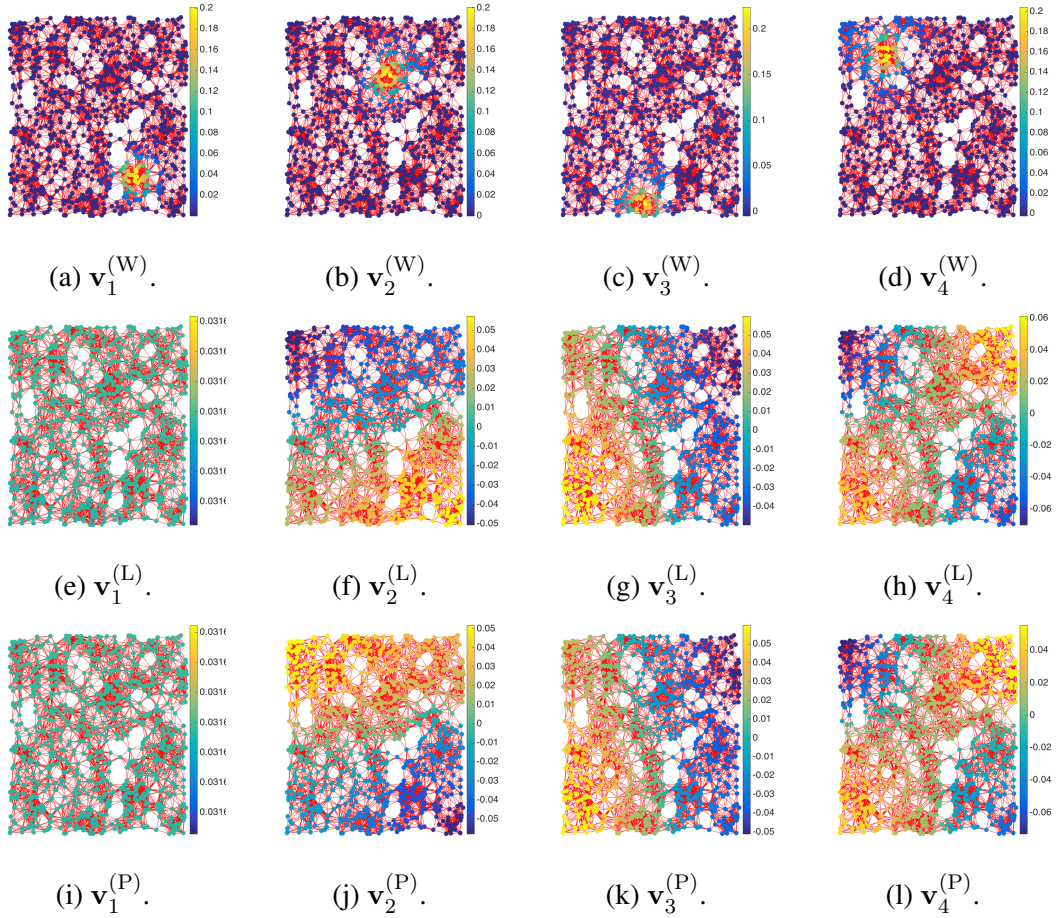
We now show that the graph Fourier basis of the adjacency matrix represents the global normalized neighboring smooth signals.

Theorem 4. For any $K \in \{1, \cdots, N\}$, $\mathrm{BL}_{\mathrm{V_W}}(K)$ is a subset of the global normalized neighboring smooth with parameter C, when $C \geq (1-\lambda_K^{(\mathrm{W})}/|\lambda_{\max}(\mathrm{W})|)^2$.

The proof is similar to Theorem 3. Note that for graph Laplacian, the eigenvalues are sorted in an ascending order; for the transition matrix and the adjacency matrix, the eigenvalues are sorted in a descending order.

Each of these three models generates smooth graph signals according to one of the four criteria in Definitions 1, 2, 3 and 4: $\mathrm{PL}(K)$ models Lipschitz smooth signals; $\mathrm{BL}_{\mathrm{V_L}}(K)$ models

total Lipschitz smooth signals; $\mathrm{BL}_{\mathrm{V_P}}(K)$ with models the local normalized neighboring smooth signals; and $\mathrm{BL}_{\mathrm{V_W}}(K)$ models the global normalized neighboring smooth signals; the corresponding graph representation dictionaries are $\mathrm{D}_{\mathrm{poly(K)}}$, $\mathrm{V_L}$, $\mathrm{V_P}$, and $\mathrm{V_W}$.



(a) $\mathbf{v}_1^{(\mathrm{W})}$. (b) $\mathbf{v}_2^{(\mathrm{W})}$. (c) $\mathbf{v}_3^{(\mathrm{W})}$. (d) $\mathbf{v}_4^{(\mathrm{W})}$.

(e) $\mathbf{v}_1^{(\mathrm{L})}$. (f) $\mathbf{v}_2^{(\mathrm{L})}$. (g) $\mathbf{v}_3^{(\mathrm{L})}$. (h) $\mathbf{v}_4^{(\mathrm{L})}$.

(i) $\mathbf{v}_1^{(\mathrm{P})}$. (j) $\mathbf{v}_2^{(\mathrm{P})}$. (k) $\mathbf{v}_3^{(\mathrm{P})}$. (l) $\mathbf{v}_4^{(\mathrm{P})}$.

Figure 3.2: Graph Fourier bases of a geometric graph. $\mathrm{V_W}$ localizes in some small regions; $\mathrm{V_L}$ and $\mathrm{V_P}$ have similar behaviors.

3.3.2 Properties

We next study the properties of graph representation dictionaries for smooth graph signals, especially for graph Fourier bases $\mathrm{V_L}$, $\mathrm{V_P}$, and $\mathrm{V_W}$. We first visualize them in Figures 3.2. Figure 3.2 compares the first four graph Fourier basis vectors of $\mathrm{V_L}$, $\mathrm{V_P}$ and $\mathrm{V_W}$ in a geometric graph. We see that $\mathrm{V_W}$ tends to localize in some small regions; $\mathrm{V_L}$ and $\mathrm{V_P}$ have similar behaviors.

We then check the properties mentioned in Section II.

Frame Bound. Graph polynomial dictionary is highly redundant and the frame bound is loose. When the graph is undirected, the adjacency matrix is symmetric, then $\mathrm{V_L}$ and $\mathrm{V_W}$ are orthonormal. It is hard to draw any meaningful conclusion when the graph is directed; we leave it for the future work.

Sparse Representations. Graph polynomial dictionary provides sparse representations for polynomial graph signals. On the other hand, the graph Fourier bases provide sparse representations for the bandlimited graph signals. For approximately bandlimited graph signals, there are some residuals coming from the high-frequency components.

Uncertainty Principles. In classical signal processing, it is well known that signals cannot localize in both time and frequency domains at the same time [84, 91]. Some previous works extend this uncertainty principle to graphs by studying how well a graph signal exactly localize in both the graph vertex and graph spectrum domain [44, 45]. Here we study how well a graph signal approximately localize in both the graph vertex and graph spectrum domain. We will see that the localization depends on the graph Fourier basis.

Definition 7. A graph signal $\mathbf{x}$ is *ϵ-vertex concentrated* on a graph vertex set Γ when it satisfies

$$\|\mathbf{x} - \mathbf{I}_\Gamma \, \mathbf{x}\|_2^2 \leq \epsilon,$$

where $\mathbf{I}_\Gamma \in \mathbb{R}^{N \times N}$ is a diagonal matrix, with $(\mathbf{I}_\Gamma)_{i,i} = 1$ when $i \in \Gamma$ and 0, otherwise.

The vertex set Γ represents a region that supports the main energy of signals. When $|\Gamma|$ is small, a ϵ-vertex concentrated signal is approximately sparse.

Definition 8. A graph signal $\mathbf{x}$ is *ϵ-spectrum concentrated* on a graph spectrum band Ω when it satisfies

$$\|\mathbf{x} - \mathrm{V}_\Omega \, \mathrm{U}_\Omega \, \mathbf{x}\|_2^2 \leq \epsilon,$$

where $\mathrm{V}_\Omega \in \mathbb{R}^{N \times |\Omega|}$ is a submatrix of V with columns selected by Ω and $\mathrm{U}_\Omega \in \mathbb{R}^{|\Omega| \times N}$ is a submatrix of V with rows selected by Ω.

The graph spectrum band Ω provides a bandlimited space that supports the main energy of signals. An equivalent formulation is $\|\widehat{\mathbf{x}} - \mathbf{I}_\Omega \, \widehat{\mathbf{x}}\|_2^2 \leq \epsilon$. Definition 8 is a simpler version of the approximately bandlimited space in [92].

We next show an uncertainty principle of the graph vertex and spectrum domains.

Theorem 5. Let a unit norm signal $\mathbf{x}$ supported on an undirected graph be ϵ_Γ-vertex concentrated and ϵ_Ω-spectrum concentrated at the same time. Then,

$$|\Gamma| \cdot |\Omega| \geq \frac{(1 - (\epsilon_\Omega + \epsilon_\Gamma))^2}{\|\mathrm{U}_\Omega\|_\infty^2}.$$

Proof. We first show $\|\mathrm{V}_\Omega \, \mathrm{U}_\Omega \, \mathbf{I}_\Gamma\|_2 \leq \|\mathrm{U}_\Omega\|_\infty \sqrt{|\Gamma| \cdot |\Omega|}$.

$$\begin{aligned}(\mathrm{V}_\Omega \, \mathrm{U}_\Omega \, \mathbf{I}_\Gamma \, \mathbf{x})_s \; &= \; \sum_{k \in \Omega} \mathrm{V}_{s,k} \left(\sum_{i \in \Gamma} \mathrm{U}_{k,i} \, x_i \right) \\ = \; \sum_{i \in \Gamma} \left(\sum_{k \in \Omega} \mathrm{V}_{s,k} \, \mathrm{U}_{k,i} \right) x_i \; &= \; \sum_i q(s,i) x_i,\end{aligned}$$

where

$$q(s,i) = \begin{cases} \sum_{k \in \Omega} \mathrm{V}_{s,k} \, \mathrm{U}_{k,i}, & i \in \Gamma; \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathbf{y}^{(i)}$ be a graph signal with $y_s^{(i)} = q(s,i)$. Then, $(\widehat{\mathbf{y}^{(i)}})_k = \mathbf{1}_{k\in\Omega}\,\mathrm{U}_{k,i}$. We then have

$$\begin{aligned}
& \|\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\|_2^2 \;\le\; \|\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\|_{\mathrm{HS}}^2 \\
= \;& \sum_{i\in\Gamma}\sum_{s}|q(s,i)|^2 \;=\sum_{i\in\Gamma}\left\|\mathbf{y}^{(i)}\right\|_2^2 \\
= \;& \sum_{i\in\Gamma}\left\|\widehat{\mathbf{y}^{(i)}}\right\|_2^2 \;=\; \sum_{i\in\Gamma}\sum_{k}\left(\mathbf{1}_{k\in\Omega}\,\mathrm{U}_{k,i}\right)^2 \\
\le \;& \|\mathrm{U}_\Omega\|_\infty^2\sum_{i\in\Gamma}\sum_{k}\mathbf{1}_{k\in\Omega} \;=\; \|\mathrm{U}_\Omega\|_\infty^2\,|\Gamma|\cdot|\Omega|.
\end{aligned}$$

We then show that $\|\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\,\mathbf{x}\|_2 \ge 1-(\epsilon_\Omega+\epsilon_\Gamma)$. Based on the assumption, we have

$$\begin{aligned}
& \|\mathbf{x}-\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\,\mathbf{x}\|_2 \\
= \;& \|\mathbf{x}-\mathbf{I}_\Gamma\,\mathbf{x}\|_2+\|\mathbf{I}_\Gamma\,\mathbf{x}-\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\,\mathbf{x}\|_2 \\
\le \;& \epsilon_\Omega+\epsilon_\Gamma.
\end{aligned}$$

Since $\mathbf{x}$ has a unit norm, by the triangle inequality, we have

$$\|\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\|_2 \ge 1-(\epsilon_\Omega+\epsilon_\Gamma).$$

Finally, we combine two results and obtain

$$|\Gamma|\cdot|\Omega| \ge \frac{\|\mathrm{V}_\Omega\,\mathrm{U}_\Omega\,\mathbf{I}_\Gamma\|_2^2}{\|\mathrm{U}_\Omega\|_\infty^2} \;>\; \frac{(1-(\epsilon_\Omega+\epsilon_\Gamma))^2}{\|\mathrm{U}_\Omega\|_\infty^2}$$

□

We see that the lower bound involves with the maximum magnitude of U_Ω. In classical signal processing, U is the discrete Fourier transform matrix, so $\|\mathrm{U}_\Omega\|_\infty = 1/\sqrt{N}$; the lower bound is $O(N)$ and signals cannot localize in both time and frequency domain. However, for complex and irregular graphs, the energy of a graph Fourier basis vector may concentrate on a few elements, that is, $\|\mathrm{U}_\Omega\|_\infty = O(1)$, as shown in Figures 3.2(a)(b)(c)(d). It is thus possible that graph signals can be localized in both the vertex and spectrum domain. We now illustrate this localization phenomenon

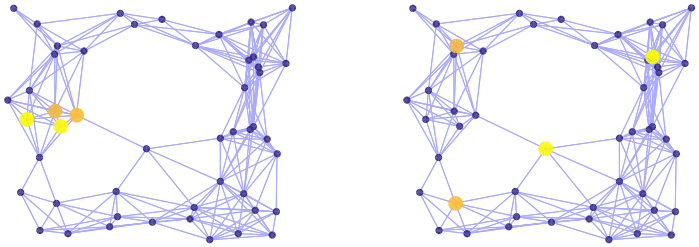
Localization Phenomenon. A part of this section has been shown in [93]. We show it here for the completeness. The *localization* of a graph signal means that most elements in a graph signal are zeros, or have small values; only a small number of elements have large magnitudes and the corresponding nodes are clustered in one subgraph with a small diameter.

Prior work uses *inverse participation ratio* (IPR) to quantify localization [94]. The IPR of a graph signal $\mathbf{x}\in\mathbb{R}^N$ is

$$\mathrm{IPR} = \frac{\sum_{i=1}^N x_i^4}{(\sum_{i=1}^N x_i^2)^2}.$$

A large IPR indicates that $\mathbf{x}$ is localized, while a small IPR indicates that $\mathbf{x}$ is not localized. The range of IPR is from 0 to 1. For example, $\mathbf{x} = [1/\sqrt{N}, 1/\sqrt{N}, \cdots, 1/\sqrt{N}]^T$ is the most

delocalized vector with IPR $= 1/N$, while $\mathbf{x} = [1, 0, \cdots, 0]^T$ is the most localized vector with IPR $= 1$. IPR has some shortcomings: a) IPR only promotes sparsity, that is, a high IPR does not necessarily mean that the nonzero elements concentrate in a clustered subgraph, which is the essence of localization (Figure 3.3); b) IPR does not work well for large-scale datasets. When N is large, even if only a small set of elements are non zero, IPR tends to be small.



(a) clustered graph signal. (b) unclustered graph signal.

Figure 3.3: Sparse graph signal. Colored nodes indicate large nonzero elements.

To solve this, we propose a novel measure to quantify the localization of a graph signal. We use *energy concentration ratio* (ECR) to quantify the energy concentration property. The ECR is defined as

$$\mathrm{ECR} = \frac{\mathrm{S}^*}{\mathrm{N}},$$

where S^* is the smallest S that satisfies $\|\mathbf{x}_S\|_2^2 \geq 95\% \|\mathbf{x}\|_2^2$, with $\mathbf{x}_S$ the first S elements with the largest magnitude in $\mathbf{x}$. This indicates that 95% energy of a graph signal is concentrated in the first S^* elements with largest magnitude. ECR ranges from 0 to 1: when the signal is energy-concentrated, the ECR is small; when the energy of the signal is evenly distributed, the ECR is 1.

We next use *normalized geodesic distance* (NGD) to quantify the clustered property. Let $\mathcal{M}$ be the set of nodes that possesses 95% energy of the whole signal. The normalized geodesic distance is defined as:

$$\mathrm{NGD} = \frac{1}{D} \frac{\sum_{i,j \in \mathcal{M}, i \neq j} d(v_i, v_j)}{n(n-1)/2},$$

where D is the diameter of the graph, $d(v_i, v_j)$ is the geodesic distance between nodes i and j. Here we use the normalized average geodesic distance as a measure to determine whether the nodes are highly connected. We use the average geodesic distance instead of the largest geodesic distance to avoid the influence of outliers. The NGD ranges from 0 to 1: when the nodes are clustered in a small subgraph, the NGD is small; when the nodes are dispersive, the NGD is large.

We use ECR and NGD together to determine the localization of graph signals. When the two measures are small, the energy of the signal is concentrated in a small set of highly connected nodes, which can be interpreted as localization.

Each graph Fourier basis vector is regarded as a graph signal. When most basis vectors in the graph Fourier basis are localized, that is, the corresponding ECRs and NGDs are small, we call that graph Fourier basis localized.

We now investigate the localization phenomenon of graph Fourier bases of several real-world networks, including the arXiv general relativity and quantum cosmology (GrQc) collaboration network [95], arXiv High Energy Physics - Theory (Hep-Th) collaboration network [95] and the Facebook 'friend circles' network [95]. We find similar localization phenomena among different datasets. Due to the limited space, we only show the result of arXiv GrQc collaboration network. The arXiv GrQc network represents the collaborations between authors based on the submitted papers in general relativity and quantum cosmology category of arXiv. When the author i and author j coauthored a paper, the graph contains an undirected edge between node i and j. The graph contains 5242 nodes and 14496 undirected edges. Since the graph is not connected, we choose a connect component with 4158 nodes and 13422 edges.

We investigate the Fourier bases of the weighted adjacency matrix, the transition matrix and the unnormalized graph Laplacian matrix in the arXiv GrQc network. Figure 3.4 illustrates the ECRs and NGDs of the first 50 graph Fourier basis vectors (low-frequency components) and the last 50 graph Fourier basis vectors (high-frequency components) of the three graph representation matrices, where the ECR and NGD are plotted as a function of the index of the corresponding graph Fourier basis vectors. We find that a large number of graph Fourier basis vectors has small ECRs and NGDs, which indicates that graph Fourier basis vectors of various graph representation matrices are localized. Among various graph representation matrices, the graph Fourier basis vectors of graph Laplacian matrix tend to be more localized, especially in high-frequency components. In low-frequency components, the graph Fourier basis vectors of adjacency matrix are more localized.

To combine the uncertainty principle previously, when the graph Fourier basis shows localization phenomenon, it is possible that a graph signal can be localized in both graph vertex and spectrum domain. Based the graph Fourier basis of the graph Laplacian, a high-frequency bandlimited signals can be well localized in the graph vertex domain; based on the graph Fourier basis of adjacency matrix, a low-frequency bandlimited signals can be well localized in the graph vertex domain. The Fourier transform is famous for capturing the global behaviors and works as a counterpart of the delta functions; however, this may not be true on graphs. The study of new role of the graph Fourier transform will be an interesting future work.

3.3.3 Tasks

As mentioned in Section II, we focus on two tasks: approximation and sampling following with recovery.

Approximation

We compare the graph Fourier bases based on different graph structure matrices.

Algorithm. We consider nonlinear approximation for the graph Fourier bases, that is, after expanding with a representation, we should choose the K largest-magnitude expansion coefficients so as to minimize the approximation error. Let $\{\phi_k \in \mathbb{R}^N\}_{k=1}^N$ and $\{\widehat{\phi}_k \in \mathbb{R}^N\}_{k=1}^N$ be a

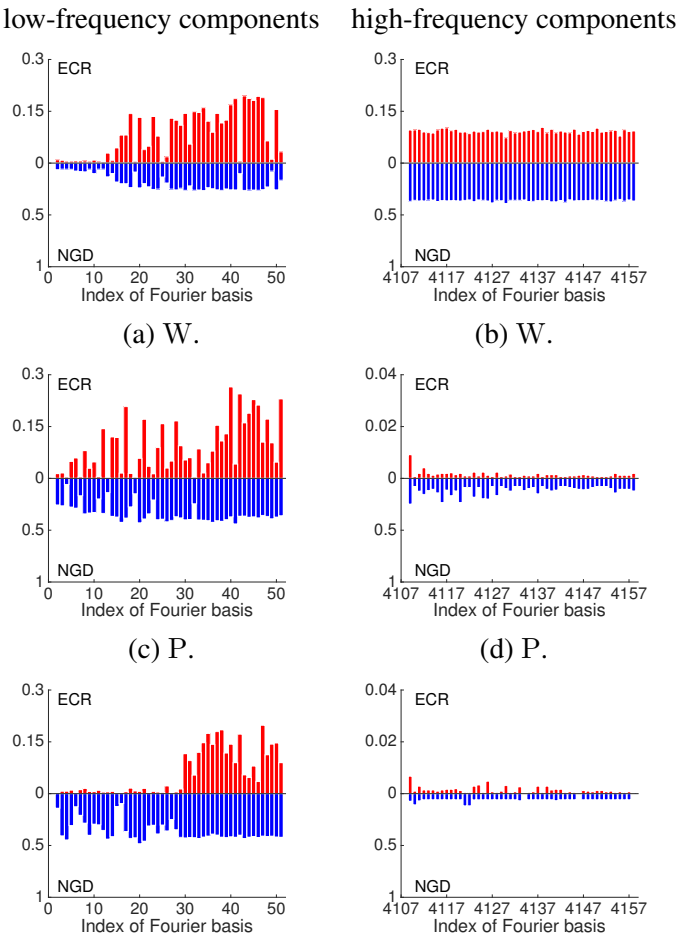


Figure 3.4: Localization of graph Fourier bases of various graph representation matrices in the arXiv GrQc network. In low-frequency components, Fourier basis vectors of the adjacency matrix are localized; in high-frequency components, Fourier basis vectors of graph Laplacian matrix are localized.

pair of biorthonormal basis and $\mathbf{x} \in \mathbb{R}^N$ be a signal. Here the graph Fourier transform matrix $\mathrm{U} = \{\widehat{\phi}_k\}_{k=1}^N$ and the graph Fourier basis $\mathrm{V} = \{\phi_k\}_{k=1}^N$. The nonlinear approximation to $\mathbf{x}$ is

$$\mathbf{x}^* = \sum_{k \in \mathcal{I}_K} \left\langle \mathbf{x}, \widehat{\phi}_k \right\rangle \phi_k, \tag{3.1}$$

where $\mathcal{I}_K$ is the index set of the K largest-magnitude expansion coefficients. When a basis promotes sparsity for $\mathbf{x}$, only a few expansion coefficients are needed to obtain a small approximation error. Note that (3.1) is a special case of (2.15) when the distance metric $d(\cdot, \cdot)$ is the ℓ_2 norm and D is a basis.

Since the the graph polynomial dictionary is redundant, we solve the following sparse coding

problem,

$$\begin{aligned} \mathbf{x}^* = \quad & \arg\min_{\mathbf{a}} \quad \left\|\mathbf{x} - \mathrm{D}_{\mathrm{poly}(2)}\,\mathbf{a}\right\|_2^2, \\ & \text{subject to :} \quad \|\mathbf{a}\|_0 \leq K, \end{aligned} \tag{3.2}$$

where $\mathrm{D}_{\mathrm{poly}(2)}$ is the graph polynomial dictionary with order 2 and $\mathbf{a}$ are expansion coefficients. The idea is to use a linear combination of a few atoms from $\mathrm{D}_{\mathrm{poly}(2)}$ to approximate the original signal. When D is an orthonormal basis, the closed-form solution is exactly (3.1). We solve (2.15) by using the orthogonal matching pursuit, which is a greedy algorithm [96]. Note that (3.2) is a special case of (2.15) when the distance metric $d(\cdot,\cdot)$ is the ℓ_2 norm.

3.4 Experimental Results

We test the four representations on two datasets, including the Minnesota road graph [97] and the U.S city graph [98].

The Minnesota road graph is a standard dataset including 2642 intersections and 3304 roads [97]. we construct a graph by modeling the intersections as nodes and the roads as undirected edges. We collect a dataset recording the wind speeds at those 2642 intersections [99]. The data records the hourly measurement of the wind speed and direction at each intersection. In this paper, we present the data of wind speed on January 1st, 2015. Figure 3.5(a) shows a snapshot of the wind speeds on the entire Minnesota road. The expansion coefficients obtained by using four representations are shown in Figure 3.5(b), (c), (d) and (e). The energies of the frequency coefficients of V_{W}, V_{L} and V_{P} mainly concentrate on the low-frequency bands; V_{L} and V_{P} are more concentrated; $\mathrm{D}_{\mathrm{poly}(2)}$ is redundant and the corresponding expansion coefficients $\mathrm{D}_{\mathrm{poly}(2)}^T\,\mathbf{x}$ are not sparse.

To make a more serious comparison, we evaluate the approximation error by using the normalized mean square error, that is,

$$\text{Normalized MSE} = \frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}, \tag{3.3}$$

where $\mathbf{x}^*$ is the approximation signal and $\mathbf{x}$ is the original signal. Figure 3.5(f) shows the approximation errors given by the four representations. The x-axis is the number of coefficients used in approximation, which is K in (3.1) and (2.15) and the y-axis is the approximation error, where lower means better. We see that V_{L} and $\mathrm{D}_{\mathrm{poly}(2)}$ tie V_{P}; all of them are much better than V_{W}. This means that the wind speeds on the Minnesota road graph are well modeled by pairwise Lipschitz smooth, total Lipschitz smooth and local normalized neighboring smooth criteria. The global normalized neighboring smooth criterion is not appropriate for this dataset.

The U.S weather station graph is a network representation of 150 weather stations across the U.S. We assign an edge when two weather stations are within 500 miles. The graph includes 150 nodes and 1033 undirected, unweighted edges. Each weather station has 365 days of recordings (one recording per day), for a total of 365 graph signals. As an example, see Figure 3.6(a). The expansion coefficients obtained by using four representations are shown in Figure 3.6(b), (c), (d) and (e). Similarly to the wind speed dataset, the energies of the frequency coefficients of V_{W}, V_{L}

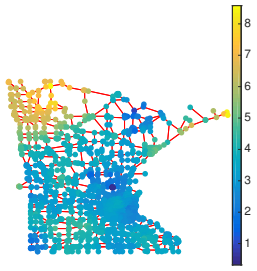
and $\mathrm{V_P}$ are mainly concentrated on the low-frequency bands; $\mathrm{V_L}$ and $\mathrm{V_P}$ are more concentrated; $\mathrm{D_{poly(2)}}$ is redundant and the corresponding expansion coefficients $\mathrm{D}_{\mathrm{poly(2)}}^T \mathbf{x}$ are not sparse.

The evaluation metric of the approximation error is also the normalized mean square error. Figure 3.6(f) shows the approximation errors given by the four representations. The results are averages over 365 graph signals. Again, we see that $\mathrm{V_L}$, $\mathrm{D_{poly(2)}}$ and $\mathrm{V_P}$ perform similarly; all of them are much better than $\mathrm{V_W}$. This means that the wind speeds on the Minnesota road graph are well modeled by pairwise Lipschitz smooth, total Lipschitz smooth and local normalized neighboring smooth criteria. The global normalized neighboring smooth criterion is not appropriate for this dataset.

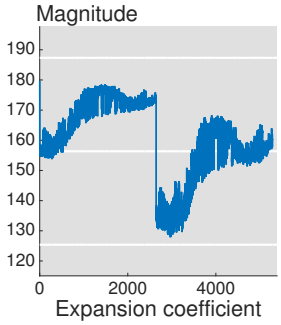
The results from two real-world datasets suggest that we should consider using pairwise Lipschitz smooth, total Lipschitz smooth and local normalized neighboring smooth criteria to model real-world smooth graph signals. In terms of the representation dictionary, among $\mathrm{V_L}$, $\mathrm{D_{poly(2)}}$ and $\mathrm{V_P}$, $\mathrm{D_{poly(2)}}$ is redundant; $\mathrm{V_P}$ is not orthonormal. We thus prefer using $\mathrm{V_L}$ because it is an orthonormal basis.

3.5 Conclusions

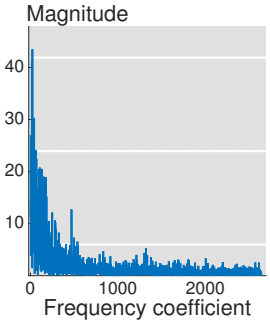
For smooth graph signals, we introduce four smoothness criterion and construct corresponding representations. We propose a generalized uncertainty principle on graphs and show the localization phenomenon of graph Fourier bases. We finally compare the empirical approximation performance of the proposed graph dictionary in two real datasets.



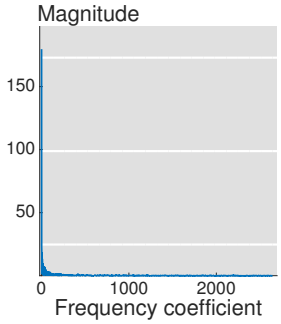
(a) Wind speed.



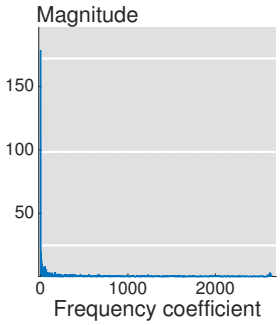
(b) Coefficients of $\mathrm{D}_{\mathrm{poly}(2)}$.



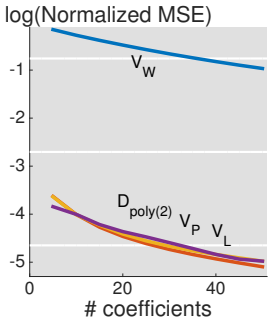
(c) Coefficients of V_{W}.



(d) Coefficients of V_{L}.

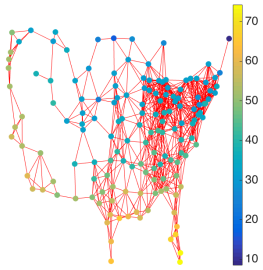


(e) Coefficients of V_{P}.

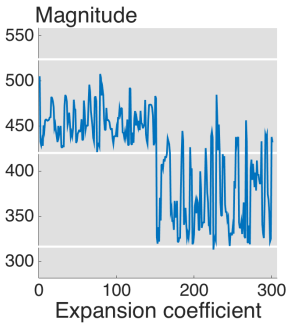


(f) Approximation error.

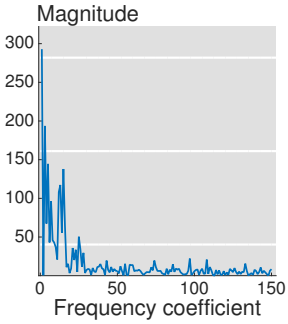
Figure 3.5: Approximation of wind speed. V_{L} and $\mathrm{D}_{\mathrm{poly}(2)}$ tie V_{P}; all of them are much better than V_{W}.



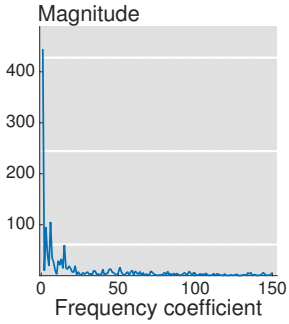
(a) Temperature.



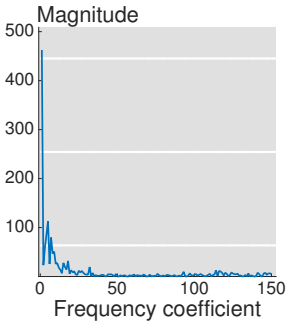
(b) Coefficients of $\mathrm{D}_{\mathrm{poly}(2)}$.



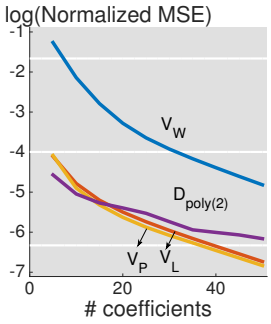
(c) Coefficients of V_{W}.



(d) Coefficients of V_{L}.



(e) Coefficients of V_{P}.



(f) Approximation err.

Figure 3.6: Approximation of temperature. V_{L} ties V_{P}; both are slightly better than $\mathrm{D}_{\mathrm{poly}(2)}$ and are much better than V_{W}.

Chapter 4

Representations of Piecewise-constant Graph Signals

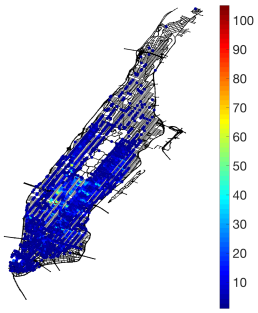
4.1 Introduction

In classical signal processing, a piecewise-constant signal means a signal that is locally constant over connected regions separated by lower-dimensional boundaries. This class of signals describes the phenomenon of abrupt changes in the mean level of a signal. It is often related to step functions, square waves and Haar wavelets and is widely used in signal processing [84]. In image processing, piecewise-constant signals are often used to model edges. A piecewise-constant interpolation provides a simple, yet effective approximation to any function: a finer piece leads to a better approximation. Piecewise-constant signals are the opposite of smooth signals. While smooth signals emphasize global trends and slow transitions, piecewise-constant graph signals emphasize localized behaviors and fast transitions.

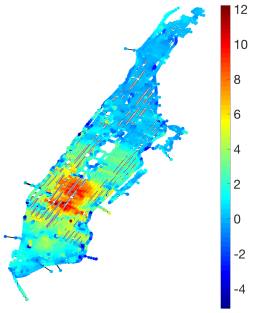
As a counterpart of piecewise-constant signal on graphs, a piecewise-constant graph signal is a graph signal that is locally constant over each connected subgraph [1]. In other words, a piecewise-constant graph signal can be expressed as a linear combination of several indicator functions that activate non-overlapping subgraphs. Piecewise-constant graph signals capture the large variations at the boundaries between pieces and ignore small variations within pieces, they allow us to find supports of localized patterns in the graph vertex domain. This is similar to what piecewise-constant time series achieve in classical signal processing. In the piecewise-constant graph signal, each piece indicates a localized pattern that exhibits homogeneous internal behavior and the number of pieces indicates the number of localized patterns. As an example, Figure 4.1 shows that a piecewise-constant graph signal can approximate well a real signal, taxi-pickup distribution in Manhattan.

Piecewise-constant graph signals have been used in many applications without having been explicitly defined; for example, in community detection, community labels form a piecewise-constant graph signal for a social network; in semi-supervised learning, classification labels form a piecewise-constant graph signal for a graph constructed from the dataset.

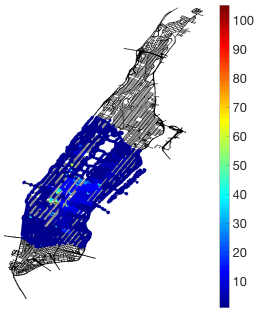
In this chapter, we define a piecewise-constant graph signal model and propose a graph dictionary that promotes sparsity in representations.



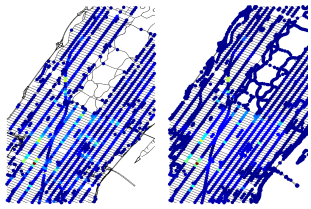
(a) Taxi-pickup distribution.



(b) Nonlinear approximation (best 50 graph frequencies).



(c) Piecewise-constant approximation (50 pieces).



(d) Zoom-in comparison real vs. PC approximation.

Figure 4.1: Piecewise-constant approximation well represents irregular, nonsmooth graph signals by capturing the large variations on the boundary of pieces and ignoring small variations inside pieces. Plot (a) shows Taxi-pickup distribution at 7 pm on Jan 1st, 2015 in Manhattan. Visually, the distribution is well approximated by a piecewise-constant (PC) graph signal with 50 pieces in Plot (c). On the other hand, the graph frequency based approximation in Plot (b) fails to capture localized variations.

4.2 Graph Signal Models

We introduce two definitions for piecewise-constant graph signals: one comes from the descriptive approach and the other one comes from the generative approach; we also show the connection between the two.

Recall that the graph difference operator (2.5) compares the signal coefficients of all the adjacent nodes and the output $\Delta\mathbf{x}$ is an *edge signal* representing the difference of $\mathbf{x}$. The ith element of $\Delta\mathbf{x}$,

$$(\Delta\mathbf{x})_i = \operatorname{sgn}(\mathrm{W}_{j,k})\sqrt{|\,\mathrm{W}_{j,k}\,|}\,(\mathbf{x}_k - \mathbf{x}_j)\,,$$

assigns the difference between two adjacent signal coefficients to the ith edge, where the ith edge connects the jth node to the kth node ($j < k$). For example, when $\mathbf{x} = \mathbf{e}_i$, $\|\Delta\mathbf{x}\|_0$ is the total out degree of the ith node; when $\mathbf{x} = \mathbf{1}$, $\|\Delta\mathbf{x}\|_0 = 0$. The term $\|\Delta\mathbf{x}\|_p^p$ also measures

the smoothness of $\mathbf{x}$. When G is an undirected graph, $\|\Delta\mathbf{x}\|_2^2 = \mathbf{x}^T \mathrm{L}\,\mathbf{x}$, where L is the graph Laplacian matrix, which measures the total Lipschiz smoothness as shown in Definition 2. When the graph is unweighted and all the edge weights are nonnegative, the elements of the ith row of Δ are simply

$$\Delta_{i,\ell} = \begin{cases} 1, & \ell = k; \\ -1, & \ell = j; \\ 0, & \text{otherwise.} \end{cases}$$

The class of piecewise-constant graph signals is a complement of the class of smooth graph signals, because many real-world graph signals contain localized information, which are hardly captured by smooth graph signals. Smooth graph signals emphasize the slow transitions over nodes; and piecewise-constant graph signals emphasize fast transitions on the graph vertex domain.

We now define piecewise-constant graph signals by using local sets, which have been used previously in graph cuts and graph signal reconstruction [22, 66].

Definition 9. Let $\{S_c\}_{c=1}^C$ be the partition of the node set $\mathcal{V}$. We call $\{S_c\}_{c=1}^C$ *local sets* when they satisfy that the subgraph corresponding to each local set is connected, that is, when G_{S_c} is connected for all c.

We can represent a local set S by using a local-set-based graph signal, $\mathbf{1}_S \in \mathbb{R}^N$, where

$$(\mathbf{1}_S)_i = \begin{cases} 1, & v_i \in S; \\ 0, & \text{otherwise.} \end{cases}$$

For a local-set-based graph signal, $\|\Delta\mathbf{1}_S\|_0$ measures how hard it is to cut the boundary edges between S and $\mathcal{V}\backslash S$ to make G_S an isolated subgraph.

Definition 10. A graph signal $\mathbf{x}$ is local-set-based piecewise-constant on $\{S_c\}_{c=1}^C$ when

$$\mathbf{x} \;=\; \sum_{c=1}^{C} a_c \mathbf{1}_{S_c},$$

where $\{S_c\}_{c=1}^C$ forms a valid series of local sets. Denote this class by $\mathrm{PC}(C)$.

When the value of the graph signal on each local set is different, $\|\Delta\mathbf{x}\|_0$ counts the total number of edges connecting nodes between local sets. Definition 10 only restricts the number of pieces C. We can put more restrictions on the shape of each local set. For example, we want a piecewise-constant graph signal to satisfy $\|\Delta\mathbf{x}\|_0 \leq K$, where K is some constant. When $(\Delta\mathbf{x})_i \neq 0$, the ith edge connects two nodes with different values. This requires that local sets form sub-graphs that do not have many in-between connections. We call this *small-cut piecewise-constant graph signals*

4.3 Graph Dictionary Construction

We now discuss representations for piecewise-constant graph signals based on a designed multiresolution local sets. The corresponding representation dictionary has a reasonable size and provides sparse representations for arbitrary piecewise-constant graph signals.

We aim to construct a series of local sets in a multiresolution fashion. We first define the multiresolution analysis on graphs.

Definition 11. A general multiresolution analysis on graphs consists of a sequence of embedded closed subspaces $V_0 \subset V_1 \subset V_2 \cdots \subset V_K$, such that

- upward completeness $\bigcup_{i=0}^{K} V_i = \mathbb{R}^N$;
- downward completeness $\bigcap_{i=0}^{K} V_i = \{c\mathbf{1}_{\mathcal{V}}, c \in \mathbb{R}\}$;
- there exists an orthonormal basis $\{\Phi\}_i$ for V_K.

Compared with the original multiresolution analysis [100], the complete space here is $\mathbb{R}^N$ instead of $\mathcal{L}_2(\mathbb{R})$ because of the discrete nature of a graph; we remove scale invariance and translation invariance because the rigorous definitions of scaling and translation for graphs are still unclear. This is the reason we call it *general multiresolution analysis on graphs*.

General Construction. The intuition behind the proposed construction is to build the connection between the subspaces and local sets: a bigger subspace corresponds to a finer resolution in the graph vertex domain, or more localized local sets. We initialize $S_{0,1} = \mathcal{V}$ to correspond to the 0th level subspace V_0, that is, $V_0 = \{c_0\mathbf{1}_{S_{0,1}}, c_0 \in \mathbb{R}\}$. We then partition $S_{0,1}$ into two disjoint local sets $S_{1,1}$ and $S_{1,2}$, corresponding to the first level subspace V_1, where $V_1 = \{c_1\mathbf{1}_{S_{1,1}} + c_2\mathbf{1}_{S_{1,2}}, c_1, c_2 \in \mathbb{R}\}$. We then recursively partition each larger local set into two smaller local sets. For the ith level subspace, we have $V_i = \sum_{j=1}^{2^i} c_j\mathbf{1}_{S_{i,j}}$ and then, we partition $S_{i,j}$ into $S_{i+1,2j-1}, S_{i+1,2j}$ for all $j = 1, 2, \ldots, 2^i$. We call $S_{i,j}$ the parent set of $S_{i+1,2j-1}, S_{i+1,2j}$ and $S_{i+1,2j-1}, S_{i+1,2j}$ are the children sets of $S_{i,j}$. When $|S_{i,j}| \leq 1$, $S_{i+1,2j-1} = S_{i,j}$ and $S_{i+1,2j} = \emptyset$. At the finest resolution, each local set corresponds to an individual node or an empty set. In other words, we build a binary decomposition tree that partitions a graph structure into multiple local sets. The ith level of the decomposition tree corresponds to the ith level subspace. The depth of the decomposition T depends on how local sets are partitioned; T ranges from N to $\lceil \log N \rceil$, where N corresponds to partitioning one node at a time and $\lceil \log N \rceil$ corresponds to an even partition at each level.

It is clear that the proposed construction of local sets satisfies three requirements in Definition 11. The initial subspace V_0 has the coast resolution. Through partitioning, local sets zoom into increasingly finer resolutions in the graph vertex domain. The subspace V_T with finest resolution zoom into each individual node and covers the entire $\mathbb{R}^N$. Classical scale invariance requires that when $f(t) \in V_0$, then $f(2^m t) \in V_m$, which is ill-posed in the graph domain because graphs are finite and discrete; the classical translation invariance requires that when $f(t) \in V_0$, then $f(t-n) \in V_0$, which is again ill-posed, this time because graphs are irregular. The essence of scaling and translation invariance, however, is to use the same function and its scales and translates to span different subspaces, which is what the proposed construction promotes. The scaling function is $\mathbf{1}_S$; the hierarchy of partition is similar to the scaling and translation, that is, when $\mathbf{1}_{S_{i,j}} \in \mathcal{V}_i$, then $\mathbf{1}_{S_{i+1,2j-1}}, \mathbf{1}_{S_{i+1,2j}} \in \mathcal{V}_{i+1}$, and when $\mathbf{1}_{S_{i+1,2j-1}} \in \mathcal{V}_{i+1}$ then $\mathbf{1}_{S_{i+1,2j}} \in \mathcal{V}_{i+1}$.

To summarize the construction, we build a local set decomposition tree by recursively partitioning a local set into two disjoint local sets until that all the local sets are individual nodes. We now show a toy example in Figure 4.2. In Partition 1, we partition the entire node set $S_{0,1} = \mathcal{V} = \{1, 2, 3, 4\}$ into two disjoint local sets $S_{1,1} = \{1, 2\}, S_{1,2} = \{3, 4\}$. Thus, $V_1 = \{c_1\mathbf{1}_{S_{1,1}} + c_2\mathbf{1}_{S_{1,2}}, c_1, c_2 \in \mathbb{R}\}$. Similarly, in Partition 2, we partition $S_{1,1}$ into two disjoint connected sets $S_{2,1} = \{1\}, S_{2,2} = \{2\}$; in Partition 3, we partition $S_{1,2}$ into $S_{2,3} = \{3\}, S_{2,4} = \{4\}$.

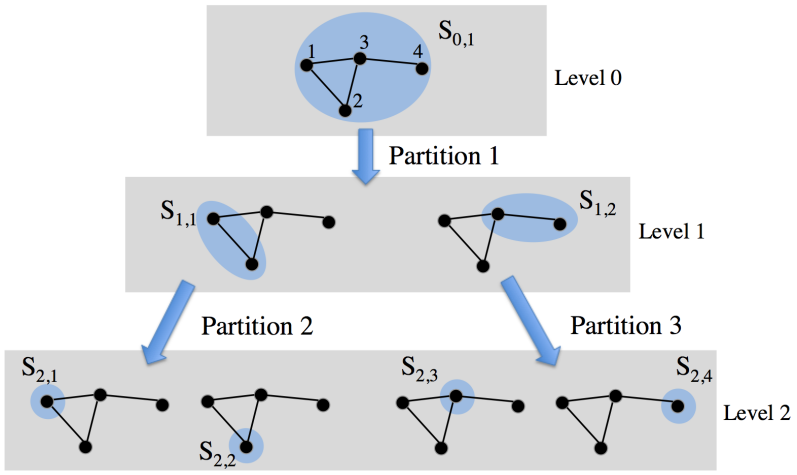


Figure 4.2: Local set decomposition. In each partition, we decompose a node set into two disjoint connected sets and generate a basis vector to the wavelet basis. $S_{0,1}$ is at Level 0, $S_{1,1}, S_{1,2}$ are at Level 1, and $S_{2,1}, S_{2,2}, S_{2,3}, S_{2,4}$ are at Level 2.

Thus, $V_2 = \{c_1\mathbf{1}_{S_{2,1}} + c_2\mathbf{1}_{S_{2,2}} + c_3\mathbf{1}_{S_{2,3}} + c_4\mathbf{1}_{S_{2,4}}, c_1, c_2, c_3, c_4 \in \mathbb{R}\} = \mathbb{R}^4$.

Graph Partition Algorithm. The graph partition is the key step to constructing local sets. In Corollary 1, we will show that any even partition can minimize the worst case when representing piecewise-constant graph signals. This suggests that when representing small-cut piecewise-constant graph signals, the sizes of the local sets matter and the shapes do not matter. Here we introduce three heuristic algorithms to implement graph partition.

The first one is based on spectral clustering [22]. We first obtain the graph Laplacian matrix of a local set and compute the eigenvector corresponding to the second smallest eigenvalue of the graph Laplacian matrix. We then set the median number of the eigenvector as the threshold; we put the nodes whose corresponding values in the eigenvector are no smaller than the threshold into one child local set and put the nodes whose corresponding values in the eigenvector are smaller than the threshold into the other child local set. This method guarantees that two child local sets have the same number of nodes, but does not guarantee that each child local set is connected.

The second one is based on spanning tree. To partition a local set, we first obtain the maximum spanning tree of the subgraph and then find a balance node in the spanning tree. The balance node partition the spanning tree into two subtrees with the closet number of nodes [78]. We remove the balance node from the spanning tree, the resulting largest connected component form one child local set and the other nodes including the balance node forms the other child local set. This method guarantees that two child local sets are connected, but does not guarantees that they have the same number of nodes. When the original subgraph is highly connected, the spanning tree loses some connection information and the shape of the local set may not capture the community in the subgraph.

The third one is based on the 2-means clustering. We first randomly select two nodes as the community center and assign every other node to its nearest community center based on the geodesic distance. We then recompute the community center for each community by minimizing

the summation of the geodesic distances to all the other nodes in the community and assign node to its nearest community center again. We keep doing this until the community centers converge after a few iterations. This method is inspired from the classical k-means clustering; it also guarantees that two child local sets are connected, but does not guarantees that they have the same number of nodes.

The proposed construction of local sets does not restrict to any particular graph partition algorithm; depending on the applications, the partition step can also be implemented by many other existing graph partition algorithms.

Dictionary Representations. We collect local sets by level in ascending order in a dictionary, with atoms corresponding to each local set, that is, $\mathrm{D}_{\mathrm{LSPC}} = \{\mathbf{1}_{S_{i,j}}\}_{i=0,j=1}^{i=T,j=2^i}$. We call it the *local-set-based piecewise-constant dictionary*. After removing empty sets, the dictionary has $2N-1$ atoms, that is, $\mathrm{D}_{\mathrm{LSPC}} \in \mathbb{R}^{N\times(2N-1)}$; each atom is a piecewise-constant graph signal with various sizes and localizing various parts of a graph. Since the size of the proposed dictionary is linear with the number of nodes, this method is easy to scale.

Wavelet Basis. We construct a wavelet basis based on the local-set-based piecewise-constant dictionary. We combine two local sets partitioned from the same parent local set to form a basis vector. Let the local sets $S_{i+1,2j-1}, S_{i+1,2j}$ have the same parent local set $S_{i,j}$, the basis vector combing these two local sets is

$$\sqrt{\frac{|S_{i+1,2j-1}||S_{i+1,2j}|}{|S_{i+1,2j-1}|+|S_{i+1,2j}|}}\left(\frac{1}{|S_{i+1,2j-1}|}\mathbf{1}_{S_{i+1,2j-1}} - \frac{1}{|S_{i+1,2j}|}\mathbf{1}_{S_{i+1,2j}}\right).$$

To represent in a matrix form, the wavelet basis is

$$\mathrm{W}_{\mathrm{LSPC}} = \mathrm{D}_{\mathrm{LSPC}}\,\mathrm{D}_2,$$

where the downsampling matrix

$$\begin{aligned}\mathrm{D}_2 &= \begin{bmatrix} \frac{1}{\|\mathbf{d}_1\|_2} & 0 & \cdots & 0 \\ 0 & g(\mathbf{d}_2,\mathbf{d}_3) & \cdots & 0 \\ 0 & -g(\mathbf{d}_3,\mathbf{d}_2) & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(\mathbf{d}_{2N-2},\mathbf{d}_{2N-1}) \\ 0 & 0 & \cdots & -g(\mathbf{d}_{2N-1},\mathbf{d}_{2N-2}) \end{bmatrix} \\ &\in \mathbb{R}^{(2N-1)\times N},\end{aligned}$$

with $\mathbf{d}_i$ is the ith column of $\mathrm{D}_{\mathrm{LSPC}}$, and

$$g(\mathbf{d}_i,\mathbf{d}_j) = \sqrt{\frac{\|\mathbf{d}_j\|_0}{(\|\mathbf{d}_i\|_0+\|\mathbf{d}_j\|_0)\,\|\mathbf{d}_i\|_0}}.$$

The downsampling matrix U_2 combines two consecutive column vectors in $\mathrm{D}_{\mathrm{LSPC}}$ to form one column vector in $\mathrm{W}_{\mathrm{LSPC}}$ and the function $g(\cdot,\cdot)$ reweighs the column vectors in $\mathrm{D}_{\mathrm{LSPC}}$ to ensure that each column vector in $\mathrm{W}_{\mathrm{LSPC}}$ has norm 1 and sums to 0.

Another explanation is that when we recursively partition a node set into two local sets, each partition generates a wavelet basis vector. We still use Figure 4.2 as an example. In Partition 1, we partition the entire node set $S_{0,1} = \{1,2,3,4\}$ into $S_{1,1} = \{1,2\}, S_{1,2} = \{3,4\}$ and generate a basis vector

$$\begin{aligned} & \sqrt{\frac{|S_{1,1}||S_{1,2}|}{|S_{1,1}|+|S_{1,2}|}}\left(\frac{1}{|S_{1,1}|}\mathbf{1}_{S_{1,1}} - \frac{1}{|S_{1,2}|}\mathbf{1}_{S_{1,2}}\right) \\ = \; & \frac{1}{2}\begin{bmatrix}1 & 1 & -1 & -1\end{bmatrix}; \end{aligned}$$

in Partition 2, we partition $S_{1,1}$ into two disjoint connected sets $S_{2,1} = \{1\}, S_{2,2} = \{2\}$ and generate a basis vector

$$\begin{aligned} & \sqrt{\frac{|S_{2,1}||S_{2,2}|}{|S_{2,1}|+|S_{2,2}|}}\left(\frac{1}{|S_{2,1}|}\mathbf{1}_{S_{2,1}} - \frac{1}{|S_{2,2}|}\mathbf{1}_{S_{2,2}}\right) \\ = \; & \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1 & 0 & 0\end{bmatrix}; \end{aligned}$$

in Partition 3, we partition $S_{1,2}$ into $S_{2,3} = \{3\}, S_{2,4} = \{4\}$ and generate a basis vector

$$\begin{aligned} & \sqrt{\frac{|S_{2,3}||S_{2,4}|}{|S_{2,3}|+|S_{2,4}|}}\left(\frac{1}{|S_{2,3}|}\mathbf{1}_{S_{2,3}} - \frac{1}{|S_{2,4}|}\mathbf{1}_{S_{2,4}}\right) \\ = \; & \frac{1}{\sqrt{2}}\begin{bmatrix}0 & 0 & 1 & -1\end{bmatrix}. \end{aligned}$$

We summarize the construction of the *local-set-based wavelet basis* in Algorithm 1.

We now analyze some properties of the proposed construction of the local sets and wavelet basis.

Theorem 6. The proposed construction of local sets satisfies the general multiresolution analysis on graphs.

We have shown this in the previous section. We list here for completeness. We next show that the local-set-based wavelet basis is a valid orthonormal basis.

Theorem 7. The local-set-based wavelet basis constructs an orthonormal basis.

We show that the local-set-based wavelet basis is a good representation for piecewise-constant graph signals through promoting the sparsity.

Theorem 8. Let W be the output of Algorithm 1 and T be the maximum level of the decomposition in Algorithm 1 . For all $\mathbf{x} \in \mathbb{R}^N$, we have

$$\left\|\mathrm{W}_{\mathrm{LSPC}}^T \mathbf{x}\right\|_0 \leq 1 + \|\Delta \mathbf{x}\|_0 \, T.$$

Algorithm 1 Local-set-based Wavelet Basis Construction

Input $G(\mathcal{V}, \mathcal{E}, \mathrm{A})$ graph
Output $\mathrm{W}_{\mathrm{LSPC}}$ wavelet basis

Function

initialize a stack of node sets $\mathbb{S}$ and a set of vectors W
push $S = \mathcal{V}$ into $\mathbb{S}$
add $\mathbf{w} = \frac{1}{\sqrt{|S|}}\mathbf{1}_S$ into $\mathrm{W}_{\mathrm{LSPC}}$
while the cardinality of the largest element of $\mathbb{S}$ is bigger than 1
 pop up one element from $\mathbb{S}$ as S
 partition S into two disjoint connected sets S_1, S_2
 push S_1, S_2 into $\mathbb{S}$
 add $\mathbf{w} = \sqrt{\frac{|S_1||S_2|}{|S_1|+|S_2|}}\left(\frac{1}{|S_1|}\mathbf{1}_{S_1} - \frac{1}{|S_2|}\mathbf{1}_{S_2}\right)$ into $\mathrm{W}_{\mathrm{LSPC}}$
end
return $\mathrm{W}_{\mathrm{LSPC}}$

The maximum level of the decomposition is determined by the choice of graph partition algorithm. Theorem 8 shows that what it matters is the cardinality of each local set, instead of the shape. To achieve the best sparse representation, we should partition each local set as evenly as possible. Note that when the partition is perfectly even, the resulting wavelet basis is the same with the classical Haar wavelet basis.

Corollary 1. Let the local-set-based wavelet basis evenly partition the node set each time. We have

$$\left\|\mathrm{W}_{\mathrm{LSPC}}^T\,\mathbf{x}\right\|_0 \leq 1 + \|\Delta\mathbf{x}\|_0 \lceil \log N \rceil.$$

We see that the local-set-based piecewise-constant wavelet basis provides a sparse representation for the piecewise-constant graph signals. Note that the graph difference operator provides more sparse representation than the local-set-based wavelet basis, however, the graph difference operator is not necessarily a one-to-one mapping and is bad at reconstruction. This is because the graph difference operator only focuses on the pairwise relationship. On the other hand, the local-set-based wavelet basis is good at reconstruction and provides multiresolution view in the graph vertex domain.

The even partition minimizes the worst case; it does not necessarily mean that the even partition is good for all the applications. For example, a graph has two communities, a huge one and a tiny one, which implies that a piecewise-constant graph signal sits on a part of either of two communities. In this case, we cut a few edges to partition two communities and assign a local set for each of them, instead of partitioning the huge community to make sure that two local sets have same cardinality.

4.4 Experimental Results

Approximation is a standard task to evaluate a representation and it is similar to compression. The goal is to use a few expansion coefficients to approximate a graph signal. We compare

the graph Fourier transform [2], the windowed graph Fourier transform [79], the local-set-based wavelet basis and dictionary. The graph Fourier transform is the eigenvector matrix of the graph shift and the windowed graph Fourier transform provides vertex-frequency analysis on graphs. For the local-set-based piecewise-constant wavelet basis and dictionary, we also consider three graph partition algorithms, including spectral clustering, spanning tree and 2-means.

Algorithm. Since the graph Fourier transform and the local-set-based wavelet bases are orthonormal bases, we consider nonlinear approximation for the graph Fourier bases, that is, after expanding in with a representation, we should choose the K largest-magnitude expansion coefficients so as to minimize the approximation error. Let $\{\phi_k \in \mathbb{R}^N\}_{k=1}^N$ and $\{\widehat{\phi}_k \in \mathbb{R}^N\}_{k=1}^N$ be a pair of biorthonormal basis and $\mathbf{x} \in \mathbb{R}^N$ be a signal. Here the graph Fourier transform matrix $\mathrm{U} = \{\widehat{\phi}_k\}_{k=1}^N$ and the graph Fourier basis $\mathrm{V} = \{\phi_k\}_{k=1}^N$. The nonlinear approximation to $\mathbf{x}$ is

$$\mathbf{x}^* = \sum_{k \in \mathcal{I}_K} \left\langle \mathbf{x}, \widehat{\phi}_k \right\rangle \phi_k, \tag{4.1}$$

where $\mathcal{I}_K$ is the index set of the K largest-magnitude expansion coefficients. When a basis promotes sparsity for $\mathbf{x}$, only a few expansion coefficients are needed to obtain a small approximation error. Note that (4.1) is a special case of (2.15) when the distance metric $d(\cdot, \cdot)$ is the ℓ_2 norm and D is an orthonormal basis.

Since the the windowed graph dictionary and the local-set-based piecewise-constant dictionaries are redundant, we solve the following sparse coding problem,

$$\begin{aligned} \mathbf{x}' = \quad & \arg\min_{\mathbf{a}} \quad \|\mathbf{x} - \mathrm{D}\,\mathbf{a}\|_2^2, \\ & \text{subject to :} \quad \|\mathbf{a}\|_0 \leq K, \end{aligned} \tag{4.2}$$

where D is a redundant dictionary and $\mathbf{a}$ is a sparse code. The idea is to use a linear combination of a few atoms from D to approximate the original signal. When D is an orthonormal basis, the closed-form solution is exactly (4.1). We solve (2.15) by using the orthogonal matching pursuit, which is a greedy algorithm [96]. Note that (4.2) is a special case of (2.15) when the distance metric $d(\cdot, \cdot)$ is the ℓ_2 norm.

Experiments. We test the four representations on two datasets, including the Minnesota road graph [97] and the U.S city graph [98].

For the Minnesota road graph, we simulate a piecewise-constant graph signal by randomly picking 5 nodes as community centers and assigning each other node to its nearest community center based on the geodesic distance. We assign a random integer to each community. The simulated graph signal is shown in Figure 4.3. The signal contains 5 piecewise constants and 84 inconsistent edges. The frequency coefficients and the wavelet coefficients obtained by using three graph partition algorithms are shown in Figure 4.3(b), (c), (d) and (e). The sparsities of the wavelet coefficients for spectral clustering, spanning tree, and 2-means are 364, 254, and 251, respectively; the proposed wavelet bases provide much better sparse representations than the graph Fourier transform.

The evaluation metric of the approximation error is the normalized mean square error, that is,

$$\text{Normalized MSE} = \frac{\|\mathbf{x}' - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2},$$

where $\mathbf{x}'$ is the approximation signal and $\mathbf{x}$ is the original signal. Figure 4.3(f) shows the approximation errors given by the four representations. The x-axis is the number of coefficients used in approximation, which is K in (4.1) and (2.15) and the y-axis is the approximation error, where lower means better. We see that the local-set-based wavelet with spectral clustering and local-set-based dictionary with spectral clustering provides much better performances and the windowed graph Fourier transform catches up with graph Fourier transform around 15 expansion coefficients. Figure 4.3(g) and (h) compares the local-set-based wavelets and dictionaries with three different partition algorithms, respectively. We see that the spanning tree and 2-means have similar performances, which are better than spectral clustering. This is consistent with the sparsities of the wavelet coefficients, where the wavelet coefficients of spanning tree and 2-means are more sparse than those of spectral clustering.

The U.S city graph is a network representation of 150 weather stations across the U.S. We assign an edge when two weather stations are within 500 miles. The graph includes 150 nodes and 1033 undirected, unweighted edges. Based on the geographical area, we partition the nodes into four communities, including the north area (N), the middle area (M), the south area (S), and the west area (W). The corresponding piecewise-constant graph signal is

$$\mathbf{x} = \mathbf{1}_N + 2 \cdot \mathbf{1}_M + 3 \cdot \mathbf{1}_S + 4 \cdot \mathbf{1}_W. \tag{4.3}$$

The graph signal is shown in Figure 4.4(a), where dark blue indicates the north area, the light indicates the middle area, the dark yellow indicates the south area and the light yellow indicates the west area. The signal contains 4 piecewise constants and 144 inconsistent edges.

The frequency coefficients and the wavelet coefficients obtained by using three graph partition algorithms are shown in Figure 4.4(b), (c), (d) and (e). The sparsities of the wavelet coefficients for spectral clustering, spanning tree, and 2-means are 45, 56, and 41, respectively; the proposed wavelet bases provide much better sparse representations than the graph Fourier transform.

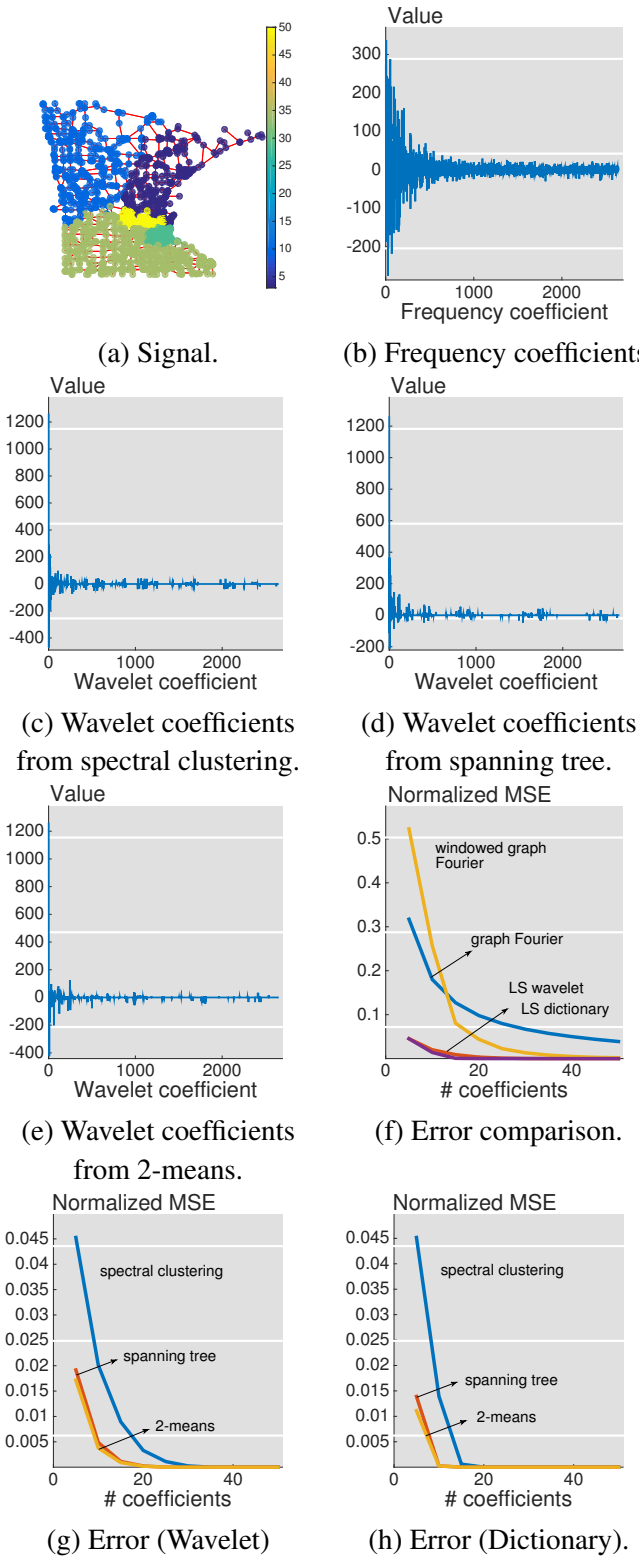
The evaluation metric of the approximation error is also the normalized mean square error. Figure 4.4(f) shows the approximation errors given by the four representations. Similarly to Figure 4.4(d), the local-set-based wavelet with spectral clustering and local-set-based dictionary with spectral clustering provides much better performances and the windowed graph Fourier transform catches up with graph Fourier transform around 25 expansion coefficients. Figure 4.4(g) and (h) compares the local-set-based wavelets and dictionaries with three different graph partition algorithms, respectively. We see that the spectral clustering provides the best performance.

To summarize the task of approximation, the proposed local set based representations provide a reliable approximation to a piecewise-constant graph signal because of the sparsity promotion.

4.5 Conclusions

For piecewise-constant graph signals, we define the piecewise-constant graph signals and construct the multiresolution local sets, a local-set-based piecewise-constant dictionary and local-set-based piecewise-constant wavelet basis, which are provably useful to provide a multiresolution analysis on graphs and promote sparsity for piecewise-constant graph signals. We finally

compare the empirical approximation performance of the proposed graph dictionary in two simulated datasets.



(a) Signal.

(b) Frequency coefficients.

(c) Wavelet coefficients from spectral clustering.

(d) Wavelet coefficients from spanning tree.

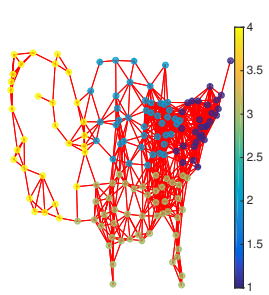
(e) Wavelet coefficients from 2-means.

(f) Error comparison.

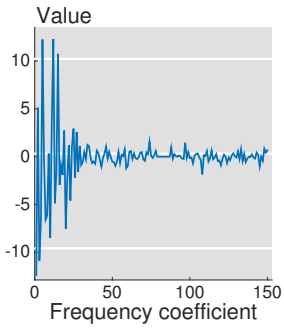
(g) Error (Wavelet)

(h) Error (Dictionary).

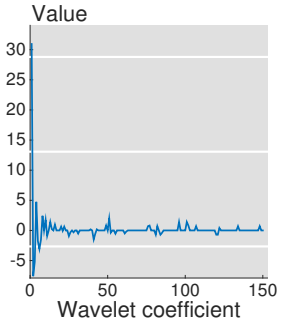
Figure 4.3: Approximation on the Minnesota road graph.



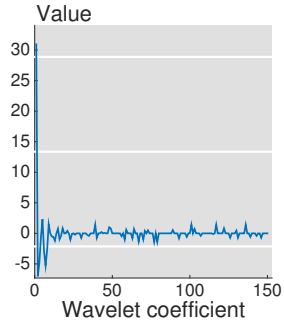
(a) Signal.



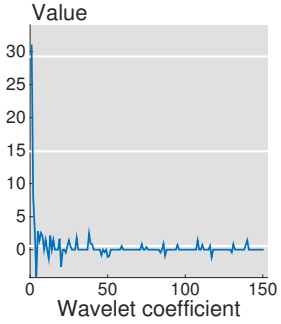
(b) Frequency coefficients.



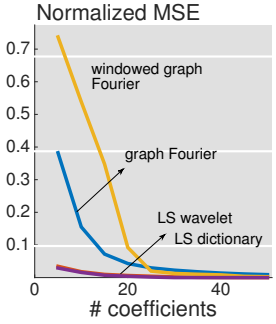
(c) Wavelet coefficients from spectral clustering.



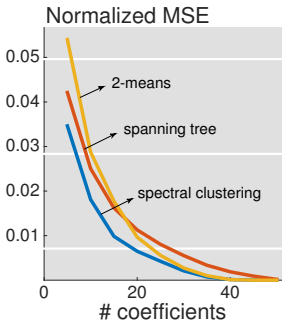
(d) Wavelet coefficients from spanning tree.



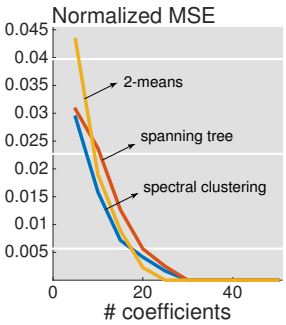
(e) Wavelet coefficients from 2-means.



(f) Error comparison.



(g) Error (wavelet).



(h) Error (dictionary).

Figure 4.4: Approximation on the U.S city graph.

Chapter 5

Representations of Piecewise-smooth Graph Signals

5.1 Introduction

A signal is piecewise-smooth when it can be broken into distinct pieces and on each piece the signal is smooth [101]. Conceptually, a piecewise-smooth signal is a combination of smooth signal and piecewise-constant signal. Images are often modeled as piecewise-smooth functions because objects are well captured within pieces and edges are well captured by the boundaries between pieces [102, 103].

To be able to deal with as wide a class of real-world graphs signals as possible, we combine smooth and piecewise-constant graph signals into piecewise-smooth graph signals. In many previous works, graph filter banks are proposed to analyze piecewise-smooth graph signals [47, 50, 51]; however, the concept of piecewise-smooth graph signals does not explicitly defined.

In this chapter, we define a piecewise-smooth graph signal model and propose a graph dictionary that promotes sparsity in representations.

5.2 Graph Signal Models

Based on smooth graph signal models, we have two types of piecewise-smooth signal models, including the piecewise-polynomial class and the piecewise-bandlimited class.

Definition 12. A graph signal $\mathbf{x}$ is piecewise-polynomial with C pieces and degree K when

$$\mathbf{x} \;=\; \sum_{c=1}^{C} \mathbf{x}^{(c)} \mathbf{1}_{S_c},$$

where $\mathbf{x}^{(c)}$ is a kth order polynomial signal on the subgraph G_{S_c} with $x_i^{(c)} \;=\; a_c + \sum_{j \in S_c} \sum_{k=1}^{K} a_{k,j,c} d^k(v_i, v_j)$. Denote this class by $\mathrm{PPL}(C, K)$.

$\mathrm{PPL}(1, K)$ is the polynomial class with degree K, $\mathrm{PL}(K)$ from Definition 5, and $\mathrm{PPL}(C, 0)$ is the piecewise-constant class with C pieces, $\mathrm{PC}(C)$ from Definition 10. The degrees of freedom for a local set S_c at the polynomial degree k is the number of origins, that is, $\left\| \begin{bmatrix} a_{k,1,c} & a_{k,2,c} & \dots & a_{k,|S_c|,c} \end{bmatrix} \right\|_0$.

Definition 13. A graph signal $\mathbf{x}$ is piecewise-bandlimited with C pieces and bandwidth K when

$$\mathbf{x} \;=\; \sum_{c=1}^{C} \mathbf{x}^{(c)} \mathbf{1}_{S_c},$$

where $\mathbf{x}^{(c)}$ is a bandlimited signal on the subgraph G_{S_c} with $x_i^{(c)} = \sum_{k=0}^{K} a_{k,c}\, \mathrm{V}_{i,k}^{(c)}$, and $\mathrm{V}^{(c)}$ is a graph Fourier basis of G_{S_c}. Denote this class by $\mathrm{PBL}_{\mathrm{V}}(C, K)$.
We use zero padding to ensure $\mathrm{V}^{(c)} \in \mathbb{R}^{N \times N}$ for each G_{S_c}. Still, $\mathrm{V}^{(c)}$ can be the eigenvector matrix of the adjacency matrix, graph Laplacian matrix or the transition matrix.

5.3 Graph Dictionary Construction

The representations of piecewise-smooth graph signals is based on the local-set piecewise-constant dictionary. To represent piecewise-smooth graph signals, we use multiple atoms for each local set. We take the piecewise-polynomial signals as an example. For each local set,

$$\mathrm{D}_{S_{i,j}} \;=\; \left[\mathbf{1} \quad \mathrm{D}_{S_{i,j}}^{(1)} \quad \mathrm{D}_{S_{i,j}}^{(2)} \quad \dots \quad \mathrm{D}_{S_{i,j}}^{(K)}\right],$$

where $(\mathrm{D}_{S_{i,j}}^{(k)})_{m,n} = d^k(v_m, v_n)$, when $v_m, v_n \in S_{i,j}$; and 0, otherwise. The number of atoms in $\mathrm{D}_{S_{i,j}}^{(k)}$ is $1 + K|S_{i,j}|$. We collect the sub-dictionaries for all the multiresolution local sets to form the *local-set-based piecewise-polynomial dictionary*, that is, $\mathrm{D}_{\mathrm{LSPP}} \;=\; \{\mathrm{D}_{S_{i,j}}\}_{i=0,j=1}^{T,2^i}$. The number of atoms in $\mathrm{D}_{\mathrm{LSPS}}$ is $O(KNT)$, where K is the maximum degree of polynomial, N is the size of the graph and T is the maximum level of the decomposition. When we use even partitioning, the total number of atoms is $O(KN \log N)$.

Similarly, to model piecewise-bandlimited signals, we replace $\mathrm{D}_{S_{i,j}}$ by the graph Fourier basis of each subgraph $G_{S_{i,j}}$ to form the *local-set-based piecewise-bandlimited dictionary*. The total number of atoms of the corresponding $\mathrm{D}_{\mathrm{LSPB}}$ is then $O(NT)$. Both local-set-based piecewise-polynomial dictionary and local-set-based piecewise-bandlimited dictionary are called local-set-based piecewise-smooth dictionary, denoted as $\mathrm{D}_{\mathrm{LSPS}}$.

Recall that for piecewise-constant signals, we use the sparse coding to do exact approximation. To minimize the approximation error in the worst case, we want even partition. However, for piecewise-smooth signals, we cannot use the sparse coding to do exact approximation. To minimize the approximation error, both the sizes and the shapes of the local sets matter for piecewise-smooth graph signals.

Theorem 9. (**The local-set based piecewise-smooth dictionary promotes the sparsity for piecewise-smooth graph signals.**) For all $\mathbf{x} \in \mathrm{PBL}_{\mathrm{V}_L}(C, K)$, where V_L is the graph Fourier basis of the graph Laplacian matrix L, we have $\|\mathbf{a}^*\|_0 \leq 2KT \,\|\Delta \mathbf{x}_{\mathrm{PC}}\|_0$, where T is the maximum level of the decomposition, $\mathbf{x}_{\mathrm{PC}}$ is a piecewise-constant signal that corresponds the same local sets with $\mathbf{x}$ and

$$\begin{aligned} \mathbf{a}^* \;=\; & \arg\min_{\mathbf{a}} \|\mathbf{a}\|_0 \,, \\ & \text{subject to } \|\mathbf{x} - \mathrm{D}_{\mathrm{LSPS}}\, \mathbf{a}\|_2^2 \leq \epsilon_{\mathrm{par}} \,\|\mathbf{x}\|_2^2 \,, \end{aligned}$$

where ϵ_{par} is a constant determined by the graph partitioning algorithm,

$$\epsilon_{\text{par}} = \frac{(\lambda_K \mathbf{x}^T\mathbf{x} - \mathbf{x}^T \operatorname{L_{cut}} \mathbf{x})}{\min_{S_c} \lambda_{K+1}^{(S_c)} \|\mathbf{x}\|_2^2}$$

with $\lambda_{K+1}^{(S_c)}$ be the $K+1$th eigenvalue of the graph Laplacian matrix of G_{S_c} and

$$\mathbf{x}^T \operatorname{L_{cut}} \mathbf{x} \;=\; \sum_{i,j \in (\mathcal{E}/\cup_c \mathcal{E}_{S_c})} \operatorname{W}_{i,j}(x_i - x_j)^2.$$

A small ϵ_{par} requires that the local sets have strong internal connections and weak external connections. It is hard to design the graph partitioning algorithm by minimizing ϵ_{par}, but ϵ_{par} can be a quantitative metric to evaluate the graph partitioning algorithm.

We further use the local-set piecewise-smooth dictionary to detect piecewise-smooth signals from random noises.

Theorem 10. We consider statistically testing the null and alternative hypothesis,

$$\begin{aligned} H_0 \;&:\; \mathbf{y} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}), \\ H_1 \;&:\; \mathbf{y} \sim \mathcal{N}(\mathbf{x}, \sigma^2 \mathbf{I}),\ \mathbf{x} \in \operatorname{PBL}_{\operatorname{V}_L}(C, K). \end{aligned}$$

We solve the following optimization problem

$$\begin{aligned} \mathbf{a}_{\mathbf{y}}^* \;&=\; \arg\min_{\mathbf{a}} \|\mathbf{y} - \operatorname{D_{LSPB}} \mathbf{a}\|_2^2 \\ &\qquad \text{subject to } \|\mathbf{a}\|_0 \leq 2KT \|\Delta \mathbf{x}_{\text{PC}}\|_0, \end{aligned}$$

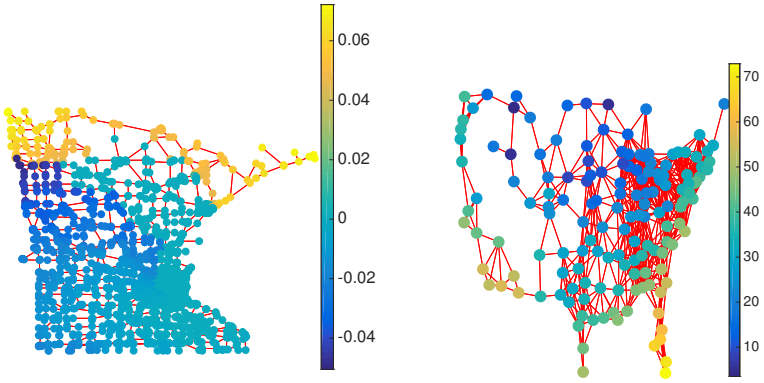
by using the matching pursuit algorithm. We reject the null if $\left\|\mathbf{a}_{\mathbf{y}}^*\right\|_\infty > \sigma\sqrt{2\log(KNT/\delta)}$. If

$$\frac{\|\mathbf{x}\|_2}{\sigma} \geq C\sqrt{2KT \|\Delta \mathbf{x}_{\text{PC}}\|_0}\sqrt{8\log(\frac{KNT}{\delta})},$$

where $C = \max_{|\Omega| \leq 2KT\|\Delta \mathbf{x}_{\text{PC}}\|_0} \left\|(\operatorname{D_{LSPB}})_\Omega\right\|_2 / (1 - \sqrt{\epsilon_{\text{par}}})$ with ϵ_{par} is a constant related to the graph partition algorithm. Then under H_0, $\mathbb{P}(\text{Reject}) \leq \delta$, and under H_1, $\mathbb{P}(\text{Reject}) \leq \delta$.

5.4 Experimental Results

Similarly to experiments for piecewise-constant graph signals in Chapter 4, we still test the representations on two datasets, the Minnesota road graph [97] and the U.S city graph [98]. On the Minnesota road graph, we simulate 100 piecewise-constant graph signals as follows: we random choose three nodes as cluster centers and assign all other nodes to their nearest cluster centers based on the geodesic distance. We assign a random integer to each cluster. We further obtain 100 piecewise-polynomial graph signals by element-wise multiplying a polynomial function, $-d^2(v_0, v) + 12d(v_0, v)$, where v_0 is a reference node that assigns randomly. As an example, see Figure 5.1(a).



(a) Minnesota. (b) U.S weather station.

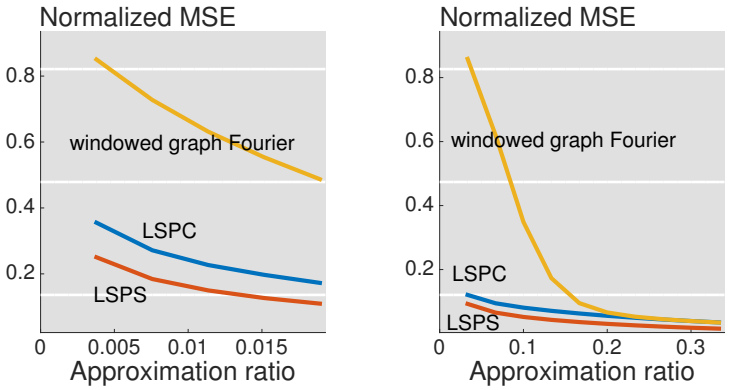
Figure 5.1: Graph signal.

On the U.S city graph, we use the real temperature measurements. The graph includes 150 weather stations and each weather station has 365 days of recordings (one recording per day), for a total of 365 graph signals. As an example, see Figure 5.1(b).

The approximation error is measured by the normalized mean square error. Figure 5.2 shows the averaged approximation errors. LSPC denotes local-set-based piecewise-constant dictionary and LSPS denotes local-set-based piecewise-smooth dictionary. For the windowed graph Fourier transform, we use 15 filters; for LSPS, three piecewise-smooth models provide tight performances; here we show the results of the piecewise-polynomial smooth model with degree $K = 2$. We see that the local-set-based dictionaries perform better than the windowed graph Fourier transform; local-set-based piecewise-smooth dictionary is slightly better than local-set-based piecewise-constant dictionary; even though the windowed graph Fourier transform is solid in theory, provides highly redundant representations and is useful for visualization, it does not well approximate complex graph signals.

5.5 Conclusions

For piecewise-smooth graph signals, we define the piecewise-smooth graph signals and construct a local-set-based piecewise-smooth dictionary, which promotes sparsity for piecewise-smooth graph signals. We then study how the proposed local-set-based piecewise-smooth dictionary works in approximation. We finally compare the empirical approximation performance of the proposed graph dictionary in two simulated datasets.



(a) Minnesota. (b) U.S weather station.

Figure 5.2: Approximation error. Approximation ratio is the percentage of used coefficients (s in (4.2)).

Part III

Sampling and Recovery of Graph Signals

Overview of Sampling and Recovery

In classical signal processing, sampling and recovery are key techniques to link continuous-time signals (functions of a real variable) and discrete-time signals (sequences indexed by integers) [84, 104]. Sampling produces a sequence from a function, and recovery produces a function from a sequence. The ability to sample a function, manipulate the resulting sequence with a discrete-time system, and then recover a function, is the foundation of digital signal processing. Conversely, the ability to recover a sequence to create a function, manipulate the resulting function with a continuous-time system, and then sample to produce a sequence, is the foundation of digital communications.

In this part, we consider sampling and recovery of graph signals. In general, graph signal sampling is a reduction of a graph signal to a small number of measurements, and graph signal recovery is a reconstruction of a graph signal from noisy, missing, or corrupted measurements. Many types of sampling are considered in literature. For example, subsampling selects one node in each measurement [62, 90, 105]; local sampling selects a set of nodes in each measurement [64]; and dynamic sampling considers time-evolving graph signals and selects one node at different time stamps in each measurement [63]. Here we focus on subsampling. The recovery techniques is usually related to the sampling procedure because the sampling procedure determines the property of the measurements.

The main application of sampling and recovery of graph signals is semi-supervised learning with graphs. Semi-supervised learning is a machine learning technique to train classifiers with both labeled and unlabeled data by assuming that unlabeled data can provide distribution information to build a stronger classifier [106]. Many algorithms for semi-supervised learning are based on graphs that are constructed from a given dataset [106], often by modeling each node as a data sample and connecting two nodes by an edge if the distance between their features is in a given range, which is similar to the construction of random geometric graphs. Based on the assumption that adjacent nodes have similar labels, semi-supervised learning diffuses label probabilities from labeled data to unlabeled data along the graph structure and classifies unlabeled data according to those label probabilities. In this case, graph signal sampling actively selects the most informative data samples as training data and query their labels; and graph signal recovery finds the labels of unlabeled data based on both graph structure and labeled data.

Mathematically, suppose that we want to sample M coefficients of a graph signal $\mathbf{x} \in \mathbb{R}^N$ to produce a sampled signal $\mathbf{x}_{\mathcal{M}} \in \mathbb{R}^M$ ($M < N$), where $\mathcal{M} = (\mathcal{M}_0, \cdots, \mathcal{M}_{M-1})$ denotes the sequence of *sampled* indices, and $\mathcal{M}_i \in \{0, 1, \cdots, N-1\}$. When samples are corrupted by noise, we consider a noisy sampled signal $\mathbf{y} = \mathbf{x}_{\mathcal{M}} + \epsilon \in \mathbb{R}^M$, where ϵ is noise. We then interpolate $\mathbf{y}$ to get $\mathbf{x}' \in \mathbb{R}^N$, which recovers $\mathbf{x}$ either exactly or approximately. The sampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$, defined as

$$\Psi_{i,j} = \begin{cases} 1, & j = \mathcal{M}_i; \\ 0, & \text{otherwise}, \end{cases} \tag{5.1}$$

and the interpolation operator Φ is a linear mapping from $\mathbb{R}^M$ to $\mathbb{R}^N$ (see Figure 5.3),

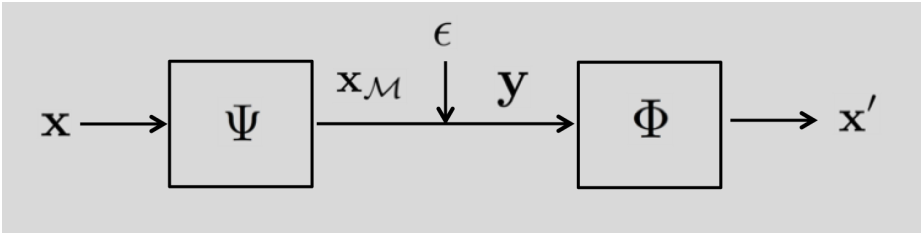


Figure 5.3: Sampling followed by recovery.

$$\text{sampling}: \quad \mathbf{y} = \Psi\mathbf{x} + \epsilon \equiv \mathbf{x}_{\mathcal{M}} + \epsilon \in \mathbb{R}^M, \tag{5.2}$$

$$\text{recovery}: \quad \mathbf{x}' = \Phi\mathbf{y} = \Phi\left(\Psi\mathbf{x} + \epsilon\right) \in \mathbb{R}^N, \tag{5.3}$$

where $\mathbf{x}' \in \mathbb{R}^N$ recovers $\mathbf{x}$ either exactly or approximately. In general, it is hard to recover $\mathbf{x}$. Here we focus on a graph signal model promoted by a graph dictionary,

$$\mathcal{X}_K = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \mathrm{D}\,\mathbf{a}, \mathbf{a} \in \mathbb{R}^S, \|\mathbf{a}\|_0 \leq K\},$$

where $\mathrm{D} \in \mathbb{R}^{N\times S}$ is a graph dictionary. The prototype of designing sampling and recovery strategies is

$$\begin{aligned} \Psi^*(\mathrm{D}), \Phi^*(\mathrm{D}) = \arg\min_{\Psi,\Phi} \max_{\|\mathbf{a}\|_0\leq K} \quad & d(\mathbf{x}', \mathbf{x}), \\ \text{subject to} \quad & \mathbf{x}' = \Phi\mathbf{y}, \\ & \mathbf{x} = \mathrm{D}\,\mathbf{a}, \end{aligned}$$

where $d(\cdot,\cdot)$ is some evaluation metric. For any graph signal $\mathbf{x} \in \mathcal{X}_K$, we want to find a pair of sampling and recovery strategies to minimize the reconstruction error. The minmax error is adopted to guarantee the sampling and recovery strategies work for any graph signal from $\mathcal{X}_K$. The optimal sampling and recovery strategies Ψ^*, Φ^* are influenced by the given graph dictionary D. We often consider fixing either the sampling strategy or the recovery strategy and optimizing over the other one.

We consider three different sampling strategies:

- *uniform sampling* means that sample indices are chosen from $\{0, 1, \ldots, N-1\}$ independently and randomly;
- *experimentally designed sampling* means that sample indices can be chosen beforehand; and
- *active sampling* means that sample indices can be chosen as a function of the sample points and the samples collected up to that instance, that is, $\mathcal{M}_i$ depends only on $\{\mathcal{M}_j, y_j\}_{j<i}$.

It is clear that uniform sampling is a subset of experimentally designed sampling, which is again a subset of active sampling.

Chapter 6

Sampling of Bandlimited Graph Signals

6.1 Introduction

As the bridge connecting sequences and functions, classical sampling theory shows that a bandlimited function can be perfectly recovered from its sampled sequence if the sampling rate is high enough [107]. More generally, we can treat any decrease in dimension via a linear operator as sampling, and, conversely, any increase in dimension via a linear operator as interpolation [84, 108]. Formulating a sampling theory in this context is equivalent to moving between higher- and lower-dimensional spaces.

A sampling theory for graphs has interesting applications. For example, given a graph representing friendship connectivity on Facebook, we can sample a fraction of users and query their hobbies and then recover all users' hobbies. The task of sampling of graph signals is, however, not well understood [60,65], because graph signals lie on complex, irregular structures. It is even more challenging to find a graph structure that is associated with the sampled signal coefficients; in the Facebook example, we sample a small fraction of users and an associated graph structure would allow us to infer new connectivity between those sampled users, even when they are not directly connected in the original graph.

Previous works on sampling theory [60, 61, 109] consider graph signals that are uniquely sampled onto a given subset of nodes. This approach is hard to apply to directed graphs. It also does not explain which graph structure supports these sampled coefficients.

In this chapter, we propose a novel sampling framework for graph signals. Here, the bandwidth definition is based on the number of non-zero signal coefficients in the graph Fourier domain. Since each signal coefficient in the graph Fourier domain corresponds to a graph frequency, the bandwidth definition is also based on the number of graph frequencies. This makes the proposed sampling framework strongly connected to linear algebra, that is, we are allowed to use simple tools from linear algebra to perform sampling on complex, irregular graphs.

6.2 Problem Formulation

Following the discussion in prologue of Part III, we aim to use a sampling operator Ψ (5.1) to measure a graph signal $\mathbf{x} \in \mathbb{R}^N$ and produce a sampled signal $\mathbf{x}_{\mathcal{M}} \in \mathbb{R}^M$ ($M < N$). We then

use a recovery operator Φ to interpolate $\mathbf{x}_{\mathcal{M}}$ and get $\mathbf{x}' \in \mathbb{R}^N$, which recovers $\mathbf{x}$ either exactly or approximately. Even in the noiseless case, perfect recovery happens for all $\mathbf{x}$ only when $\Phi\Psi$ is the identity matrix. This is not possible in general because $\mathrm{rank}(\Phi\Psi) \leq M < N$; it is, however, possible to do this for signals with specific structure that we will define as bandlimited graph signals, as in classical discrete signal processing.

We now define a class of bandlimited graph signals, which makes perfect recovery possible.

Definition 14. A graph signal is called *bandlimited* when there exists a $K \in \{0, 1, \cdots, N-1\}$ such that its graph Fourier transform $\widehat{\mathbf{x}}$ satisfies

$$\widehat{x}_k = 0 \quad \text{for all} \quad k \geq K.$$

The smallest such K is called the *bandwidth* of $\mathbf{x}$. A graph signal that is not bandlimited is called a *full-band graph signal*.

Note that the bandlimited graph signals here do not necessarily mean low-pass, or smooth. Since we do not specify the ordering of frequencies, we can reorder the eigenvalues and permute the corresponding eigenvectors in the graph Fourier transform matrix to choose any band in the graph Fourier domain. The bandlimited graph signals are smooth only when we sort the eigenvalues in a descending order. The bandlimited restriction here is equivalent to limiting the number of non-zero signal coefficients in the graph Fourier domain with known supports. This generalization is potentially useful to represent non-smooth graph signals.

Definition 15. The set of graph signals in $\mathbb{R}^N$ with bandwidth of at most K is a closed subspace denoted $\mathrm{BL}_K(\mathrm{V}^{-1})$, with V^{-1} as in (2.6).

When defining the bandwidth, we focus on the number of graph frequencies, while previous works [60] focus on the value of graph frequencies. There are two shortcomings to using the values of graph frequencies: (a) When considering the values of graph frequencies, we ignore the discrete nature of graphs; because graph frequencies are discrete, two cut-off graph frequencies on the same graph can lead to the same bandlimited space. For example, assume a graph has graph frequencies 0, 0.1, 0.4, 0.6 and 2; when we set the cut-off frequency to either 0.2 or 0.3, they lead to the same bandlimited space; (b) The values of graph frequencies cannot be compared between different graphs. Since each graph has its own graph frequencies, a same value of the cut-off graph frequency on two graphs can mean different things. For example, one graph has graph frequencies as 0, 0.1, 0.2, 0.4 and 2, and another has graph frequencies 0, 1.1, 1.6, 1.8, and 2; when we set the cut-off frequency to 1, that is, we preserve all the graph frequencies that are no greater than 1, first graph preserves three out of four graph frequencies and the second graph only preserves one out of four. The values of graph frequencies thus do not necessarily give a direct and intuitive understanding about the bandlimited space. Another key advantage of using the number of graph frequencies is to build a strong connection to linear algebra allowing for the use of simple tools from linear algebra in sampling and interpolation of bandlimited graph signals.

6.3 Methodology

6.3.1 Sampling Theory for Graph Signals

In Theorem 5.2 in [84], the authors show the recovery for vectors via projection, which lays the theoretical foundation for the classical sampling theory. Following the theorem, we obtain the following result, the proof of which can be found in [108].

Theorem 11. Let Ψ satisfy

$$\operatorname{rank}(\Psi \, \mathrm{V}_{(K)}) = K,$$

where $\mathrm{V}_{(K)} \in \mathbb{C}^{N \times K}$ denotes the first K columns of V. For all $\mathbf{x} \in \mathrm{BL}_K(\mathrm{V}^{-1})$, perfect recovery, $\mathbf{x} = \Phi\Psi\mathbf{x}$, is achieved by choosing

$$\Phi = \mathrm{V}_{(K)} \, \mathrm{U},$$

with $\mathrm{U} \, \Psi \, \mathrm{V}_{(K)}$ a $K \times K$ identity matrix.

Theorem 11 is applicable for all graph signals that have a few non-zero elements in the graph Fourier domain with known supports, that is, $K < N$.

Similarly to the classical sampling theory, the sampling rate has a lower bound for graph signals as well, that is, the sample size M should be no smaller than the bandwidth K. When $M < K$, $\operatorname{rank}(\mathrm{U} \, \Psi \, \mathrm{V}_{(K)}) \leq \operatorname{rank}(\mathrm{U}) \leq M < K$, and thus, $\mathrm{U} \, \Psi \, \mathrm{V}_{(K)}$ can never be an identity matrix. For $\mathrm{U} \, \Psi \, \mathrm{V}_{(K)}$ to be an identity matrix, U is the inverse of $\Psi \, \mathrm{V}_{(K)}$ when $M = K$; it is a pseudo-inverse of $\Psi \, \mathrm{V}_{(K)}$ when $M > K$, where the redundancy can be useful for reducing the influence of noise. For simplicity, we only consider $M = K$ and U invertible. When $M > K$, we simply select K out of M sampled signal coefficients to ensure that the sample size and the bandwidth are the same.

From Theorem 11, we see that an arbitrary sampling operator may not lead to perfect recovery even for bandlimited graph signals. When the sampling operator Ψ satisfies the full-rank assumption (6.1), we call it a *qualified sampling operator*. To satisfy (6.1), the sampling operator should select at least one set of K linearly-independent rows in $\mathrm{V}_{(K)}$. Since V is invertible, the column vectors in V are linearly independent and $\operatorname{rank}(\mathrm{V}_{(K)}) = K$ always holds; in other words, at least one set of K linearly-independent rows in $\mathrm{V}_{(K)}$ always exists. Since the graph shift A is given, one can find such a set independently of the graph signal. Given such a set, Theorem 11 guarantees perfect recovery of bandlimited graph signals. To find linearly-independent rows in a matrix, fast algorithms exist, such as QR decomposition; see [75, 84]. Since we only need to know the graph structure to design a *qualified sampling operator*, this follows the experimentally designed sampling. We will expand this topic in Section 6.3.5.

6.3.2 Sampled Graph Signal

We just showed that perfect recovery is possible when the graph signal is bandlimited. We now show that the sampled signal coefficients form a new graph signal, whose corresponding graph shift can be constructed from the original graph shift.

Although the following results can be generalized to $M > K$ easily, we only consider $M = K$ for simplicity. Let the sampling operator Ψ and the interpolation operator Φ satisfy

Figure 6.1: Sampling followed by interpolation. The arrows indicate that the edges are directed.

the conditions in Theorem 11. For all $\mathbf{x} \in \mathrm{BL}_K(\mathrm{V}^{-1})$, we have

$$\begin{aligned} \mathbf{x} \;=\; \Phi\Psi\mathbf{x} \;=\; \Phi\mathbf{x}_{\mathcal{M}} \;&\overset{(a)}{=}\; \mathrm{V}_{(K)}\,\mathrm{U}\,\mathbf{x}_{\mathcal{M}} \\ &\overset{(b)}{=}\; \mathrm{V}_{(K)}\,\widehat{\mathbf{x}}_{(K)}, \end{aligned}$$

where $\widehat{\mathbf{x}}_{(K)}$ denotes the first K coefficients of $\widehat{\mathbf{x}}$, (a) follows from Theorem 11 and (b) from Definition 14. We thus get

$$\widehat{\mathbf{x}}_{(K)} \;=\; \mathrm{U}\,\mathbf{x}_{\mathcal{M}},$$

and

$$\mathbf{x}_{\mathcal{M}} \;=\; \mathrm{U}^{-1}\,\mathrm{U}\,\mathbf{x}_{\mathcal{M}} = \mathrm{U}^{-1}\,\widehat{\mathbf{x}}_{(K)}.$$

From what we have seen, the sampled signal coefficients $x_{\mathcal{M}}$ and the frequency content $\widehat{\mathbf{x}}_{(K)}$ form a Fourier pair because $\mathbf{x}_{\mathcal{M}}$ can be constructed from $\widehat{\mathbf{x}}_{(K)}$ through U^{-1} and $\widehat{\mathbf{x}}_{(K)}$ can also be constructed from $\mathbf{x}_{\mathcal{M}}$ through U. This implies that, according to the spectral decomposition (2.6), $\mathbf{x}_{\mathcal{M}}$ is a graph signal associated with the graph Fourier transform matrix U and a new graph shift

$$\mathrm{A}_{\mathcal{M}} = \mathrm{U}^{-1}\,\Lambda_{(K)}\,\mathrm{U} \in \mathbb{C}^{K\times K},$$

where $\Lambda_{(K)} \in \mathbb{C}^{K\times K}$ is a diagonal matrix that samples the first K eigenvalues of Λ. This leads to the following theorem.

Theorem 12. Let $\mathbf{x} \in \mathrm{BL}_K(\mathrm{V}^{-1})$ and let

$$\mathbf{x}_{\mathcal{M}} = \Psi x \in \mathbb{R}^K$$

be its sampled version, where Ψ is a qualified sampling operator. Then, the graph shift associated with the graph signal $\mathbf{x}_{\mathcal{M}}$ is

$$\mathrm{A}_{\mathcal{M}} = \mathrm{U}^{-1}\,\Lambda_{(K)}\,\mathrm{U} \in \mathbb{C}^{K\times K}, \tag{6.1}$$

with $\mathrm{U} = (\Psi\,\mathrm{V}_{(K)})^{-1}$.

From Theorem 12, we see that the graph shift $\mathrm{A}_{\mathcal{M}}$ is constructed by sampling the rows of the eigenvector matrix and sampling the first K eigenvalues of the original graph shift A. We simply say that $\mathrm{A}_{\mathcal{M}}$ is sampled from A, preserving certain information in the graph Fourier domain.

Since the bandwidth of $\mathbf{x}$ is K, the first K coefficients in the frequency domain are $\widehat{\mathbf{x}}_{(K)} = \widehat{\mathbf{x}}_{\mathcal{M}}$, and the other $N-K$ coefficients are $\widehat{\mathbf{x}}_{(-K)} = 0$; in other words, the frequency contents of the original graph signal $\mathbf{x}$ and the sampled graph signal $\mathbf{x}_{\mathcal{M}}$ are equivalent after performing their corresponding graph Fourier transforms.

Similarly to Theorem 11, by reordering the eigenvalues and permuting the corresponding eigenvectors in the graph Fourier transform matrix, Theorem 12 is applicable to all graph signals that have limited support in the graph Fourier domain.

6.3.3 Property of A Sampled Graph Signal

We argued that $\mathrm{A}_{\mathcal{M}} = \mathrm{U}^{-1}\,\Lambda_{(K)}\,\mathrm{U}$ is the graph shift that supports the sampled signal coefficients $\mathbf{x}_{\mathcal{M}}$ following from a mathematical equivalence between the graph Fourier transform for the sampled graph signal and the graph shift. We, in fact, implicitly proposed an approach to sampling graphs. Since sampled graphs always lose information, we now study which information $\mathrm{A}_{\mathcal{M}}$ preserves.

Theorem 13. For all $\mathbf{x} \in \mathrm{BL}_K(\mathrm{V}^{-1})$,

$$\mathbf{x}_{\mathcal{M}} - \mathrm{A}_{\mathcal{M}}\,\mathbf{x}_{\mathcal{M}} \;=\; \Psi\,(x - \mathrm{A}\,x)\,.$$

Proof.

$$\begin{aligned}\mathbf{x}_{\mathcal{M}} - \mathrm{A}_{\mathcal{M}}\,\mathbf{x}_{\mathcal{M}} &= \mathrm{U}^{-1}\,\widehat{\mathbf{x}}_{\mathcal{M}} - \mathrm{U}^{-1}\,\Lambda_{(K)}\,\mathrm{U}\,\mathrm{U}^{-1}\,\widehat{\mathbf{x}}_{\mathcal{M}}\\ &= \Psi\,\mathrm{V}_{(K)}(\mathbf{I} - \Lambda_{(K)})\widehat{\mathbf{x}}_{(K)}\\ &= \Psi\,(x - \mathrm{A}\,x)\,,\end{aligned}$$

where the last equality follows from $\mathbf{x} \in \mathrm{BL}_K(\mathrm{V}^{-1})$. □

The term $x - \mathrm{A}\,x$ measures the difference between the original graph signal and its shifted version. This is also called the first-order difference of $\mathbf{x}$, while the term $\Psi\,(x - \mathrm{A}\,x)$ measures the first-order difference of $\mathbf{x}$ at sampled indices. Furthermore, $\|\mathbf{x} - \mathrm{A}\,\mathbf{x}\|_p^p$ is the graph total variation based on the ℓ_p-*norm*, which is a quantitative characteristic that measures the smoothness of a graph signal [76]. When using a sampled graph to represent the sampled signal coefficients, we lose the connectivity information between the sampled nodes and all the other nodes; despite this, $\mathrm{A}_{\mathcal{M}}$ still preserves the first-order difference at sampled indices. Instead of focusing on preserving connectivity properties as in prior work [24], we emphasize the interplay between signals and structures.

6.3.4 Example

We consider a five-node directed graph with graph shift

$$\mathrm{A} = \begin{bmatrix} 0 & \frac{2}{5} & \frac{2}{5} & 0 & \frac{1}{5} \\ \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 \\ \frac{1}{2} & \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \end{bmatrix}.$$

The corresponding inverse graph Fourier transform matrix is

$$\mathrm{V} = \begin{bmatrix} 0.45 & 0.19 & 0.25 & 0.35 & -0.40 \\ 0.45 & 0.40 & 0.16 & -0.74 & 0.18 \\ 0.45 & 0.08 & -0.56 & 0.29 & 0.36 \\ 0.45 & -0.66 & -0.41 & -0.47 & -0.57 \\ 0.45 & -0.60 & 0.66 & 0.13 & 0.59 \end{bmatrix},$$

and the frequencies are

$$\Lambda = \operatorname{diag}\begin{bmatrix} 1 & 0.39 & -0.12 & -0.44 & -0.83 \end{bmatrix}.$$

Let $K = 3$; generate a bandlimited graph signal $\mathbf{x} \in \mathrm{BL}_3(\mathrm{V}^{-1})$ as

$$\widehat{\mathbf{x}} = \begin{bmatrix} 0.5 & 0.2 & 0.1 & 0 & 0 \end{bmatrix}^T,$$

with

$$\mathbf{x} = \begin{bmatrix} 0.29 & 0.32 & 0.18 & 0.05 & 0.17 \end{bmatrix}^T,$$

and the first-order difference of x is

$$\mathbf{x} - \mathrm{A}\,\mathbf{x} = \begin{bmatrix} 0.05 & 0.07 & -0.05 & -0.13 & 0.0002 \end{bmatrix}^T.$$

We can check the first three columns of V to see that all sets of three rows are independent. According to the sampling theorem, we can then recover $\mathbf{x}$ perfectly by sampling any three of its coefficients; for example, sample the first, second and the fourth coefficients. Then, $\mathcal{M} = (1, 2, 4)$, $\mathbf{x}_{\mathcal{M}} = \begin{bmatrix} 0.29 & 0.32 & 0.05 \end{bmatrix}^T$, and the sampling operator

$$\Psi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

is qualified. We recover $\mathbf{x}$ by using the following interpolation operator (see Figure 6.1)

$$\Phi = \mathrm{V}_{(3)}(\Psi\,\mathrm{V}_{(3)})^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2.7 & 2.87 & 0.83 \\ 0 & 0 & 1 \\ 5.04 & -3.98 & -0.05 \end{bmatrix}.$$

The inverse graph Fourier transform matrix for the sampled signal is

$$\mathrm{U}^{-1} = \Psi\,\mathrm{V}_{(3)} = \begin{bmatrix} 0.45 & 0.19 & 0.25 \\ 0.45 & 0.40 & 0.16 \\ 0.45 & -0.66 & -0.41 \end{bmatrix},$$

and the sampled frequencies are

$$\Lambda_{(3)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.39 & 0 \\ 0 & 0 & -0.12 \end{bmatrix}.$$

The sampled graph shift is then constructed as

$$\mathrm{A}_{\mathcal{M}} = \mathrm{U}^{-1}\,\Lambda_{(3)}\,\mathrm{U} = \begin{bmatrix} 0.07 & 0.75 & 0.32 \\ -0.23 & 0.96 & 0.28 \\ 1.17 & -0.56 & 0.39 \end{bmatrix}.$$

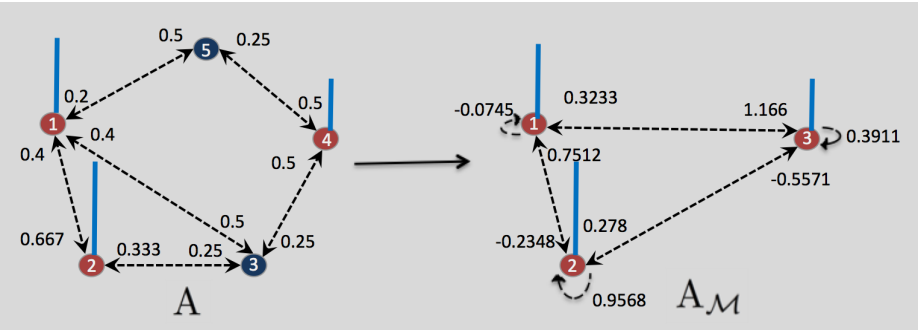


Figure 6.2: Sampling a graph.

The first-order difference of $\mathbf{x}_{\mathcal{M}}$ is

$$\mathbf{x}_{\mathcal{M}} - \mathrm{A}_{\mathcal{M}}\,\mathbf{x}_{\mathcal{M}} = \begin{bmatrix} 0.05 & 0.07 & -0.13 \end{bmatrix}^T \;=\; \Psi(\mathbf{x} - \mathrm{A}\,\mathbf{x}).$$

We see that the sampled graph shift contains self-loops and negative weights and seems to be dissimilar to A, but $\mathrm{A}_{\mathcal{M}}$ preserves a part of the frequency content of A because U^{-1} is sampled from V and $\Lambda_{(3)}$ is sampled from A. $\mathrm{A}_{\mathcal{M}}$ also preserves the first-order difference of x, which validates Theorem 13.

6.3.5 Sampling with A Qualified Sampling Operator

As shown in Section 6.3.1, only a qualified sampling operator (6.1) can lead to perfect recovery for bandlimited graph signals. Since a qualified sampling operator (6.1) is designed via the graph structure, it belongs to experimentally designed sampling. The design consist in finding K linearly independent rows in $\mathrm{V}_{(K)}$, which gives multiple choices. In this section, we propose an optimal approach to designing a qualified sampling operators by minimizing the effect of noise for general graphs. We then show that for some specific graphs, random sampling also leads to perfect recovery with high probability.

6.3.6 Experimentally Designed Sampling

We now show how to design a qualified sampling operator on any given graph that is robust to noise. We then compare this optimal sampling operator with a random sampling operator on a sensor network.

Optimal Sampling Operator

As mentioned in Section 6.3.1, at least one set of K linearly-independent rows in $\mathrm{V}_{(K)}$ always exists. When we have multiple choices of K linearly-independent rows, we aim to find the optimal one to minimize the effect of noise.

We consider a model where noise $\mathbf{e}$ is introduced during sampling as follows,

$$\mathbf{y} \;=\; \Psi\mathbf{x} + \mathbf{e},$$

where Ψ is a qualified sampling operator. The recovered graph signal, $\mathbf{x}'_e$, is then

$$\mathbf{x}'_e \;=\; \Phi\mathbf{y} \;=\; \Phi\Psi\mathbf{x} + \Phi\mathbf{e} \;=\; \mathbf{x} + \Phi\mathbf{e}.$$

To bound the effect of noise, we have

$$\begin{aligned} \|\mathbf{x}' - \mathbf{x}\|_2 \;&=\; \|\Phi\mathbf{e}\|_2 \;=\; \left\|\mathrm{V}_{(K)}\,\mathrm{U}\,\mathbf{e}\right\|_2 \\ &\leq\; \left\|\mathrm{V}_{(K)}\right\|_2 \|\mathrm{U}\|_2 \,\|\mathbf{e}\|_2\,, \end{aligned}$$

where the inequality follows from the definition of the spectral norm. Since $\left\|\mathrm{V}_{(K)}\right\|_2$ and $\|\mathbf{e}\|_2$ are fixed, we want U to have a small spectral norm. From this perspective, for each feasible Ψ, we compute the inverse or pseudo-inverse of $\Psi\,\mathrm{V}_{(K)}$ to obtain U; the best choice comes from the U with the smallest spectral norm. This is equivalent to maximizing the smallest singular value of $\Psi\,\mathrm{V}_{(K)}$,

$$\Psi^{opt} \;=\; \arg\max_{\Psi} \sigma_{\min}(\Psi\,\mathrm{V}_{(K)}), \tag{6.2}$$

where $\sigma_{\min}$ denotes the smallest singular value. The solution of (6.2) is optimal in terms of minimizing the effect of noise; we simply call it *optimal sampling operator*. Since we restrict the form of Ψ in (5.1), (6.2) is non-deterministic polynomial-time hard. To solve (6.2), we can use a greedy algorithm as shown in Algorithm 2. In a previous work, the authors solved a similar optimization problem for matrix approximation and showed that the greedy algorithm gives a good approximation to the global optimum [110]. Note that $\mathcal{M}$ is the sampling sequence, indicating which rows to select, and $(\mathrm{V}_{(K)})_{\mathcal{M}}$ denotes the sampled rows from $\mathrm{V}_{(K)}$. When increasing the number of samples, the smallest singular value of $\Psi\,\mathrm{V}_{(K)}$ grows, and thus, redundant samples make the algorithm robust to noise.

Algorithm 2 Optimal Sampling Operator via Greedy Algorithm

Input $\mathrm{V}_{(K)}$ the first K columns of V
M the number of samples
Output $\mathcal{M}$ sampling set
Function
while $|\mathcal{M}| < M$
$m \;=\; \arg\max_i \sigma_{\min}\left((\mathrm{V}_{(K)})_{\mathcal{M}+\{i\}}\right)$
$\mathcal{M} \leftarrow \mathcal{M} + \{m\}$
end
return $\mathcal{M}$

6.3.7 Relations & Extensions

We now discuss three topics: relation to the sampling theory for finite discrete-time signals, relation to compressed sensing, and how to handle a full-band graph signal.

6.3.8 Relation to Sampling Theory for Finite Discrete-Time Signals

We call the graph that supports a finite discrete-time signal a *finite discrete-time graph*, which specifies the time-ordering from the past to future. The finite discrete-time graph can be represented by the cyclic permutation matrix [76, 84],

$$\mathrm{A} \;=\; \begin{bmatrix} 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 1 & 0 \end{bmatrix} \;=\; \mathrm{V}\,\Lambda\,\mathrm{V}^{-1}, \tag{6.3}$$

where the eigenvector matrix

$$\mathrm{V} = \begin{bmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \cdots & \mathbf{v}_{N-1} \end{bmatrix} = \left[\tfrac{1}{\sqrt{N}} (W^{jk})^* \right]_{j,k=0,\cdots N-1} \tag{6.4}$$

is the Hermitian transpose of the N-point discrete Fourier transform matrix, $\mathrm{V} = \mathbf{DFT}^*$, V^{-1} is the N-point discrete Fourier transform matrix, $\mathrm{V}^{-1} = \mathbf{DFT}$, and the eigenvalue matrix is

$$\Lambda = \mathrm{diag} \begin{bmatrix} W^0 & W^1 & \cdots & W^{N-1} \end{bmatrix}, \tag{6.5}$$

where $W = e^{-2\pi j/N}$. We see that Definitions 14, 15 and Theorem 11 are immediately applicable to finite discrete-time signals and are consistent with sampling of such signals [84].

Definition 16. A discrete-time signal is called *bandlimited* when there exists $K \in \{0, 1, \cdots, N-1\}$ such that its discrete Fourier transform $\widehat{\mathbf{x}}$ satisfies

$$\widehat{x}_k = 0 \quad \text{for all} \quad k \geq K.$$

The smallest such K is called the *bandwidth* of $\mathbf{x}$. A discrete-time signal that is not bandlimited is called a *full-band discrete-time signal*.

Definition 17. The set of discrete-time signals in $\mathbb{C}^N$ with bandwidth of at most K is a closed subspace denoted $\mathrm{BL}_K(\mathbf{DFT})$, with $\mathbf{DFT}$ the discrete Fourier transform matrix.

With this definition of the discrete Fourier transform matrix, the highest frequency is in the middle of the spectrum (although this is just a matter of ordering). From Definitions 16 and 17, we can permute the rows in the discrete Fourier transform matrix to choose any frequency band. Since the discrete Fourier transform matrix is a Vandermonde matrix, any K rows of $\mathbf{DFT}^*_{(K)}$ are independent [75, 84]; in other words, $\mathrm{rank}(\Psi\,\mathbf{DFT}^*_{(K)}) = K$ always holds when $M \geq K$. We apply now Theorem 11 to obtain the following result.

Corollary 2. Let Ψ satisfy that the sampling number is no smaller than the bandwidth, $M \geq K$. For all $\mathbf{x} \in \mathrm{BL}_K(\mathbf{DFT})$, perfect recovery, $\mathbf{x} = \Phi\Psi\mathbf{x}$, is achieved by choosing

$$\Phi = \mathbf{DFT}^*_{(K)}\,\mathrm{U},$$

with $\mathrm{U}\,\Psi\,\mathbf{DFT}^*_{(K)}$ a $K \times K$ identity matrix, and $\mathbf{DFT}^*_{(K)}$ denotes the first K columns of $\mathbf{DFT}^*$.

From Corollary 2, we can perfectly recover a discrete-time signal when it is bandlimited.

Similarly to Theorem 12, we can show that a new graph shift can be constructed from the finite discrete-time graph. Multiple sampling mechanisms can be used to sample a new graph shift; an intuitive one is as follows: let $\mathbf{x} \in \mathbb{C}^N$ be a finite discrete-time signal, where N is even. Reorder the frequencies in (6.5), by putting the frequencies with even indices first,

$$\widetilde{\Lambda} = \operatorname{diag}\begin{bmatrix}\lambda_0 & \lambda_2 & \cdots & \lambda_{N-2} & \lambda_1 & \lambda_3 & \cdots & \lambda_{N-1}\end{bmatrix}.$$

Similarly, reorder the columns of V in (6.4) by putting the columns with even indices first

$$\widetilde{\mathrm{V}} = \begin{bmatrix}\mathbf{v}_0 & \mathbf{v}_2 & \cdots & \mathbf{v}_{N-2} & \mathbf{v}_1 & \mathbf{v}_3 & \cdots & \mathbf{v}_{N-1}\end{bmatrix}.$$

One can check that $\widetilde{\mathrm{V}}\widetilde{\Lambda}\widetilde{\mathrm{V}}^{-1}$ is still the same cyclic permutation matrix. Suppose we want to preserve the first $N/2$ frequency components in $\widetilde{\Lambda}$; the sampled frequencies are then

$$\widetilde{\Lambda}_{(N/2)} = \operatorname{diag}\begin{bmatrix}\lambda_0 & \lambda_2 & \cdots & \lambda_{N-2}\end{bmatrix}.$$

Let a sampling operator Ψ choose the first $N/2$ rows in $\widetilde{\mathrm{V}}_{(N/2)}$,

$$\Psi\widetilde{\mathrm{V}}_{(N/2)} = \left[\tfrac{1}{\sqrt{N}}(W^{2jk})^*\right]_{j,k=0,\cdots N/2-1},$$

which is the Hermitian transpose of the discrete Fourier transform of size $N/2$ and satisfies $\operatorname{rank}(\Psi\widetilde{\mathrm{V}}_{(N/2)}) = N/2$ in Theorem 12. The sampled graph Fourier transform matrix $\mathrm{U} = (\Psi\widetilde{\mathrm{V}}_{(N/2)}))^{-1}$ is the discrete Fourier transform of size $N/2$. The sampled graph shift is then constructed as

$$\mathrm{A}_{\mathcal{M}} = \mathrm{U}^{-1}\,\widetilde{\Lambda}_{(N/2)}\,\mathrm{U},$$

which is exactly the $N/2 \times N/2$ cyclic permutation matrix. Hence, we have shown that by choosing an appropriate sampling mechanism, a smaller finite discrete-time graph is obtained from a larger finite discrete-time graph by using Theorem 12. We note that using a different ordering or sampling operator would result in a graph shift that can be different and non-intuitive. This is simply a matter of choosing different frequency components.

6.3.9 Relation to Compressed Sensing

Compressed sensing is a sampling framework to recover sparse signals with a few measurements [111]. The theory asserts that a few samples guarantee the recovery of the original signals when the signals and the sampling approaches are well-defined in some theoretical aspects. To be more specific, given the sampling operator $\Psi \in \mathbb{R}^{M\times N}, M \ll N$, and the sampled signal $\mathbf{x}_{\mathcal{M}} = \Psi\mathbf{x}$, a sparse signal $\mathbf{x} \in \mathbb{R}^N$ is recovered by solving

$$\min_{\mathbf{x}} \ \|\mathbf{x}\|_0, \ \text{subject to}\, \mathbf{x}_{\mathcal{M}} = \Psi\mathbf{x}. \tag{6.6}$$

Since the l_0 norm is not convex, the optimization is a non-deterministic polynomial-time hard problem. To obtain a computationally efficient algorithm, the l_1-norm based algorithm, known as

basis pursuit or basis pursuit with denoising, recovers the sparse signal with small approximation error [112].

In the standard compressed sensing theory, signals have to be sparse or approximately sparse to guarantee accurate recovery properties. In [113], the authors proposed a general way to perform compressed sensing with non-sparse signals using dictionaries. Specifically, a general signal $\mathbf{x} \in \mathbb{R}^N$ is recovered by

$$\min_{\mathbf{x}} \ \|\mathrm{D}\,\mathbf{x}|_0, \ \text{ subject to } \mathbf{x}_{\mathcal{M}} = \Psi\mathbf{x}, \tag{6.7}$$

where D is a dictionary designed to make $\mathrm{D}\,\mathbf{x}$ sparse. When specifying $\mathbf{x}$ to be a graph signal, and D to be the appropriate graph Fourier transform of the graph on which the signal resides, $\mathrm{D}\,\mathbf{x}$ represents the frequency content of $\mathbf{x}$, which is sparse when $\mathbf{x}$ is of limited bandwidth. Equation (6.7) recovers a bandlimited graph signal from a few sampled signal coefficients via an optimization approach. The proposed sampling theory deals with the cases where the frequencies corresponding to non-zero elements are known, and can be reordered to form a bandlimited graph signal. Compressed sensing deals with the cases where the frequencies corresponding to non-zero elements are unknown, which is a more general and harder problem. If we have access to the position of the non-zero elements, the proposed sampling theory uses the smallest number of samples to achieve perfect recovery.

6.3.10 Graph Downsampling & Graph Filter Banks

In classical signal processing, sampling refers to sampling a continuous function and downsampling refers to sampling a sequence. Both concepts use fewer samples to represent the overall shape of the original signal. Since a graph signal is discrete in nature, sampling and downsampling are the same. Previous works implemented graph downsampling via graph coloring [47] or minimum spanning tree [114].

The proposed sampling theory provides a family of qualified sampling operators (6.1) with an optimal sampling operator as in (6.2). To downsample a graph by 2, one can set the bandwidth to a half of the number of nodes, that is, $K = N/2$, and use (6.2) to obtain an optimal sampling operator. An example for the finite discrete-time signals was shown in Section 6.3.8.

As shown in Theorem 11, perfect recovery is achieved when graph signals are bandlimited. To handle full-band graph signals, we propose an approach based on graph filter banks, where each channel does not need to recover perfectly but in conjunction they do.

Let $\mathbf{x}$ be a full-band graph signal, which, without loss of generality, we can express without loss of generality as the addition of two bandlimited signals supported on the same graph, that is, $\mathbf{x} = \mathbf{x}^l + \mathbf{x}^h$, where

$$\mathbf{x}^l = \mathrm{P}^l\,\mathbf{x}, \ \ \mathbf{x}^h = \mathrm{P}^h\,\mathbf{x},$$

and

$$\mathrm{P}^l \ = \ \mathrm{V}\begin{bmatrix}\mathbf{I}_K & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathrm{V}^{-1}, \ \ \mathrm{P}^h \ = = \mathrm{V}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_{N-K}\end{bmatrix}\mathrm{V}^{-1}.$$

We see that $\mathbf{x}^l$ contains the first K frequencies, $\mathbf{x}^h$ contains the other $N-K$ frequencies, and each is bandlimited. We do sampling and interpolation for $\mathbf{x}^l$ and $\mathbf{x}^h$ in two channels, respectively.

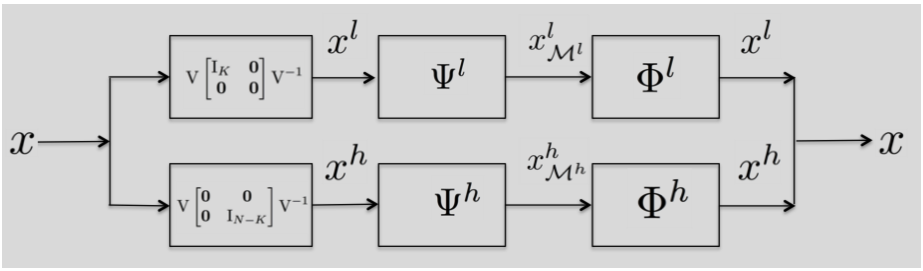


Figure 6.3: Graph filter bank that splits the graph signal into two bandlimited graph signals. In each channel, we perform sampling and interpolation, following Theorem 11. Finally, we add the results from both channels to obtain the original full-band graph signal.

Take the first channel as an example. Following Theorems 11 and 12, we use a qualified sampling operator Ψ^l to sample $\mathbf{x}^l$, and obtain the sampled signal coefficients as $\mathbf{x}^l_{\mathcal{M}^l} = \Psi^l \mathbf{x}^l$, with the corresponding graph as $\mathrm{A}_{\mathcal{M}^l}$. We can recover $\mathbf{x}^l$ by using interpolation operator Φ^l as $\mathbf{x}^l = \Phi^l \mathbf{x}^l_{\mathcal{M}^l}$. Finally, we add the results from both channels to obtain the original full-band graph signal (also illustrated in Figure 6.3). The main benefit of working with a graph filter bank is that, instead of dealing with a long graph signal with a large graph, we are allowed to focus on the frequency bands of interests and deal with a shorter graph signal with a small graph in each channel.

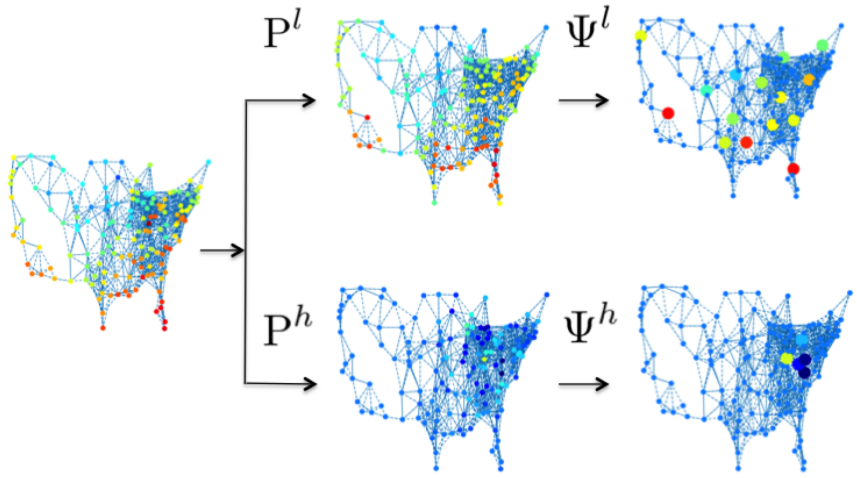
We do not restrict the samples from two bands, $\mathbf{x}^l_{\mathcal{M}^l}$ and $\mathbf{x}^h_{\mathcal{M}^h}$ the same size because we can adaptively design the sampling and interpolation operators based on their own sizes. This is similar to the filter banks in the classical literature where the spectrum is not evenly partitioned between the channels [115]. We see that the above idea can easily be generalized to multiple channels by splitting the original graphs signal into multiple bandlimited graph signals; instead of dealing with a huge graph, we work with multiple small graphs, which makes computation easier.

6.4 Experimental Results

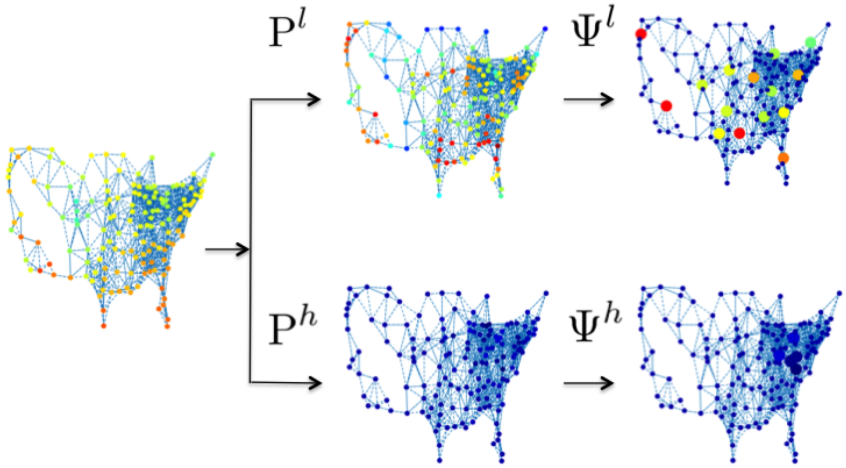
We now show an example where we analyze graph signals by using the proposed graph filter banks. We consider that the weather stations across the U.S. form a graph and temperature values measured at each weather station in one day form a graph signal. Suppose that a high-frequency component represents some pattern of weather change; we want to detect this pattern given the temperature values. We can decompose a graph signal of temperature values into a low-frequency channel (largest 15 frequencies) and a high-frequency channel (smallest 5 frequencies). In each channel, we sample the bandlimited graph signal to obtain a sparse and loseless representation. Figure 6.4 shows a comparison between temperature values on January 1st, 2013 and May 1st, 2013. We intentionally added some high-frequency component to the temperature values on January 1st, 2013. We see that the high-frequency channel in Figure 6.4 (a) detects the change, while the high-frequency channel in Figure 6.4 (b) does not.

Directly using the graph frequency components can also detect the high-frequency components, but the graph frequency components cannot be easily visualized. Since the graph structure

and the decomposed channels are fixed, the optimal sampling operator and corresponding interpolation operator in each channel can be designed in advance, which means that we just need to look at the sampled coefficients of a fixed set of nodes to check whether a channel is activated. The graph filter bank is thus fast and visualization friendly.



(a) Temperature on January 1st, 2013 with high-frequency pattern.



(b) Temperature on May 1st, 2013.

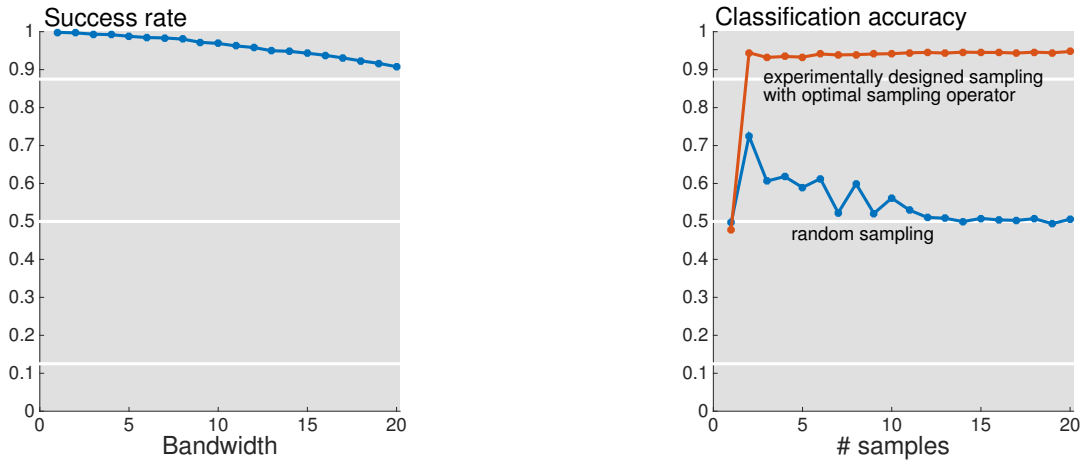
Figure 6.4: Graph filter banks analysis.

The proposed sampling theory on graphs can be applied to semi-supervised learning, whose goal is to classify data with a few labeled and a large number of unlabeled samples [106]. One approach is based on graphs, where the labels are often assumed to be smooth on graphs. From a perspective of signal processing, smoothness can be expressed as lowpass nature of the signal. Recovering smooth labels on graphs is then equivalent to interpolating a low-pass graph signal. We now look at two examples, including classification of online blogs and handwritten digits.

Sampling Online Blogs

We first aim to investigate the success rate of perfect recovery by using random sampling, and then classify the labels of the online blogs. We consider a dataset of $N = 1224$ online political blogs as either conservative or liberal [116]. We represent conservative labels as $+1$ and liberal ones as -1. The blogs are represented by a graph in which nodes represent blogs, and directed graph edges correspond to hyperlink references between blogs. The graph signal here is the label assigned to the blogs, called the labeling signal. We use the spectral decomposition in (2.6) for this online-blog graph to get the graph frequencies in a descending order and the corresponding graph Fourier transform matrix. The labeling signal is a full-band signal, but approximately bandlimited.

To investigate the success rate of perfect recovery by using random sampling, we vary the bandwidth K of the labeling signal with an interval of 1 from 1 to 20, randomly sample K rows from the first K columns of the graph Fourier transform matrix, and check whether the $K \times K$ matrix has full rank. For each bandwidth, we randomly sample 10,000 times, and count the number of successes to obtain the success rate. Figure 6.5 (a) shows the resulting success rate. We see that the success rate decreases as we increase the bandwidth; it is above 90%, when the bandwidth is no greater than 20. It means that we can achieve perfect recovery by using random sampling with a fairly high probability. As the bandwidth increases, even if we get an equal number of samples, the success rate still decreases, because when we take more samples, it is easier to get a sample correlated with the previous samples.



(a) Success rate as a function of bandwidth. (b) Classification accuracy as a function of the number of samples.

Figure 6.5: Classification for online blogs. When increasing the bandwidth, it is harder to find a qualified sampling operator. The experimentally designed sampling with the optimal sampling operator outperforms random sampling.

Since a qualified sampling operator is independent of graph signals, we precompute the qualified sampling operator for the online-blog graph, as discussed in Section 6.3.1. When the labeling signal is bandlimited, we can sample M labels from it by using a qualified sampling operator, and recover the labeling signal by using the corresponding interpolation operator. In other words,

we can design a set of blogs to label before querying any label. Most of the time, however, the labeling signal is not bandlimited, and it is not possible to achieve perfect recovery. Since we only care about the sign of the labels, we use only the low frequency content to approximate the labeling signal; after that, we set a threshold to assign labels. To minimize the influence from the high-frequency content, we can use the optimal sampling operator in Algorithm 2.

We solve the following optimization problem to recover the low frequency content,

$$\widehat{\mathbf{x}}_{(K)}^{opt} \;=\; \arg \min_{\widehat{\mathbf{x}}_{(K)} \in \mathbb{R}^K} \; \left\| \mathrm{sgn}(\Psi \, \mathrm{V}_{(K)} \, \widehat{\mathbf{x}}_{(K)}) - \mathbf{x}_{\mathcal{M}} \right\|_2^2, \tag{6.8}$$

where $\Psi \in \mathbb{R}^{M \times N}$ is a sampling operator, $\mathbf{x}_{\mathcal{M}} \in \mathbb{R}^M$ is a vector of the sampled labels whose element is either $+1$ or -1, and $\mathrm{sgn}(\cdot)$ sets all positive values to $+1$ and all negative values to -1. Note that without $\mathrm{sgn}(*)$, the solution of (6.8) is $(\Psi \, \mathrm{V}_{(K)})^{-1} \mathbf{x}_{\mathcal{M}}$ in Theorem 11, which perfectly recovers the labeling signal when it is bandlimited. When the labeling signal is not bandlimited, the solution of (6.8) approximates the low-frequency content. The ℓ_2 norm (6.8) can be relaxed by the logit function and solved by logistic regression [117]. The recovered labels are then $\mathbf{x}^{opt} = \mathrm{sgn}(\mathrm{V}_{(K)} \, \widehat{\mathbf{x}}_{(K)}^{opt})$.

Figure 6.5 (b) compares the classification accuracies between optimal sampling and random sampling by varying the sample size with an interval of 1 from 1 to 20. We see that the optimal sampling significantly outperforms random sampling, and random sampling does not improve with more samples, because the interpolation operator (6.8) assumes that the sampling operator is qualified, which is not always true for random sampling as shown in Figure 6.5 (a). Note that classification accuracy for the optimal sampling is as high as 94.44% by only sampling two blogs, and the classification accuracy gets slightly better as we increases the number of samples. Compared with the previous results [118], to achieve around 94% classification accuracy,

- harmonic functions on graph samples 120 blogs;
- graph Laplacian regularization samples 120 blogs;
- graph total variation regularization samples 10 blogs; and
- the proposed optimal sampling operator (6.2) samples 2 blogs.

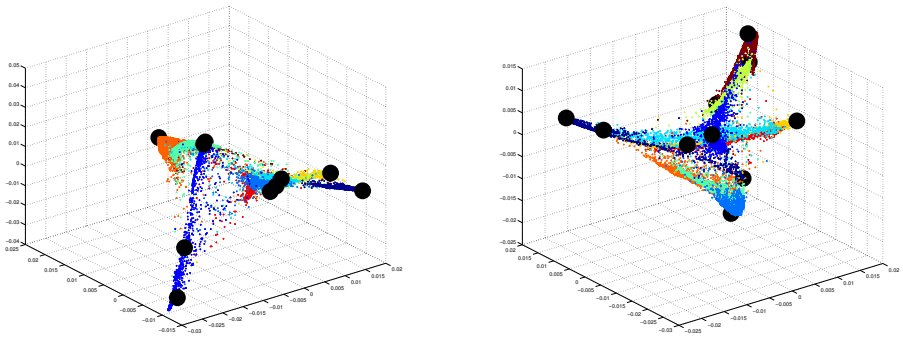
The improvement comes from the fact that, instead of sampling randomly as in [118], we use the optimal sampling operator to choose samples based on the graph structure.

Classification for Handwritten Digits

We aim to use the proposed sampling theory to classify handwritten digits and achieve high classification accuracy with fewer samples.

We work with two handwritten digit datasets, the MNIST [119] and the USPS [120]. Each dataset includes ten classes (0-9 digit characters). The MNIST dataset includes 60,000 samples in total. We randomly select 1000 samples for each digit character, for a total of $N = 10,000$ digit images; each image is normalized to $28 \times 28 = 784$ pixels. The USPS dataset includes 11,000 samples in total. We use all the images in the dataset; each image is normalized to $16 \times 16 = 256$ pixels.

Since same digits produce similar images, it is intuitive to build a graph to reflect the relational dependencies among images. For each dataset, we construct a 12-nearest neighbor graph



(a) MNIST. (b) USPS.

Figure 6.6: Graph representations of the MNIST and USPS datasets. For both datasets, the nodes (digit images) with the same digit characters are shown in the same color and the big black dots indicate 10 sampled nodes by using the optimal sampling operators in Algorithm 2.

to represent the digit images. The nodes represent digit images and each node is connected to 12 other nodes that represent the most similar digit images; the similarity is measured by the Euclidean distance. The graph shift is constructed as $\mathrm{A}_{i,j} = \mathrm{P}_{i,j} \,/\, \sum_i \mathrm{P}_{i,j}$, with

$$\mathrm{P}_{i,j} \;=\; \exp\left(\frac{-N^2 \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2}{\sum_{i,j} \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_2}\right),$$

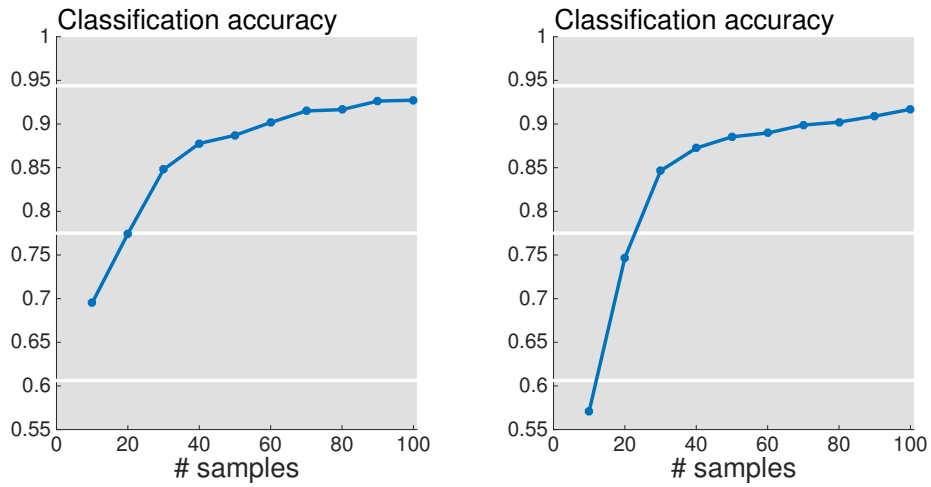
with $\mathbf{f}_i$ a vector representing the digit image. The graph shift is asymmetric, representing a directed graph, which cannot be handled by graph Laplacian-based methods.

Similarly to Section 6.4, we aim to label all the digit images by actively querying the labels of a few images. To handle 10-class classification, we form a ground-truth matrix X of size $N \times 10$. The element $\mathrm{X}_{i,j}$ is $+1$, indicating the membership of the ith image in the jth digit class, and is -1 otherwise. We obtain the optimal sampling operator Ψ as shown in Algorithm 2. The querying samples are then $\mathrm{X}_{\mathcal{M}} = \Psi\,\mathrm{X} \in \mathbb{R}^{M \times 10}$. We recover the low frequency content as

$$\widehat{\mathrm{X}}_{(K)}^{opt} \;=\; \arg \min_{\widehat{\mathrm{X}}_{(K)} \in \mathbb{R}^{K \times 10}} \left\| \mathrm{sgn}(\Psi\,\mathrm{V}_{(K)}\,\widehat{\mathrm{X}}_{(K)}) - \mathrm{X}_{\mathcal{M}} \right\|_2^2. \tag{6.9}$$

We solve (6.9) approximately by using logistic regression and then obtain the estimated label matrix $\mathrm{X}^{opt} = \mathrm{V}_{(K)}\,\widehat{\mathrm{X}}_{(K)}^{opt} \in \mathbb{R}^{N \times 10}$, whose element $(\mathrm{X}^{opt})_{i,j}$ shows a confidence of labeling the ith image as the jth digit. We finally label each digit image by choosing the one with largest value in each row of X^{opt}.

The graph representations of the MNIST and USPS datasets, and the optimal sampling sets are shown in Figure 6.6. The coordinates of nodes come from the corresponding rows of the first three columns of the inverse graph Fourier transform. We see that the images with the same digit characters form clusters, and the optimal sampling operator chooses representative samples from different clusters.



(a) MNIST. (b) USPS.

Figure 6.7: Classification accuracy of the MNIST and USPS datasets as a function of the number of querying samples.

Figure 6.7 shows the classification accuracy by varying the sample size with an interval of 10 from 10 to 100 for both datasets. For the MNIST dataset, we query $0.1\% - 1\%$ images; for the USPS dataset, we query $0.09\% - 0.9\%$ images. We achieve around 90% classification accuracy by querying only 0.5% images for both datasets. Compared with the previous results [61], in the USPS dataset, given 100 samples,

- local linear reconstruction is around 65%;
- normalized cut based active learning is around 70%;
- graph sampling based active semi-supervised learning is around 85%; and
- the proposed optimal sampling operator (6.2) with the interpolation operator (6.9) achieves 91.69%.

6.5 Conclusions

In this chapter, we proposed a novel sampling framework for graph signals that follows the same paradigm as classical sampling theory and strongly connects to linear algebra. We showed that perfect recovery is possible when graph signals are bandlimited. The sampled signal coefficients then form a new graph signal, whose corresponding graph structure is constructed from the original graph structure, preserving the first-order difference of the original graph signal. We studied a qualified sampling operator for both random sampling and experimentally designed sampling. We further established the connection to the sampling theory for finite discrete-time signal processing and previous works on the sampling theory on graphs, and showed how to handle full-band graphs signals by using graph filter banks. We showed applications to semi-supervised classification of online blogs and digit images, where the proposed sampling and interpolation operators perform competitively.

Chapter 7

Sampling of Approximately Bandlimited Graph Signals

7.1 Introduction

Chapter 6 considers sampling bandlimited graph signals under the experimentally designed sampling. It shows that for a noiseless bandlimited graph signal, experimentally designed sampling guarantees perfect recovery while uniform sampling cannot.

In this chapter, we extend this discussion to *approximately bandlimited* graph signals and build a theoretical foundation to understand sampling and recovery of this class under uniform sampling, experimentally designed sampling and active sampling. We propose a recovery strategy to compare uniform sampling and experimentally designed sampling; this recovery strategy is an unbiased estimator for low-frequency components and achieves the optimal rate of convergence under some assumptions on the graph structures. In spirit, our work follows previous work that studied the theoretical capabilities of passive and active sampling for recovering functions from samples [121, 122]; the difference is that we consider a discrete setting and deal with irregular structures. For a smooth function, active sampling, experimentally designed sampling and uniform sampling have the same performance from a statistical perspective [121, 123]. For approximately bandlimited graph signals, however, we will see that while active sampling achieves the same rate of convergence as experimentally designed sampling, experimentally designed sampling fundamentally outperforms uniform sampling when the graph is irregular.

7.2 Problem Formulation

The bandlimited graph signals in Definition 14 is rather restrictive, making it impractical in real-world applications. We thus propose another class of smooth graph signals that relaxes the requirement of bandlimitedness, but still promotes smoothness.

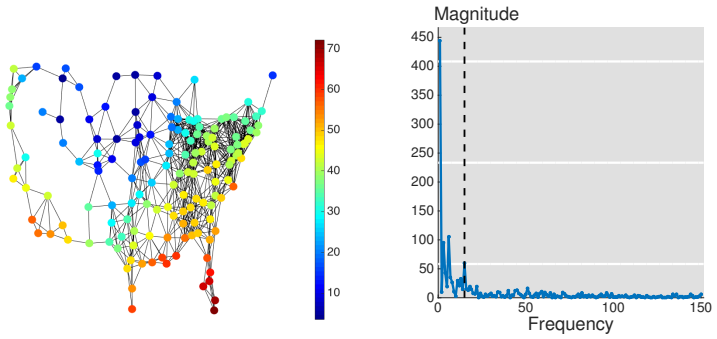
Definition 18. A graph signal $\mathbf{x} \in \mathbb{R}^N$ is *approximately bandlimited* on a graph $\mathrm{A} \in \mathbb{R}^{N \times N}$ with parameters $\beta \geq 1$ and $\mu \geq 0$, when there exists a $K \in \{0, 1, \cdots, N-1\}$ such that the graph

Fourier transform $\widehat{\mathbf{x}}$ satisfies

$$\sum_{k=K}^{N-1}(1+k^{2\beta})\widehat{x}_k^2 \leq \mu \left\|\mathbf{x}\right\|_2^2 . \tag{7.1}$$

Denote this class of graph signals by $\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)$.

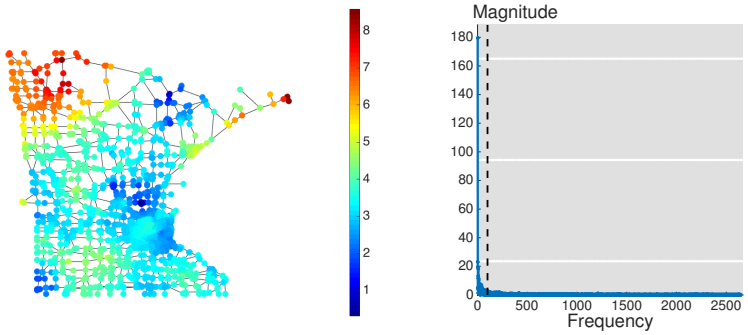
The approximately bandlimited class allows for a tail after the first K frequency components , whose shape and decay are controlled by μ and β; the smaller the μ, the less energy from the high-frequency components is allowed in the tail, and the larger the β, the higher the penalty on the energy from those high-frequency components. The class of $\mathrm{ABL}_{\mathrm{A}}(K)$ is similar to the ellipsoid constraints in previous literature [124], where all the frequency components are considered in the constraints; in other words, $\mathrm{ABL}_{\mathrm{A}}(K)$ poses fewer restrictions on the low-frequency components. Many real graph signals exhibit the approximately bandlimited property; for example, Figures 7.1 and 7.2 show that the temperature readings across the U.S and wind speeds across Minnesota are approximately bandlimited graph signals. We have found that the approximately bandlimited class is more powerful than the bandlimited class when representing real graph signals.



(a) Temperature. (b) Frequency content.

Figure 7.1: Temperature readings across the U.S is an approximately bandlimited graph signal. After the first ten frequency components (black dashed line), energy decays fast.

The goal in this chapter is to study the fundamental limitations of these three sampling strategies when recovering approximately bandlimited graph signals. This study is related to many real-world applications. For example, in semi-supervised learning, datasets are modeled as a graph with data samples as nodes and similarities between those data samples as edges. Features and labels associated with data samples form approximately bandlimited graph signals. We aim to select data samples as labeled data and recover the labels for the unlabeled data. The sampling strategy helps select the most informative data samples and minimizes the recovery error. Some other applications include route planning based on wind speed [99], sensor position selection [125] and compressive spectral clustering [126].



(a) Wind. (b) Frequency content.

Figure 7.2: Wind speed across Minnesota is an approximately bandlimited graph signal. After the first 100 frequency components (black dashed line), energy decays fast.

7.3 Methodology

7.3.1 Fundamental Limits of Sampling Strategies

In this section, we study the fundamental limitations of the three sampling strategies for recovering $\mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$ by showing minimax lower bounds. We do this by following the minimax decision rule and finding tight lower bounds for the minimax risk over all possible recovery strategies [127]. In other words, we try to minimize the recovery error in the worst case and use a tight lower bound to describe this minimax error.

We start by introducing some notation [121].

Definition 19. For a recovery strategy $(\mathbf{x}^*, \mathcal{M})$, and a vector $\mathbf{x} \in \mathbb{R}^N$, the recovery strategy risk is

$$R(\mathbf{x}^*, \mathcal{M}, \mathbf{x}) \;=\; \mathbb{E}_{\mathbf{x},\mathcal{M}}[d^2(\mathbf{x}^*, \mathbf{x})],$$

where $\mathbb{E}_{\mathbf{x},\mathcal{M}}$ is the expectation with respect to the probability measure of $\{x_i, y_i\}_{i\in\mathcal{M}}$ and $d(\mathbf{x}^*, \mathbf{x})$ is the error metric. Here we use the ℓ_2 norm $\|\mathbf{x}^* - \mathbf{x}\|_2$. The maximum risk of a recovery strategy is $\sup_{\mathbf{x}\in\mathbb{R}^N} R(\mathbf{x}^*, \mathcal{M}, \mathbf{x})$.

The lower bounds we present will be of the form

$$\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta} \sup_{\mathbf{x}\in\mathbb{R}^N} \mathbb{E}_{\mathbf{x},\mathcal{M}}[d^2(\mathbf{x}^*, \mathbf{x})] \geq c\phi_m^2, \text{ for all } m \geq m_0, \tag{7.2}$$

where $\inf$ is the infimum (greatest lower bound) and $\sup$ is the supremum (least upper bound), m is the number of samples, $m_0 \in \mathbb{N}$, $c > 0$ is a constant, ϕ_m is a positive sequence converging to zero, and Θ is the set of all recovery strategies. The sequence ϕ_m^2 is denoted as a lower rate of convergence.

The upper bounds on the maximum risk are usually obtained through explicit recovery strategies, as will be shown in Section 7.3.2. The upper bounds we present will be of the form

$$\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta} \sup_{\mathbf{x}\in\mathbb{R}^N} \mathbb{E}_{\mathbf{x},\mathcal{M}}[d^2(\mathbf{x}^*, \mathbf{x})] \leq C\phi_m^2, \text{ for all } m \geq m_0, \tag{7.3}$$

where $C > 0$ is a constant. If (7.2) and (7.3) both hold, then ϕ_m is said to be the optimal rate of convergence and there is no recovery strategy that asymptotically outperforms the proposed one. When talking about optimal rates of convergence, we bound the sequence by a polynomial and make statements of the form: Given $\gamma_1 < \gamma < \gamma_2$, a rate of convergence ϕ_m^2 is equivalent to $m^{-\gamma}$ if and only if $m^{-\gamma_2} < \phi_m^2 < m^{-\gamma_1}$ for n large enough.

Note that the general bounds are for arbitrary graphs and thus involve parameters that depend on the graph structure; given the graph structure, we can specify the parameters and show explicit rates of convergence, as we will do in Section 7.3.6.

Let $\mathrm{V}_{(2,K)}$ be the sub-matrix of V, consisting of the Kth to the $(2K-1)$th columns of V.

Theorem 14. Given the class $\mathrm{ABL_A}(K, \beta, \mu)$:
(1) Under uniform sampling,

$$\begin{aligned}
&\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{u}}} \sup_{\mathbf{x}\in\mathrm{ABL_A}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \right) \\
\geq\ & \max_{K\leq\kappa_0\leq N} \frac{c_1\mu}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu\,\|\mathbf{x}\|_2^2}{\sigma^2\kappa_0^{2\beta+2}N} \left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2 m \right),
\end{aligned}$$

where $c_1 > 0$, $0 < c < 1$, and Θ_{u} denotes the set of all recovery strategies based on uniform sampling;
(2) Under experimentally designed sampling,

$$\begin{aligned}
&\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}} \sup_{\mathbf{x}\in\mathrm{ABL_A}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \right) \\
\geq\ & \max_{K\leq\kappa_0\leq N} \frac{c_1\mu}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu\,\|\mathbf{x}\|_2^2}{\sigma^2\kappa_0^{2\beta+2}} \left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2 m \right),
\end{aligned}$$

where $c_1 > 0$, $0 < c < 1$, and Θ_{e} denotes the set of all recovery strategies based on experimentally designed sampling;
(3) Under active sampling,

$$\begin{aligned}
&\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{a}}} \sup_{\mathbf{x}\in\mathrm{ABL_A}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \right) \\
\geq\ & \max_{K\leq\kappa_0\leq N} \frac{c_1\mu}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu\,\|\mathbf{x}\|_2^2}{\sigma^2\kappa_0^{2\beta+2}} \left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2 m \right),
\end{aligned}$$

where $c_1 > 0$, $0 < c < 1$, and Θ_{a} denotes the set of all recovery strategies based on active sampling.

See Appendix 14.1 for the proof of these results. From Theorem 14, we see that experimentally designed sampling has the same minimax lower bound as active sampling, which means that collecting the feedback before taking samples does not fundamentally improve the recovery performance. We also see that the three minimax lower bounds depend on the properties of $\mathrm{V}_{(2,\kappa_0)}$, which depend on the graph structure. When each row of $\mathrm{V}_{(2,\kappa_0)}$ has roughly similar

energy, $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2$ and $N\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2$ are similar; when the energy is concentrated in a few rows, $N\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2$ is much larger than $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2$; in other words, the minimax lower bound for experimentally designed sampling is tighter than that for uniform sampling. This happens in many real-world graphs that have complex, irregular structure. The minimax lower bounds thus show the potential advantage of experimentally designed sampling and active sampling over uniform sampling. We will elaborate on this in Sections 7.3.6 and 7.4.

7.3.2 Recovery Strategy

In the previous section, we presented the minimax lower bounds for each of the three sampling strategies and showed that active sampling cannot fundamentally perform better than experimentally designed sampling. We now propose a recovery strategy for both uniform sampling and experimentally designed sampling and evaluate its statistical properties.

7.3.3 Algorithm

To analyze uniform sampling and experimentally designed sampling in a similar manner, we consider *sampling score-based sampling* that unifies both sampling strategies. Sampling score-based sampling means that the sample indices are chosen from an importance sampling distribution that is proportional to some sampling score. Let $\{\pi_i\}_{i=1}^N$ be a set of sampling scores, where π_i denotes the probability to choose the ith sample in each random trial. When the sampling score for each node is the same, we get uniform sampling; when the sampling score for each node is designed based on the graph structure, we get experimentally designed sampling.

For $\mathrm{ABL}_{\mathrm{A}}(K)$, most of the energy is concentrated in the first K frequency components and the graph signal can be approximately recovered by using those. We consider the following recovery strategy to estimate those frequency components by projecting samples onto the bandlimited space spanned by the first K frequency components.

Algorithm 1. We sample a graph signal m times. Each time, we independently choose a node $\mathcal{M}_j = i$, $j = 1, \ldots, m$, with probability π_i, and take a measurement $y_{\mathcal{M}_j}$. Let $\Psi \in \mathbb{R}^{m \times N}$ be the sampling operator, $\mathrm{V}_{(K)} \in \mathbb{R}^{N \times K}$ be the first K columns of V, and $\mathrm{D} \in \mathbb{R}^{N \times N}$ be a diagonal rescaling matrix with $\mathrm{D}_{i,i} = 1/\sqrt{m\pi_i}$. We recover the original graph signal by solving the following optimization problem:

$$\mathbf{x}_{\mathrm{SP}}^* = \mathrm{V}_{(K)} \arg\min_{\widehat{\mathbf{x}}_{(K)}} \left\|\Psi^T \Psi\, \mathrm{D}^2\, \Psi^T \mathbf{y} - \mathrm{V}_{(K)}\, \widehat{\mathbf{x}}_{(K)}\right\|_2^2 \tag{7.4}$$

$$= \mathrm{V}_{(K)}\, \mathrm{U}_{(K)}\, \Psi^T \Psi\, \mathrm{D}^2\, \Psi^T \mathbf{y}, \tag{7.5}$$

where $\mathbf{y}$ is a vector representation of the samples (11.2).

We call Algorithm 1 *sampled projection* (SampleProj) because we reweigh and project the samples onto the bandlimited space. $\Psi^T \mathbf{y} \in \mathbb{R}^N$ rescales the sampled signal $\mathbf{y} \in \mathbb{R}^m$ through zero padding. The rescaling matrix D^2 compensates for non-uniform weights in sampling and equalizes samples from different sampling scores. The term $\Psi^T \Psi\, \mathrm{D}^2\, \Psi^T \mathbf{y}$ denotes the equalized

and zero-padding samples. The objective function (7.4) minimizes the distance between our estimate and the samples. The intuition behind the solution (7.5) is that we project the equalized and zero-padding samples onto a bandlimited space spanned by $\mathrm{V}_{(K)}$ to remove aliasing. The term $\mathrm{U}_{(K)}\,\Psi^T\Psi\,\mathrm{D}^2\,\Psi^T\mathbf{y}$ is an unbiased estimator for the first K frequency components, which will be shown later. Thus, in expectation, the recovered graph signal $\mathbf{x}^*_{\mathrm{SP}}$ is a linear approximation of any graph signal $\mathbf{x}$. When $\mathbf{x}$ is bandlimited, $\mathbf{x}^*_{\mathrm{SP}}$ perfectly recovers $\mathbf{x}$; when $\mathbf{x}$ is approximately bandlimited, $\mathbf{x}^*_{\mathrm{SP}}$ has a bias due to the existence of high-frequency components.

It is also intuitive to consider the following recovery strategy [128],

$$\begin{aligned}\mathbf{x}^*_{\mathrm{LS}} &= \mathrm{V}_{(K)}\arg\min_{\widehat{\mathbf{x}}_{(K)}}\left\|\mathrm{D}\,\Psi^T\mathbf{y}-\mathrm{D}\,\Psi^T\Psi\,\mathrm{V}_{(K)}\,\widehat{\mathbf{x}}_{(K)}\right\|_2^2\\ &= \mathrm{V}_{(K)}\left(\mathrm{D}^2\,\Psi^T\Psi\,\mathrm{V}_{(K)}\right)^{\dagger}\mathrm{D}\,\Psi^T\mathbf{y},\end{aligned}$$

where we fit the sampled elements of the recovered graph signal to the samples by solving the least squares problem and $(\cdot)^{\dagger}$ is the pseudo-inverse operator [84]. The corresponding expected recovered graph signal is

$$\mathbb{E}\mathbf{x}^*_{\mathrm{LS}} = \mathrm{V}_{(K)}\,\widehat{\mathbf{x}}_{(K)} + \mathrm{V}_{(K)}\,\mathrm{P}\,\mathrm{V}_{(-K)}\,\widehat{\mathbf{x}}_{(-K)},$$

where $\mathrm{P} = \left(\mathrm{D}^2\,\Psi^T\Psi\,\mathrm{V}_{(K)}\right)^{\dagger}\mathrm{D}\,\Psi^T\Psi = \left(\mathrm{U}_{(K)}\,\Psi^T\Psi\,\mathrm{D}^2\,\Psi^T\Psi\,\mathrm{V}_{(K)}\right)^{-1}\mathrm{U}_{(K)}\,\Psi^T\Psi\,\mathrm{D}^2\,\Psi^T\Psi$, $\mathrm{V}_{(-K)}$ chooses the last K columns of V, and $\widehat{\mathbf{x}}_{(-K)}$ chooses the last K columns of $\mathbf{x}$. When $\mathbf{x}$ is bandlimited, $\mathbf{x}^*_{\mathrm{LS}}$ perfectly recovers $\mathbf{x}$; when $\mathbf{x}$ is approximately bandlimited, however, $\mathbf{x}^*_{\mathrm{LS}}$ is a mixture of low-frequency and high-frequency components and it is hard to show its statistical properties. Moreover, it is less computationally efficient to compute $\mathbf{x}^*_{\mathrm{LS}}$ than $\mathbf{x}^*_{\mathrm{SP}}$ because of the presence of the inverse term.

Since the definitions of the bandlimited class and the approximately bandlimited class, and the proposed sampling strategies are all based on the graph Fourier transform instead of on graph shift, all the proposed methods work for other versions of the graph Fourier transform as well.

7.3.4 Statistical Analysis

We study the statistical properties of Algorithm 1 by providing the bias, covariance, mean square error (MSE), and an optimal sampling distribution.

The following lemma shows that the sampled projection estimator is an unbiased estimator of the first K frequency components for any sampling scores.

Lemma 1. The sampled projection estimator with bandwidth K and arbitrary sampling scores is an unbiased estimator of the first K frequency components, that is,

$$\mathbb{E}\mathbf{x}^* \;=\; \mathrm{V}_{(K)}\,\mathrm{U}_{(K)}\,\mathbf{x}, \quad \text{for all } \mathbf{x},$$

where $\mathbf{x}^*$ is the solution of Algorithm 1.

Lemma 2 gives the exact covariance of the sampled projection estimator.

Lemma 2. The covariance of sampled projection estimator $\mathbf{x}^*$ has the following property:

$$\begin{aligned}\mathrm{Tr}(\mathrm{Covar}\,[\mathbf{x}^*]) &= \mathbb{E}\left\|\mathbf{x}^*-\mathbb{E}\,[\mathbf{x}^*]\right\|_2^2\\ &= \mathrm{Tr}\left(\mathrm{U}_{(K)}\,\mathrm{W}_C\,\mathrm{V}_{(K)}\right)-\frac{1}{m}\left\|\widehat{\mathbf{x}}_{(K)}\right\|_2^2,\end{aligned}$$

where Tr(·) is the trace operator and W_C is a diagonal matrix with $(\mathrm{W}_C)_{i,i} = (x_i^2 + \sigma^2)/(m\pi_i)$.

Theorem 15 shows the exact MSE of sampled projection estimator and an upper bound.

Theorem 15. For $\mathbf{x} \in \mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$, let $\mathbf{x}^*$ be the sampled projection estimator with bandwidth $\kappa \geq K$. Then,

$$\begin{aligned} & \mathbb{E} \left\| \mathbf{x}^* - \mathbf{x} \right\|_2^2 \\ = & \left\| \mathrm{V}_{(-\kappa)} \widehat{\mathbf{x}}_{(-\kappa)} \right\|_2^2 + \mathrm{Tr} \left(\mathrm{U}_{(\kappa)} \mathrm{W}_C \mathrm{V}_{(\kappa)} \right) - \frac{1}{m} \left\| \widehat{\mathbf{x}}_{(\kappa)} \right\|_2^2 \\ \leq & \frac{\mu}{1 + \kappa^{2\beta}} \left\| \mathbf{x} \right\|_2^2 + \mathrm{Tr} \left(\mathrm{U}_{(\kappa)} \mathrm{W}_C \mathrm{V}_{(\kappa)} \right). \end{aligned} \tag{7.6}$$

We merge the proofs of Lemmas 7, 2 and Theorem 15 in Appendix 14.2. The main idea follows from the bias-variance tradeoff. The bias term $\mu/(1 + \kappa^{2\beta}) \left\| \mathbf{x} \right\|_2^2$ comes from the high-frequency components and $\mathrm{Tr} \left(\mathrm{U}_{(\kappa)} \mathrm{W}_C \mathrm{V}_{(\kappa)} \right)$ comes from the covariance. The last inequality in Theorem 15 comes from relaxing the bias term and omitting a constant, which has nothing to do with the sampling scores. Thus, optimizing the upper bound over sampling scores is equivalent to optimizing the exact MSE.

We next study the MSEs of sampled projection estimator based on both uniform sampling and experimentally designed sampling.

Corollary 3. The upper bound of MSE of uniform sampling is

$$\mathbb{E} \left\| \mathbf{x}^* - \mathbf{x} \right\|_2^2 \leq \frac{\mu}{1 + \kappa^{2\beta}} \left\| \mathbf{x} \right\|_2^2 + \frac{N}{m} \sum_{k=1}^{\kappa} \sum_{i=1}^{N} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2 \right).$$

The upper bound just specifies that the sampling score be $\pi_i = 1/N$ for all i. For experimentally designed sampling, we are allowed to study the graph structure and design sample indices. In the following corollary, we show a set of optimal sampling scores of the sampled projection estimator by minimizing the upper bound of MSE.

Corollary 4. The optimal sampling score of the sampled projection estimator with bandwidth $\kappa \geq K$ is

$$\pi_i \propto \sqrt{\left(\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \right) (x_i^2 + \sigma^2)}.$$

The corresponding upper bound of MSE is

$$\begin{aligned} & \mathbb{E} \left\| \mathbf{x}^* - \mathbb{E}\mathbf{x}^* \right\|_2^2 \\ \leq & \frac{\mu}{1 + \kappa^{2\beta}} \left\| \mathbf{x} \right\|_2^2 + \frac{1}{m} \left(\sum_{i=1}^{N} \sqrt{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2 \right)} \right)^2. \end{aligned}$$

Proof. To obtain the optimal sampling score for the estimator, we minimize the MSE given in Theorem 15 and solve the following optimization problem.

$$\begin{aligned} & \min_{\pi_i} \mathrm{Tr} \left(\mathrm{U}_{(\kappa)} \mathrm{W}_C \mathrm{V}_{(\kappa)} \right), \\ & \text{subject to } \sum_i \pi_i = 1, \pi_i \geq 0. \end{aligned}$$

The objective function is the variance term of the MSE derived in Theorem 15. Since the bias term has nothing to do with the sampling score, minimizing the variance term is equivalent to minimizing the MSE. The constraints require $\{\pi_i\}_{i=1}^N$ to be a valid probability distribution. The Lagrangian function is then

$$\begin{aligned} L(\pi_i, \eta_0, \eta_i) &= \sum_{i=1}^{N} \left(\frac{x_i^2 + \sigma^2}{m\pi_i} \sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \right) \\ &\quad + \eta_0 \left(\sum_{i=1}^{N} \pi_i - 1 \right) + \sum_{i=1}^{N} \eta_i \pi_i, \end{aligned}$$

where η_0, η_i are Lagrangian multipliers. We set the derivative of the Lagrangian function to zero,

$$\frac{dL}{d\pi_i} = -\frac{x_i^2 + \sigma^2}{m\pi_l^2} \sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 + \eta_0 + \eta_i = 0,$$

and obtain the optimal sampling score

$$\pi_i \propto \sqrt{\left(\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \right) (x_i^2 + \sigma^2)}. \tag{7.7}$$

Substituting the optimal sampling score π_i to the upper bound of the MSE (7.6),

$$\begin{aligned} &\mathbb{E} \left\| \mathbf{x}^* - \mathbb{E}\mathbf{x}^* \right\|_2^2 \overset{(a)}{\leq} \frac{\mu}{1 + \kappa^{2\beta}} \left\| \mathbf{x} \right\|_2^2 \\ + &\frac{1}{m} \sum_{i=1}^{N} \frac{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 (x_i^2 + \sigma^2)}{\sqrt{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2 \right)}} \sum_{i=1}^{N} \sqrt{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2 \right)} \\ = &\frac{\mu}{1 + \kappa^{2\beta}} \left\| \mathbf{x} \right\|_2^2 + \frac{1}{m} \left(\sum_{i=1}^{N} \sqrt{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2 \right)} \right)^2, \end{aligned}$$

where (a) follows from substituting the optimal sampling score (7.7) into the upper bound of the MSE (7.6). □

We see that the optimal sampling score includes a trade-off between signal and noise. In practice, we cannot access $\mathbf{x}$ and thus need to approximate the ratio between each x_i and σ^2. For active sampling, we can collect feedback to approximate each signal coefficient x_i; for experimentally designed sampling, we approximate beforehand; one way is to use the graph structure to sketch the shape of $\mathbf{x}$. We first use the Cauchy-Schwarz inequality to bound x_i,

$$|x_i| = |\mathbf{v}_i^T \widehat{\mathbf{x}}| \leq \left\| \mathbf{v}_i \right\|_2 \left\| \widehat{\mathbf{x}} \right\|_2 = \left\| \mathbf{v}_i \right\|_2 \left\| \mathbf{x} \right\|_2,$$

where $\mathbf{v}_i$ is the ith row of V. Thus, in the upper bound, we have a tradeoff between signal and noise: when the signal-to-noise ratio (SNR) $\|\mathbf{x}\|_2/\sigma^2$ is small, the approximate optimal sampling score is

$$\pi_i \propto \left\|\mathbf{v}_{i,(K)}\right\|_2 = \sqrt{\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2},$$

which is the square root of the leverage score of $\mathrm{V}_{(K)}$.

When the SNR $\|\mathbf{x}\|_2/\sigma^2$ is large, the approximate optimal sampling score is

$$\pi_i \propto \sqrt{\left(\sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2\right) \|\mathbf{v}_i\|_2^2}.$$

For approximately bandlimited signals, when β is large and μ is small, the main energy concentrates in the first K frequency components, $\|\mathbf{v}_i\|_2$ is concentrated in $\left\|\mathbf{v}_{i,(K)}\right\|_2$, where $\mathbf{v}_{i,(K)}$ is the first K elements in $\mathbf{v}_i$. Specifically,

$$\begin{aligned}
|x_i| &= |\mathbf{v}_i^T \widehat{\mathbf{x}}| = |\mathbf{v}_{i,(K)}^T \widehat{\mathbf{x}}_{(K)} + \mathbf{v}_{i,(-K)}^T \widehat{\mathbf{x}}_{(-K)}| \\
&\leq \left\|\mathbf{v}_{i,(K)}\right\|_2 \left\|\widehat{\mathbf{x}}_{(K)}\right\|_2 + \left\|\mathbf{v}_{i,(-K)}\right\|_2 \left\|\widehat{\mathbf{x}}_{(-K)}\right\|_2 \\
&\leq \left\|\mathbf{v}_{i,(K)}\right\|_2 \left\|\widehat{\mathbf{x}}_{(K)}\right\|_2 + \left\|\mathbf{v}_{i,(-K)}\right\|_2 \sqrt{\frac{\mu}{1+K^{2\beta}}} \|\mathbf{x}\|_2 \\
&\approx \left\|\mathbf{v}_{i,(K)}\right\|_2 \left\|\widehat{\mathbf{x}}_{(K)}\right\|_2 .
\end{aligned}$$

In this case, when the SNR $\|\mathbf{x}\|_2/\sigma^2$ is large, the approximate optimal sampling score is

$$\pi_i \propto \left\|\mathbf{v}_{i,(K)}\right\|_2^2 = \sum_{k=1}^{\kappa} \mathrm{U}_{k,i}^2,$$

which is the leverage score of $\mathrm{V}_{(K)}$.

7.3.5 Relation to the Sampling Theory on Graphs

Sampling theory on graphs in Chapter 6 considers a bandlimited graph signal under the experimentally designed sampling. It shows that for a noiseless bandlimited graph signal, experimentally designed sampling guarantees perfect recovery while uniform sampling cannot, which also implies that active sampling cannot perform better than experimentally designed sampling. The recovery strategy is to solve the following optimization problem,

$$\mathbf{x}_{\mathrm{ST}}^* = \arg \min_{\mathbf{x} \in \mathrm{BL}_{\mathrm{A}}(K)} \|\Psi \mathbf{x} - \mathbf{y}\|_2^2 = \mathrm{V}_{(K)} (\Psi \mathrm{V}_{(K)})^{\dagger} \mathbf{y}, \tag{7.8}$$

where Ψ is the sampling operator (5.1) and $\mathbf{y}$ is a vector representation of the samples (11.2). When the original graph signal is bandlimited, the estimator (7.8) is unbiased and its MSE comes

solely from the variance term caused by noise, that is,

$$\begin{aligned} & \mathbb{E}\,\|\mathbf{x}^*_{\mathrm{ST}} - \mathbf{x}\|_2^2 \\ = \;& \mathbb{E}\,\left\|\mathrm{V}_{(K)}(\Psi\,\mathrm{V}_{(K)})^\dagger(\Psi\mathbf{x}+\epsilon) - \mathbf{x}\right\|_2^2 \\ = \;& \mathbb{E}\,\left\|\mathrm{V}_{(K)}(\Psi\,\mathrm{V}_{(K)})^\dagger\epsilon\right\|_2^2 \;=\; \mathbb{E}\,\left\|(\Psi\,\mathrm{V}_{(K)})^\dagger\epsilon\right\|_2^2. \end{aligned}$$

In [90], the authors propose an optimal sampling operator based on minimizing $\left\|(\Psi\,\mathrm{V}_{(k)})^\dagger\right\|_2^2$. The sample set is deterministic and the procedure is computationally efficient when the sample size is small. Compared with sampling score-based sampling, however, the optimal sampling operator is computationally inefficient when the sample size is large. When the original graph signal is not bandlimited, similarly to $\mathbf{x}^*_{\mathrm{LS}}$, the solution of (7.8) is biased, that is,

$$\begin{aligned} \mathbb{E}\mathbf{x}^* &= \mathrm{V}_{(K)}(\Psi\,\mathrm{V}_{(K)})^\dagger\mathbb{E}(\Psi\mathbf{x}+\epsilon) \\ &= \mathrm{V}_{(K)}(\Psi\,\mathrm{V}_{(K)})^\dagger\Psi\left(\mathrm{V}_{(K)}\,\widehat{\mathbf{x}}_{(K)} + \mathrm{V}_{(-K)}\,\widehat{\mathbf{x}}_{(-K)}\right) \\ &= \mathrm{V}_{(K)}\,\widehat{\mathbf{x}}_{(K)} + \mathrm{V}_{(K)}(\Psi\,\mathrm{V}_{(K)})^\dagger\,\mathrm{V}_{(-K)}\,\widehat{\mathbf{x}}_{(-K)}. \end{aligned}$$

We see that the high-frequency components $\mathrm{V}_{(-K)}\,\widehat{\mathbf{x}}_{(-K)}$ are projected on the low-frequency space spanned by $\mathrm{V}_{(K)}$, which causes aliasing.

7.3.6 Optimal Rates of Convergence

In this section, we introduce two types of graphs and show that the proposed recovery strategies achieve the optimal rates of convergence on both. Type-1 graphs model regular graphs, where the corresponding graph Fourier bases are not sparse and elements have similar magnitudes; examples are circulant graphs and nearest-neighbor graphs [129]. Since the energy is evenly spread over the graph Fourier basis, each element contains similar amount of information; for such graphs, experimentally designed sampling performs similarly to uniform sampling. Type-2 graphs model irregular graphs, where the corresponding graph Fourier bases are sparse and elements do not have similar magnitudes; examples are small-world graphs and scale-free graphs [26]. Since the energy is concentrated in a few elements in the graph Fourier basis, these elements are more informative and should be selected. For such graphs, experimentally designed sampling outperforms uniform sampling.

7.3.7 Type-1 Graphs

Definition 20. A graph $\mathrm{A} \in \mathbb{R}^{N\times N}$ is of *Type-1*, when its graph Fourier basis satisfies

$$|\,\mathrm{V}_{i,j}\,| = O(N^{-1/2}), \quad \text{for all } i,j = 0,1,\cdots,N-1.$$

For a Type-1 graph, elements in V have roughly similar magnitudes, that is, the energy is evenly spread over V.

Based on Theorem 15, we can specify the parameters for a Type-1 graph and show the following results.

Corollary 5. Let $\mathrm{A} \in \mathbb{R}^{N\times N}$ be a Type-1 graph, for the class $\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)$.

- Let $\mathbf{x}^*$ be the sampled projection estimator with the bandwidth $\kappa \geq K$ and uniform sampling; we have

$$\mathbb{E}\left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right) \leq C\, m^{-\frac{2\beta}{2\beta+1}},$$

  where C is a positive constant, m is the number of samples (5.1), β is the spectral decay factor in the approximately bandlimited class (7.1), and the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N;
- Let $\mathbf{x}^*$ be the sampled projection estimator with the bandwidth $\kappa \geq K$ and the optimal sampling score in Corollary 4; we have

$$\mathbb{E}\left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right) \leq C\, m^{-\frac{2\beta}{2\beta+1}},$$

  where C is a positive constant, and the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N.

When $m \gg N$, we set $\kappa = N$, the bias term is then zero, and both upper bounds are actually $C\, m^{-1}$. We see that uniform sampling and optimal sampling score based sampling have the same convergence rate, that is, experimentally designed sampling does not perform better than uniform sampling for the Type-1 graphs.

Based on Theorem 14 and Corollary 5, we conclude the following.

Corollary 6. Let $\mathrm{A} \in \mathbb{R}^{N\times N}$ be a Type-1 graph, for the class $\mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$.

- Under uniform sampling,

$$\begin{aligned} c\, m^{-\frac{2\beta}{2\beta+1}} \leq & \\ \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{u}}} \sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} & \mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\|\mathbf{x}^* - \mathbf{x}\|_2^2\right) \\ & \leq C\, m^{-\frac{2\beta}{2\beta+1}}, \end{aligned}$$

  where constant $C > c > 0$, and the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N;
- Under experimentally designed sampling,

$$\begin{aligned} c\, m^{-\frac{2\beta}{2\beta+1}} \leq & \\ \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}} \sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} & \mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\|\mathbf{x}^* - \mathbf{x}\|_2^2\right) \\ & \leq C\, m^{-\frac{2\beta}{2\beta+1}}, \end{aligned}$$

  where constant $C > c > 0$, and the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N.

We merge the proofs of Corollaries 5 and 6 in Appendix 14.3.

We see that under both random and experimentally designed sampling, the lower and upper bounds have the same rate of convergence, which achieves the optimum. In addition, random and

experimentally designed sampling have the same optimal rate of convergence and we can thus conclude that experimentally designed sampling does not perform asymptotically better than uniform sampling for the Type-1 graphs. The sampled projection estimator attains the optimal rate of convergence.

7.3.8 Type-2 Graphs

Definition 21. A graph $\mathrm{A} \in \mathbb{R}^{N\times N}$ is of *Type-2*, when its graph Fourier basis satisfies

$$|\mathbf{v}_{i,T}| = O(1), \text{and } |\mathbf{v}_{i,T+1}| \ll O(1), \text{ for all } i = 0, \cdots, N-1,$$

where $\mathbf{v}_{i,T}$ is the Tth largest elements in the ith column of V and $T \ll N$ is some constant.

A Type-2 graph requires that each column vector of V be approximately sparse. When we take a few columns from V to form a submatrix, the energy in the submatrix concentrates in a few rows of the submatrix. This is equivalent to the sampling scores being approximately sparse. Simulations show that star graphs, scale-free graphs and small-world graphs approximately fall into this type of graphs.

Based on Theorem 15, we can specify the parameters for a Type-2 graph and show the following results.

Corollary 7. Let $\mathrm{A} \in \mathbb{R}^{N\times N}$ be a Type-2 graph, for the class $\mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$.

- Let $\mathbf{x}^*$ be the sampled projection estimator with the bandwidth $\kappa \geq K$ and uniform sampling; we have

$$\mathbb{E}\left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right) \leq C\, m^{-\frac{2\beta}{2\beta+\gamma}},$$

  where C is a positive constant, and the rate is achieved when κ is of the order of $m^{1/(2\beta+\gamma)}$, $\gamma = \log(N)/\log(\kappa) \geq 1$;
- Let $\mathbf{x}^*$ be the sampled projection estimator with the bandwidth $\kappa \geq K$ and optimal sampling score based sampling; we have

$$\mathbb{E}\left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right) \leq C\, m^{-\frac{2\beta}{2\beta+1}},$$

  where C is a positive constant, the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N.

Based on Theorem 14 and Corollary 7, we conclude the following.

Corollary 8. Let $\mathrm{A} \in \mathbb{R}^{N\times N}$ be a Type-2 graph with parameter K_0, for the class $\mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$.

- Under uniform sampling,

$$c\, m^{-\frac{2\beta}{2\beta+1}} \leq \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{u}}} \sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}\right) \leq C\, m^{-\frac{2\beta}{2\beta+\gamma}},$$

where constant $C > c > 0$, and the rate is achieved when κ is of the order of $m^{1/(2\beta+\gamma)}$ and $\gamma = \log(N)/\log(\kappa) \geq 1$;

- Under experimentally designed sampling, there exists a $\gamma > 1$,

$$c\, m^{-\frac{2\beta}{2\beta+1}} \leq \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}} \sup_{\mathbf{x}\in \mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\frac{\|\mathbf{x}^* - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \right) \leq C\, m^{-\frac{2\beta}{2\beta+1}},$$

where C is a positive constant, the rate is achieved when κ is of the order of $m^{1/(2\beta+1)}$ and upper bounded by N.

We merge the proofs of Corollaries 7 and 8 in Appendix 14.4.

We see that under both uniform and experimentally designed sampling, the lower and upper bounds have the same rate of convergence, which achieves the optimum. However, experimentally designed sampling has a larger optimal rate of convergence, and we can thus conclude that experimentally designed sampling exhibits asymptotically better performance than uniform sampling for a Type-2 graph. The sampled projection estimator attains the optimal rate of convergence.

7.4 Experimental Results

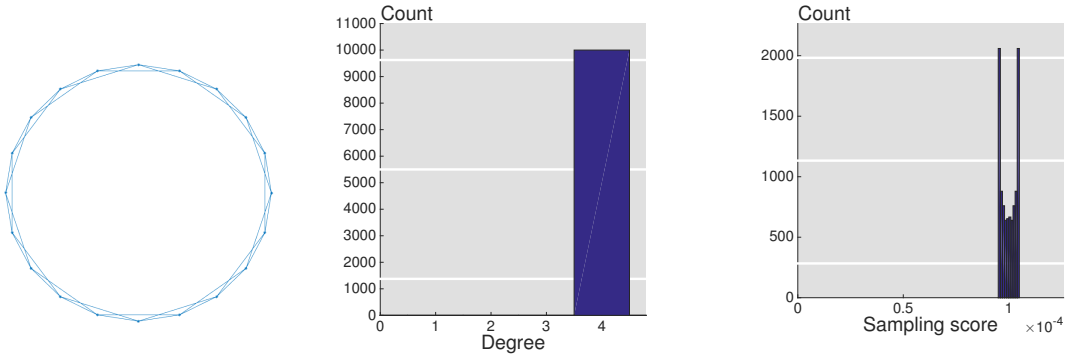
In this section, we validate the proposed recovery strategy on five specific graphs: a ring graph, an Erdős-Rényi graph, a random geometric graph, a small-world graph and a power-law graph. Based on the graph structure, we roughly label each as a Type-1 or Type-2, and then, for each, we compare the empirical performance of the proposed recovery strategy based on uniform and experimentally designed sampling. For experimentally designed sampling, we use both the leverage score of $\mathrm{V}_{(K)}$ (approximately optimal in the noiseless case) and the square root of the leverage score of $\mathrm{V}_{(K)}$ (approximately optimal in the noisy case). In the experiments, the graph Fourier basis is the eigenvector matrix of the adjacency matrix. We find similar results when the graph Fourier basis is the eigenvector matrix of the graph Laplacian matrix.

7.4.1 Simulated Graphs

We now introduce the five graphs, each with $10,000$ nodes, used in our experiments.

Ring graph with k-nearest neighbors. A ring graph is a graph where each node connects to its k-nearest neighbors. We use a ring graph where each node connects to exactly four nearest neighbors. The eigenvectors are similar to the discrete cosine transform and the energy evenly spreads to each element in V [129] ; this is thus a Type-1 graph. Figure 7.3 illustrates some properties of the ring graph: Figure 7.3(a) shows the graph plot (for easier visualization, only 20 nodes are shown and with enough zoom, one can clearly see that each node connects to exactly four nearest neighbors; Figure 7.3(b) shows the histogram of the degrees that concentrate on 4, as expected; and Figure 7.3(c) shows the histogram of the leverage scores of $\mathrm{V}_{(20)}$, which are the

optimal sampling scores when the SNR is large. Note that we set the bandwidth $K = 20$ to show the low-frequency band of the graph Fourier transform matrix. We see that the leverage scores concentrate around 10^{-4}; this means that each node has the same probability to be chosen and uniform sampling is approximately the optimal sampling strategy.

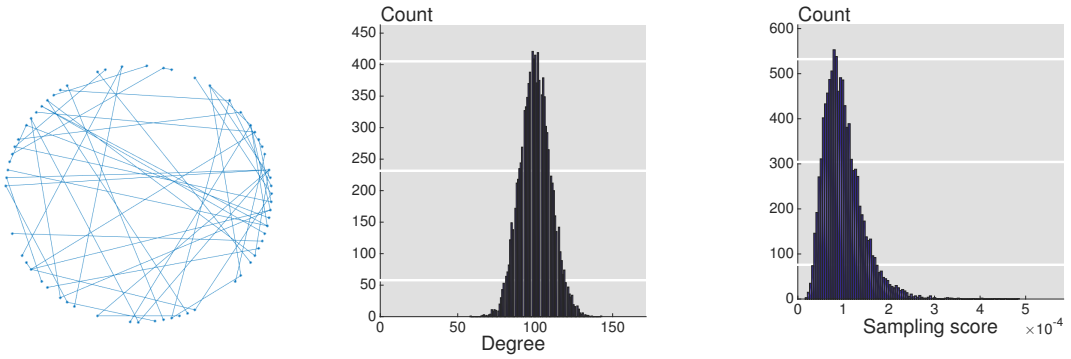


(a) Graph plot. (b) Histogram of the degrees. (c) Histogram of the leverage scores.

Figure 7.3: Properties of a ring graph. Plot (c) shows the histogram of the leverage scores, which are optimal sampling scores when the SNR is large.

Erdős-Rényi graph. An Erdős-Rényi graph is a random graph where each pair of nodes is connected with some probability [26]. We use a graph where each pair of nodes is connected with probability of 0.01, that is, each node has 100 neighbors in expectation. Figure 7.4 illustrates some properties of the Erdős-Rényi graph. Figure 7.4(a) shows the graph plot (for easier visualization, only 100 nodes are shown); Figure 7.4(b) shows the histogram of the degrees that concentrate around 100, as expected; and Figure 7.4(c) shows the histogram of the leverage scores of $\mathrm{V}_{(20)}$, which are the optimal sampling scores when the SNR is large. We see that the leverage scores concentrate around 10^{-4}; this means that each node has the same probability to be chosen and uniform sampling is approximately the optimal sampling strategy, similarly to the ring graph. Based on the above, an Erdős-Rényi graph is approximately a Type-1 graph.

Random geometric graph. A random geometric graph is a spatial graph where each of the nodes is assigned random coordinates in the box $[0, 1]^d$ and an edge appears when the distance between two nodes is in a given range [130]. We used a graph lying in the box $[0, 1]^2$, and two nodes are connected when the Euclidean distance is less than 0.03. Figure 7.5 illustrates some properties of the random geometric graph: Figure 7.5(a) shows the graph plot; Figure 7.5(b) shows the histogram of the degrees that concentrate around 30, as expected (this matches previous assertion that given proper parameters, the degree distribution of a random geometric graph is the same as that of an Erdős-Rényi graph [130]); Figure 7.5(c) shows the histogram of the leverage scores of $\mathrm{V}_{(20)}$, which are the optimal sampling scores when the SNR is large; and Figure 7.5(d) shows the histogram of the leverage score on a log-scale. We see that the histogram of the leverage scores is skewed; this means that a few nodes are more important than other nodes during sampling and have much higher probabilities to be chosen. In [130], the authors show that the cluster properties are different for a random geometric graph and an Erdős-Rényi graph. The proposed sampling scores capture these cluster properties through the decomposition



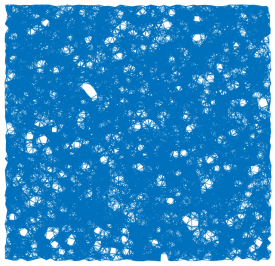
(a) Graph plot. (b) Histogram of the degrees. (c) Histogram of the leverage scores.

Figure 7.4: Properties of an Erdős-Rényi graph. Plot (c) shows the histogram of the leverage scores, which are the optimal sampling scores when the SNR is large.

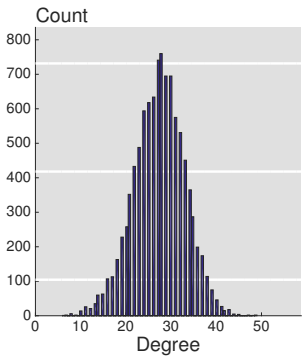
of the graph adjacency matrix. Based on the above, a random geometric graph is approximately a Type-2 graph.

Small-world graph. A small-world graph is a graph where most nodes are not neighbors of one another, but can be reached from any other node by a small number of hops (steps) [26, 131]. It is well known that many real-world graphs, including social networks, the connectivity of the Internet, and gene networks show small-world graph characteristics. We use a graph generated from the WattsStrogatz model, where a ring graph is first built, followed by the rewiring of the edges with probability 0.01%. Figure 7.6 illustrates some properties of the small-world graph: Figure 7.6(a) shows the graph plot; Figure 7.6(b) shows the histogram of the degrees that concentrate around 2 (a few nodes have 6 neighbors because of rewiring); Figure 7.6(c) shows the histogram of the leverage scores of $\mathrm{V}_{(20)}$, which are the optimal sampling scores when the SNR is large; and Figure 7.6(d) shows the histogram of the leverage scores on a log-scale. We see that the histogram of the leverage scores is skewed; this means that a few nodes are more important than others during sampling and have much higher probabilities to be chosen, similarly to the random geometric graph. Based on the above, a small-world graph is approximately a Type-2 graph.

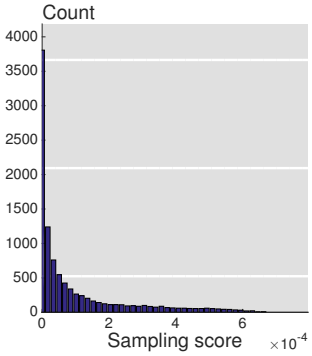
Power-law graph. A power-law graph is a graph where the more connected a node is, the more likely it is to receive new links, known as a preferential attachment graph [26, 131]. It is well known that the degree distribution of a preferential attachment graph follows the power law. We use a graph generated from the Barabási-Albert model, where new nodes are added to the network one at a time. Each new node is connected to one existing node with a probability that is proportional to the number of links that the existing nodes already have. Figure 7.7 illustrates some properties of the small-world graph: Figure 7.7(a) shows the graph plot; Figure 7.7(b) shows the histogram of the degrees that is skewed, which clearly follows the power law; Figure 7.7(c) shows the histogram of the leverage scores of $\mathrm{V}_{(20)}$, which are the optimal sampling scores when the SNR is large; and Figure 7.7(d) shows the histogram of the leverage scores on a log-scale. We see that the histogram of the leverage scores is skewed; this means that a few nodes are more important than others during sampling and have much higher probabilities to be chosen, similarly to the random geometric graph and the small-world graph. Based on the above,



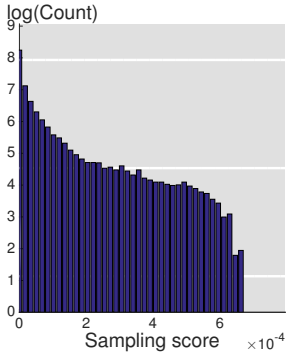
(a) Graph plot.



(b) Histogram of the degrees.



(c) Histogram of the leverage scores.



(d) Histogram of the leverage scores (log-scale).

Figure 7.5: Properties of a random geometric graph. Plot (c) shows the histogram of the leverages score, which are the optimal sampling scores when the SNR is large. Plot (d) shows the log-scale histogram of the leverage scores, which confirms that the leverage scores approximately follow a power-law distribution.

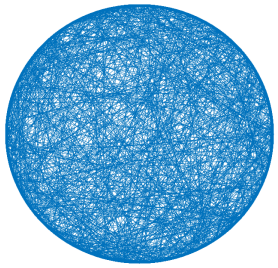
a power-law graph is approximately a Type-2 graph.

Types. Based on our observation of the graph Fourier transform matrices of each graph, the ring graph and the Erdős-Rényi graph approximately satisfy the requirement to be the Type-1 graph, while the random geometric graph, the small-world graph and the power-law graph approximately satisfy the requirement to be the Type-2 graph. We thus expect that the experimentally designed sampling has similar performance to uniform sampling for the ring graph and the Erdős-Rényi graph, while it outperforms uniform sampling for the random geometric graph, the small-world graph and the power-law graph.

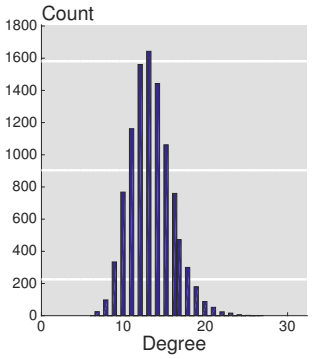
7.4.2 Simulated Graph Signals

For each graph A, we generate 1,000 graph signals through the following two steps: We first generate the graph frequency components as

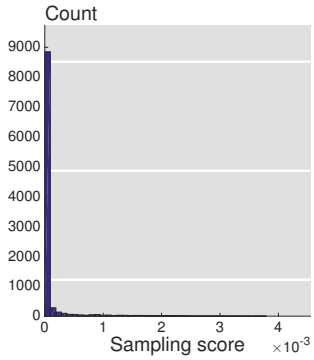
$$\widehat{x}_k \begin{cases} \sim \mathcal{N}(1, 0.5^2) & \text{when } k < K; \\ = K^{2\beta}/k^{2\beta} & \text{when } k \geq K. \end{cases} \tag{7.9}$$



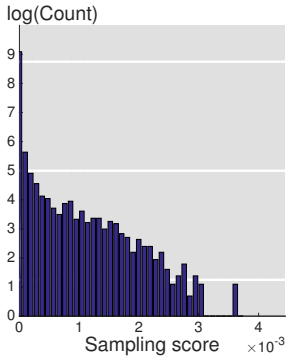
(a) Graph plot.



(b) Histogram of the degrees.



(c) Histogram of the leverage scores.



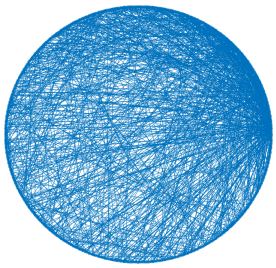
(d) Histogram of the leverage scores (log-scale).

Figure 7.6: Properties of a small-world graph. Plot (c) shows the histogram of the leverage score, which is the optimal sampling score when the SNR is large. Plot (d) shows the log-scale histogram of the leverage scores, which confirms that the leverage scores approximately follow a power-law distribution.

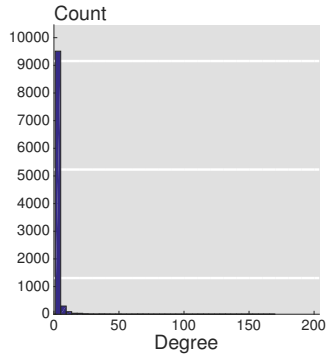
We then normalize $\widehat{\mathbf{x}}$ to have unit norm, and obtain $\mathbf{x} = \mathrm{V}\,\widehat{\mathbf{x}}$. It is clear that $\mathbf{x} \in \mathrm{ABL}_{\mathrm{A}}(K, \beta, \mu)$, where $K = 10$ and β varies as 0.5 and 1. During sampling, we simulate noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$, vary the sample size m from $1,000$ to $20,000$, and vary σ^2 from low noise level 10^{-4} ($\|\mathbf{x}\|_2 \,/\, \|\epsilon\|_2 = 100$) to high noise level 0.02 ($\|\mathbf{x}\|_2 \,/\, \|\epsilon\|_2 = 0.5$). During recovery, we set the bandwidth $K = \max(10, m^{1/2\beta+1})$ as suggested in Corollaries 6 and 8.

7.4.3 Results

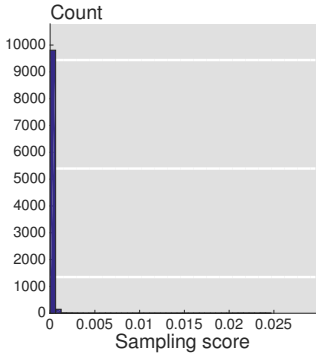
We compare four sampling strategies, including uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple), and degree based sampling (in red). Note that the last three sampling strategies all belong to experimentally designed sampling because they are designed based on the structure of the graph. As shown in Section 7.3.4, leverage score based sampling is approximately optimal when the SNR is large, square root of the leverage score based sampling is approximately the optimal when the SNR is small. We also use the degrees as the sampling scores because previous works show that



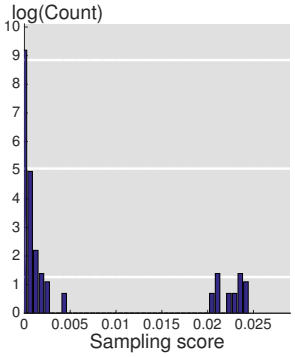
(a) Graph plot.



(b) Histogram of the degrees.



(c) Histogram of the leverage scores.



(d) Histogram of the leverage scores. (log-scale).

Figure 7.7: Properties of a power-law graph. Plot (c) shows the histogram of the leverage score, which is the optimal sampling score when the SNR is large. Plot (d) shows the log-scale histogram of the leverage scores, which confirms that the leverage scores approximately follow a power-law distribution.

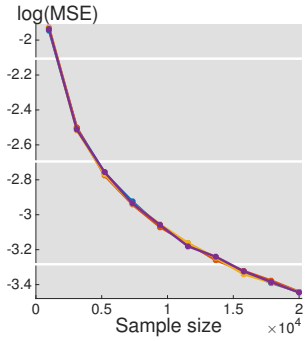
the largest eigenvectors of adjacency matrices often have most of their mass localized on high-degree nodes [94, 132], which implies the high correlation between degree and leverage score. We evaluate the recovery performance by using the MSE,

$$\text{MSE} \;=\; \|\mathbf{x}^* - \mathbf{x}\|_2^2 \,,$$

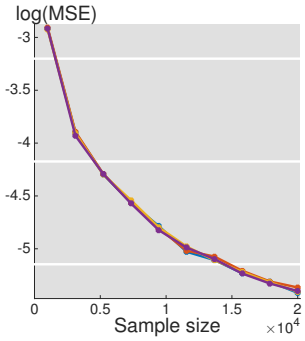
where $\mathbf{x}^*$ is the recovered graph signal and $\mathbf{x}$ is the original graph signal. The simulation results for the ring graph, the Erdős-Rényi graph, the random geometric graph, the small-world graph and the power-law graph are shown in Figures 7.8, 7.9, 7.10, 7.11 and 7.12, respectively. We summarize the important points below.

- All of the sampling strategies perform similarly on the ring graph and the Erdős-Rényi graph, matching Corollary 6.
- Experimentally designed sampling outperforms uniform sampling on the random geometric graph, the small-world graph and the power-law graph, matching Corollary 8. Especially for the small-world graph and the power-law graph, uniform sampling is much worse than experimentally designed sampling.

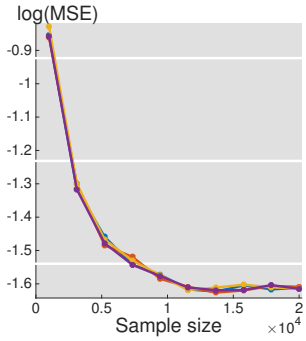
- Leverage score based sampling outperforms all other sampling strategies when the noise level is low.
- Square root of the leverage score based sampling outperforms all other sampling strategies when the noise level is high.
- Degree based sampling outperforms uniform sampling because of its correlation to the leverage score based sampling, but for the small-world graph, degree based sampling is still much worse than leverage score based sampling.
- When β is larger, the recovery performance is better because less energy is concentrated in the high-frequency band for approximately bandlimited graph signals.
- When σ^2 is smaller, the recovery performance is better because of less noise.
- The degree distribution is not a reliable indicator of when experimentally designed sampling outperforms uniform sampling. The degree distributions of the Erdős-Rényi graph and the random geometric graph are similar, but experimentally designed sampling only outperforms uniform sampling on the random geometric graph. This implies that the first-order information provided by the degree is not sufficient in designing samples.



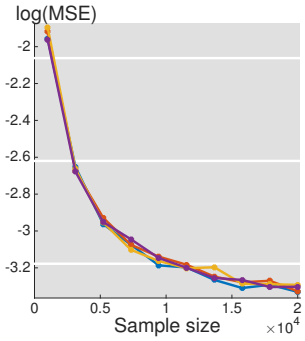
(a) $\beta = 0.5, \sigma^2 = 10^{-4}$



(b) $\beta = 1, \sigma^2 = 10^{-4}$

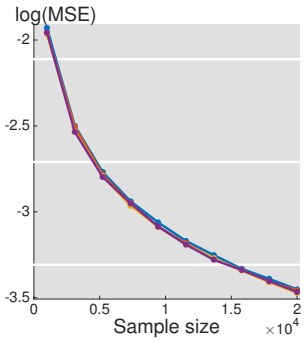


(d) $\beta = 0.5, \sigma^2 = 2 \times 10^{-2}$

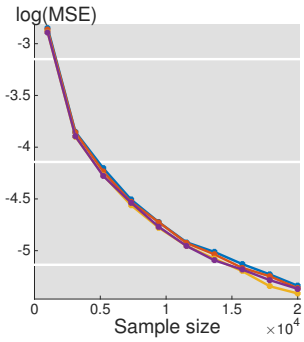


(e) $\beta = 1, \sigma^2 = 2 \times 10^{-2}$

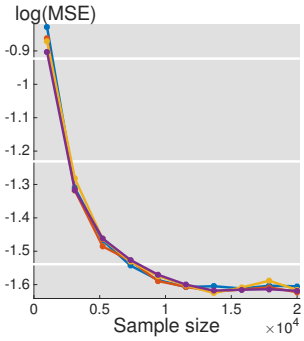
Figure 7.8: MSE comparison for the ring graph for uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple) and degree based sampling (in red).



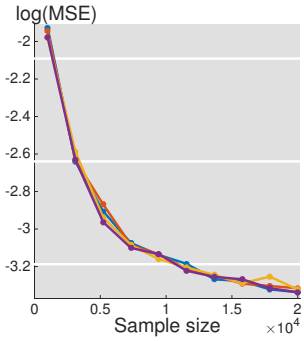
(a) $\beta = 0.5, \sigma^2 = 10^{-4}$.



(b) $\beta = 1, \sigma^2 = 10^{-4}$.



(c) $\beta = 0.5, \sigma^2 = 2 \times 10^{-2}$.



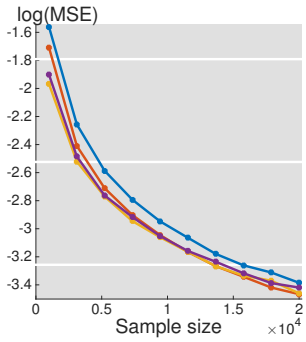
(d) $\beta = 1, \sigma^2 = 2 \times 10^{-2}$.

Figure 7.9: MSE comparison for the Erdős-Rényi graph for uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple) and degree based sampling (in red).

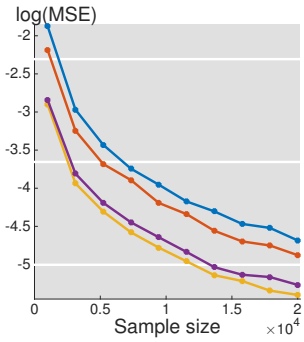
7.4.4 Discussion

Graph Fourier basis is critical for understanding the graph structure. For example, given the graph Fourier basis, Theorem 14 shows that active sampling does not fundamentally outperform experimentally designed sampling while Corollary 8 shows that experimentally designed sampling fundamentally outperforms uniform sampling on type-2 graphs. In other words, graph Fourier basis is critical for choosing samples.

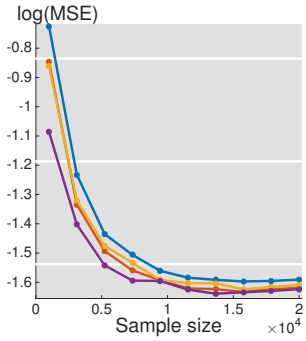
For irregular (Type-2) graphs, using experimentally designed sampling to choose anchor points can fundamentally aid semi-supervised learning (not true for regular, Type-1 graphs), a technique for training classifiers with both labeled and unlabeled data. Semi-supervised learning assumes that unlabeled data can provide distribution information to build a stronger classifier [106]. Many algorithms for semi-supervised learning are based on graphs that are constructed from a given dataset [106], often by modeling each node as a data sample and connecting two nodes by an edge if the distance between their features is in a given range, which is similar to the construction of random geometric graphs. Based on the assumption that adjacent nodes have similar labels, semi-supervised learning diffuses label probabilities from labeled data to unlabeled data along the graph structure and classifies unlabeled data according to those label



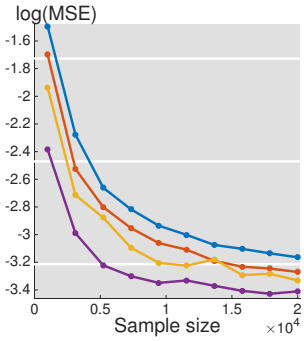
(a) $\beta = 0.5, \sigma^2 = 10^{-4}$.



(b) $\beta = 1, \sigma^2 = 10^{-4}$.



(c) $\beta = 0.5, \sigma^2 = 2 \times 10^{-2}$.



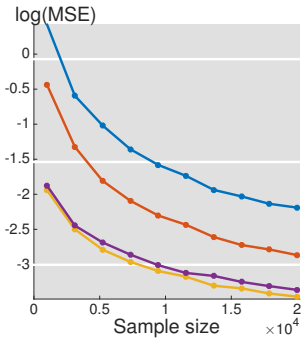
(d) $\beta = 1, \sigma^2 = 2 \times 10^{-2}$.

Figure 7.10: MSE comparison for the random geometric graph for uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple) and degree based sampling (in red).

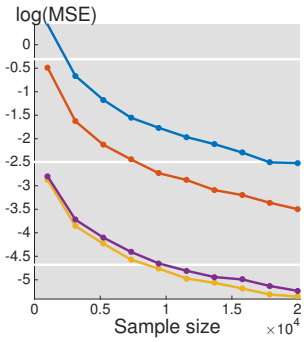
probabilities. While in some works [89, 133], training data is selected uniformly and randomly, in others, algorithms are designed adapted to the structure [61, 134], which is essentially equivalent to experimentally designed sampling we propose. In other words, experimentally designed sampling is used implicitly without being able to articulate when and why it works; this paper, on the other hand, provides a comprehensive explanation of why experimentally designed sampling helps semi-supervised learning through showing the lower and upper bounds of three sampling strategies.

7.5 Conclusions

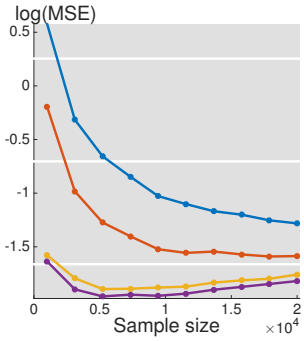
In this chapter, we built a theoretical foundation for the recovery of approximately bandlimited graph signals, which generalizes the class of bandlimited graph signals, under uniform sampling, experimentally designed sampling and active sampling. We showed that experimentally designed sampling and active sampling have the same fundamental limitations, and can outperform uniform sampling on irregular graphs. We proposed a recovery strategy and analyze its statistical properties. We showed that the proposed recovery strategy attains the optimal rates of convergence on two specific types of graphs. To validate the recovery strategy, we test it on five



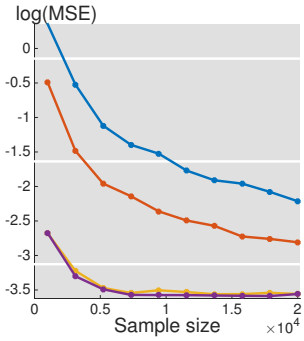
(a) $\beta = 0.5, \sigma^2 = 10^{-4}$.



(b) $\beta = 1, \sigma^2 = 10^{-4}$.



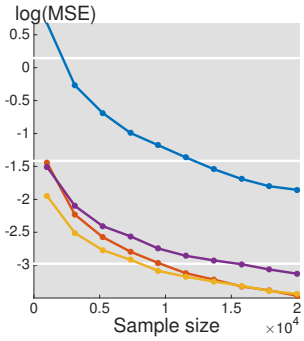
(c) $\beta = 0.5, \sigma^2 = 2 \times 10^{-2}$.



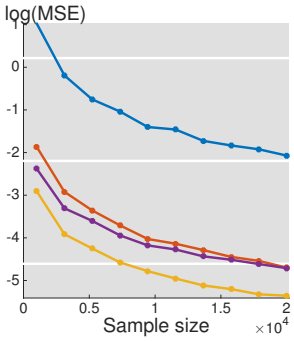
(d) $\beta = 1, \sigma^2 = 2 \times 10^{-2}$.

Figure 7.11: MSE comparison for the small-world graph for uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple) and degree based sampling (in red).

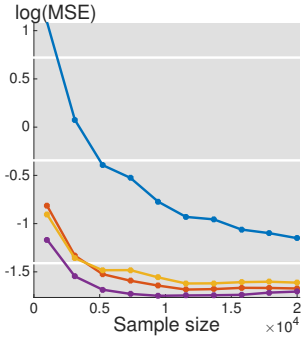
specific types of graphs: a ring graph with k nearest neighbors, an Erdős-Rényi graph, a random geometric graph, a small-world graph and a power-law graph, and show that experimental results match the proposed theory well. This chapter also provides a comprehensive explanation for why experimentally designed sampling works for semi-supervised learning with graphs and shows the critical role of the graph Fourier basis in analyzing graph structures.



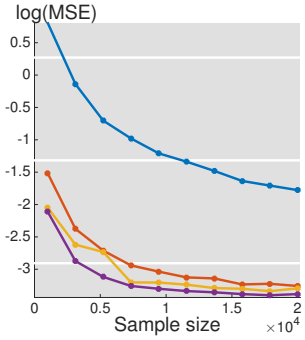
(a) $\beta = 0.5, \sigma^2 = 10^{-4}$.



(b) $\beta = 1, \sigma^2 = 10^{-4}$.



(c) $\beta = 0.5, \sigma^2 = 2 \times 10^{-2}$.



(d) $\beta = 1, \sigma^2 = 2 \times 10^{-2}$.

Figure 7.12: MSE comparison for the power-law graph for uniform sampling (in blue), leverage score based sampling (in orange), square root of the leverage score based sampling (in purple) and degree based sampling (in red).

Chapter 8

Recovery of Smooth Graph Signals

8.1 Introduction

In this chapter, we we consider the classical signal processing task of signal recovery within the context of graphs. Graph signal recovery attempts to recover one or multiple graph signals that are assumed to be smooth with respect to underlying graphs, from noisy, missing, or corrupted measurements. The smoothness constraint assumes that the signal samples at neighboring vertices are similar [76].

Most signal recovery problems in the current literature include image denoising [88,135], signal inpainting [136–138], and sensing [111,139], but are limited to signals with regular structure, such as time series. For example, Image denoising recovers an image from noisy observations. Standard techniques include Gaussian smoothing, Wiener local empirical filtering, and wavelet thresholding methods (see [135] and references therein); signal inpainting reconstructs lost or deteriorated parts of signals, including images and videos. Standard techniques include total variation-based methods [136–138, 140], image model-based methods [141], and sparse representations [142]; compressed sensing acquires and reconstructs signals by taking only a limited number of measurements [143, 144]. It assumes that signals are sparse and finds solutions to underdetermined linear systems by ℓ_1 techniques; matrix completion recovers the entire matrix from a subset of its entries by assuming that the matrix is of low rank. It was originally proposed in [145] and extensions include a noisy version in [146, 147] and decentralized algorithms via graphs [148]; and robust principal component analysis recovers a low-rank matrix from corrupted measurements [143, 144]; it separates an image into two parts: a smooth background and a sparse foreground. In contrast to principal component analysis, it is robust to grossly corrupted entries.

Existing work related to signal recovery based on spectral graph theory includes: 1) interpolation of bandlimited graph signals to recover bandlimited graph signals from a set with specific properties, called the uniqueness set [65, 109]. Extensions include the sampling theorem on graphs [60] and fast distributed algorithms [66]; and 2) smooth regularization on graphs to recover smooth graph signals from random samples [82, 149, 150].

We propose a graph signal model, cast graph signal recovery as an optimization problem, and provide a general solution by using the alternating direction method of multipliers. We

show that many classical recovery problems, such as signal inpainting [136–138], matrix completion [145, 146], and robust principal component analysis [143, 144], are related to the graph signal recovery problem. We propose theoretical solutions and new algorithms for graph signal inpainting, graph signal matrix completion, and anomaly detection of graph signals, all applicable to semi-supervised classification, regression, and matrix completion. Finally, we validate the proposed methods on real-world recovery problems, including online blog classification, bridge condition identification, temperature estimation, recommender system, and expert opinion combination.

8.2 Problem Formulation

Following the discussion in prologue of Part III, we aim to use a recovery operator Φ to recover a graph signal from noisy, missing, or corrupted measurements. This is hard in general; it is, however, possible to do this for signals with specific structure, such as smooth graph signals. Signal smoothness is a qualitative characteristic that expresses how much signal samples vary with respect to the underlying signal representation domain. To quantify it, we uses the ℓ_p-norm based graph total variation (2.9)[1] We normalize the graph shift to guarantee that the shifted signal is properly scaled with respect to the original one. When $p = 2$, we call (2.9) the *quadratic form of graph total variation*.

We now formulate the general recovery problem for graph signals to unify multiple signal completion and denoising problems and generalize them to arbitrary graphs. In the sections that follow, we consider specific cases of the graph signal recovery problem, propose appropriate solutions, and discuss their implementations and properties.

Let $\mathbf{x}^{(\ell)} \in \mathbb{C}^N$, $1 \leq \ell \leq L$, be graph signals residing on the graph $G = (\mathcal{V}, \mathrm{A})$, and let X be the $N \times L$ matrix of graph signals,

$$\mathrm{X} = \begin{bmatrix} \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \dots & \mathbf{x}^{(L)} \end{bmatrix}. \tag{8.1}$$

Assume that we do not know these signals exactly, but for each signal we have a corresponding measurement $\mathbf{t}^{(\ell)}$. Since each $\mathbf{t}^{(\ell)}$ can be corrupted by noise and outliers, we consider the $N \times L$ matrix of measurements to be

$$\mathrm{T} = \begin{bmatrix} \mathbf{t}^{(1)} & \mathbf{t}^{(2)} & \dots & \mathbf{t}^{(L)} \end{bmatrix} = \mathrm{X} + \mathrm{W} + \mathcal{E}, \tag{8.2}$$

where matrices W and $\mathcal{E}$ contain noise and outliers, respectively. Note that an outlier is an observation point that is distant from other observations, which may be due to variability in the measurement. We assume that the noise coefficients in W have small magnitudes, i.e., they can be upper-bounded by a small value, and that the matrix $\mathcal{E}$ is sparse, containing few non-zero coefficients of large magnitude. Furthermore, when certain nodes on a large graph are not accessible, the measurement $\mathbf{t}^{(\ell)}$ may be incomplete. To reflect this, we denote the sets of indices of accessible and inaccessible nodes as $\mathcal{M}$ and $\mathcal{U}$, respectively. Note that inaccessible

[1] For simplicity, throughout this paper we assume that the graph shift A has been normalized to satisfy $|\lambda_{\max}(\mathrm{A})| = 1$.

nodes denote that values on those nodes are far from the ground-truth because of corruption, or because we do not have access to them.

Signal recovery from inaccessible measurements requires additional knowledge of signal properties. In this work, we make the following assumptions: (a) the signals of interest $\mathbf{x}^{(\ell)}$, are smooth with respect to the representation graph $G = (\mathcal{V}, \mathrm{A})$; we express this by requiring that the variation of recovered signals be small; (b) since the signals of interest $\mathbf{x}^{(\ell)}$ are all supported on the same graph structure, we assume that these graph signals are similar and provide redundant information; we express this by requiring that the matrix of graph signals X has low rank; (c) the outliers happen with a small probability; we express this by requiring that the matrix $\mathcal{E}$ be sparse; and (d) the noise has small magnitude; we express this by requiring that the matrix W be upper-bounded. We thus formulate the problem as follows:

$$\widehat{\mathrm{X}}, \widehat{\mathrm{W}}, \widehat{\mathcal{E}} \;=\; \arg\min_{\mathrm{X},\mathrm{W},\mathcal{E}} \; \alpha\, \mathrm{S}_2(\mathrm{X}) + \beta\, \mathrm{rank}(\mathrm{X}) + \gamma\, \|\mathcal{E}\|_0\,, \tag{8.3}$$

$$\text{subject to} \qquad \|\mathrm{W}\|_F^2 \;\leq\; \epsilon^2, \tag{8.4}$$

$$\mathrm{T}_{\mathcal{M}} = (\mathrm{X} + \mathrm{W} + \mathcal{E})_{\mathcal{M}}, \tag{8.5}$$

where $\widehat{\mathrm{X}}, \widehat{\mathrm{W}}, \widehat{\mathcal{E}}$ denote the optimal solutions of the graph signal matrix, the noise matrix, and the outlier matrix, respectively, ϵ controls the noise level, α, β, γ are tuning parameters, and

$$\mathrm{S}_2(\mathrm{X}) \;=\; \sum_{\ell=1}^{L} \mathrm{S}_2(\mathbf{x}^{(\ell)}) = \|\mathrm{X} - \mathrm{A}\,\mathrm{X}\|_F^2\,,$$

where $\|\cdot\|_F$ denotes the Frobenius norm and represents the cumulative quadratic form of the graph total variation (2.9) for all graph signals, and $\|\mathcal{E}\|_0$ is the ℓ_0-norm that is defined as the number of nonzero entries in $\mathcal{E}$. The general problem (8.3) recovers the graph signal matrix (8.1) from the noisy measurements (8.2), possibly when only a subset of nodes is accessible.

Instead of using the graph total variation based on ℓ_1 norm [76], we use the quadratic form of the graph total variation (2.9) for two reasons. First, it is computationally easier to optimize than the ℓ_1-norm based graph total variation. Second, the ℓ_1-norm based graph total variation, which penalizes less transient changes than the quadratic form, is good at separating smooth from non-smooth parts of graph signals; the goal here, however, is to force graph signals at each node to be smooth. We thus use the quadratic form of the graph total variation in this paper and, by a slight abuse of notation, call it graph total variation for simplicity.

8.3 Methodology

The minimization problem (8.3) with conditions (8.4) and (8.5) reflects all of the above assumptions: (a) minimizing the graph total variation $\mathrm{S}_2(\mathrm{X})$ forces the recovered signals to be smooth and to lie in the subspace of "low" graph frequencies [76]; (b) minimizing the rank of X forces the graph signals to be similar and provides redundant information; (c) minimizing the ℓ_0-norm $\|\mathcal{E}\|_0$ forces the outlier matrix to have few non-zero coefficients; (d) condition (8.4) captures the

assumption that the coefficients of W have small magnitudes; and (e) condition (8.5) ensures that the solution coincides with the measurements on the accessible nodes.

Unfortunately, solving (8.3) is hard because of the rank and the ℓ_0-norm [151, 152]. To solve it efficiently, we relax and reformulate (8.3) as follows:

$$\widehat{\mathrm{X}}, \widehat{\mathrm{W}}, \widehat{\mathcal{E}} \;=\; \arg\min_{\mathrm{X},\mathrm{W},\mathcal{E}} \; \alpha\, \mathrm{S}_2(\mathrm{X}) + \beta\, \|\mathrm{X}\|_* + \gamma\, \|\mathcal{E}\|_1\,, \tag{8.6}$$

$$\text{subject to} \qquad \|\mathrm{W}\|_F^2 \;\leq\; \epsilon^2, \tag{8.7}$$

$$\mathrm{T}_{\mathcal{M}} = (\mathrm{X} + \mathrm{W} + \mathcal{E})_{\mathcal{M}}. \tag{8.8}$$

In (8.6), we replace the rank of X with the nuclear norm, $\|\mathrm{X}\|_*$, defined as the sum of all the singular values of X, which still promotes low rank [145, 146]. We further replace the ℓ_0-norm of $\mathcal{E}$ with the ℓ_1-norm, which still promotes sparsity of $\mathcal{E}$ [151, 152]. The minimization problems (8.3) and (8.6) follow the same assumptions and promote the same properties, but (8.3) is an ideal version, while (8.6) is a practically feasible version, because it is a convex problem and thus easier to solve. We call (8.6) the *graph signal recovery* (GSR) problem; see Table 8.1.

To solve (8.6) efficiently, we use the alternating direction method of multipliers (ADMM) [153]. ADMM is an algorithm that is intended to take advantage of both the decomposability and the superior convergence properties of the method of multipliers. In ADMM, we first formulate an augmented function and then iteratively update each variable in an alternating or sequential fashion, ensuring the convergence of the method [153]. We leave the derivation to the Appendix, and summarize the implementation in Algorithm 3. Note that in Algorithm 3, $\mathrm{Y}_1, \mathrm{Y}_2$ are Lagrangian

Algorithm 3 Graph Signal Recovery

Input T matrix of measurements
Output $\widehat{\mathrm{X}}$ matrix of graph signals
$\widehat{\mathrm{W}}$ matrix of outliers
$\widehat{\mathcal{E}}$ matrix of noise

Function **GSR**(T)
while the stopping criterion is not satisfied
  while the stopping criterion is not satisfied
    $\mathrm{X} \leftarrow \mathrm{D}_{\beta\eta^{-1}}\left(\mathrm{X} + t(\mathrm{T} - \mathrm{W} - \mathcal{E} - \mathrm{C} - \eta^{-1}(\mathrm{Y}_1 + \mathrm{Y}_2) - \mathrm{Z})\right)$
  end
  $\mathrm{W} \leftarrow \eta(\mathrm{T} - \mathrm{X} - \mathcal{E} - \mathrm{C} - \eta^{-1}\,\mathrm{Y}_1)/(\eta + 2)$
  while the stopping criterion is not satisfied
    $\mathcal{E} \leftarrow \Theta_{\gamma\eta^{-1}}\left(\mathrm{X} + t(\mathrm{T} - \mathrm{X} - \mathrm{W} - \mathrm{C} - \eta^{-1}\,\mathrm{Y}_1)\right)$
  end
  $\mathrm{Z} \leftarrow (\mathbf{I} + 2\alpha\eta^{-1}(\mathbf{I} - \mathrm{A})^*(\mathbf{I} - \mathrm{A}))^{-1}(\mathrm{X} - \eta^{-1}\,\mathrm{Y}_2)$
  $\mathrm{C}_{\mathcal{M}} \leftarrow 0, \mathrm{C}_{\mathcal{U}} \leftarrow (\mathrm{T} - \mathrm{X} - \mathrm{W} - \mathcal{E} - \eta^{-1}\,\mathrm{Y}_1)_{\mathcal{U}}$,
  $\mathrm{Y}_1 \leftarrow \mathrm{Y}_1 - \eta(\mathrm{T} - \mathrm{X} - \mathrm{W} - \mathcal{E} - \mathrm{C})$
  $\mathrm{Y}_2 \leftarrow \mathrm{Y}_2 - \eta(\mathrm{X} - \mathrm{Z})$
end
return $\widehat{\mathrm{X}} \leftarrow \mathrm{X}$, $\widehat{\mathrm{W}} \leftarrow \mathrm{W}$, $\widehat{\mathcal{E}} \leftarrow \mathcal{E}$

multipliers, η is pre-defined, the step size t is chosen from backtracking line search [154], and

operators Θ_τ and D_τ are defined for $\tau \geq 0$ as follows: $\Theta_\tau(\mathrm{X})$ "shrinks" every element of X by τ so that the (n, m)th element of $\Theta_\tau(\mathrm{X})$ is

$$\Theta_\tau(\mathrm{X})_{n,m} = \begin{cases} \mathrm{X}_{n,m} - \tau, & \text{when} \quad \mathrm{X}_{n,m} \geq \tau, \\ \mathrm{X}_{n,m} + \tau, & \text{when} \quad \mathrm{X}_{n,m} \leq -\tau, \\ 0, & \text{otherwise.} \end{cases} \tag{8.9}$$

Similarly, $\mathrm{D}_\tau(\mathrm{X})$ "shrinks" the singular values of X,

$$\mathrm{D}_\tau(\mathrm{X}) = \mathrm{U}\, \Theta_\tau\left(\Sigma\right) \mathrm{Q}^*, \tag{8.10}$$

where $\mathrm{X} = \mathrm{U}\, \Sigma\, \mathrm{Q}^*$ denotes the singular value decomposition of X [84] and $*$ denotes the Hermitian transpose. The following stopping criterion is used in the paper: the difference of the objective function between two consecutive iterations is smaller than 10^{-8}. The bulk of the computational cost is in the singular value decomposition (8.10) when updating X, which is also involved in the standard implementation of matrix completion.

We now review several well-known algorithms for signal recovery, including signal inpainting, matrix completion, and robust principal component analysis, and show how they can be formulated as special cases of the graph signal recovery problem (8.3). In Sections 8.3.1 and 8.3.2, we show graph counterparts of the signal inpainting and matrix completion problems by minimizing the graph total variation. In Section 8.3.3, we show anomaly detection on graphs, which is inspired by robust principal component analysis.

Signal inpainting. Signal inpainting is a process of recovering inaccessible or corrupted signal samples from accessible samples using regularization [136–138, 140], that is, minimization of the signal's total variation. The measurement is typically modeled as

$$\mathbf{t} = \mathbf{x} + \mathbf{w} \in \mathbb{R}^N, \tag{8.11}$$

where $\mathbf{x}$ is the true signal, and $\mathbf{w}$ is the noise. Assuming we can access a subset of indices, denoted as $\mathcal{M}$, the task is then to recover the entire true signal $\mathbf{x}$, based on the accessible measurement $\mathbf{t}_\mathcal{M}$. We assume that the true signal $\mathbf{x}$ is smooth, that is, its variation is small. The variation is expressed by a *total variation* function

$$\mathrm{TV}\left(\mathbf{x}\right) \;=\; \sum_{i=1}^{N} |x_i - x_{i-1 \bmod N}|. \tag{8.12}$$

We then recover the signal $\mathbf{x}$ by solving the following optimization problem:

$$\widehat{\mathbf{x}} \;=\; \arg\min_{\mathbf{x}} \; \mathrm{TV}(\mathbf{x}), \tag{8.13}$$

$$\text{subject to} \qquad \|(\mathbf{x} - \mathbf{t})_\mathcal{M}\|_2^2 \;\leq\; \epsilon^2. \tag{8.14}$$

The condition (8.14) controls how well the accessible measurements are preserved. Both the ℓ_1 norm based graph total variation and the quadratic form of the graph total variation (2.9) are used. Thus, (8.13) is a special case of (8.6) when the graph shift is the cyclic permutation matrix,

$\alpha = 1$, $L = 1$, $\beta = \gamma = 0$, $\mathcal{E} = 0$, and conditions (8.7) and (8.8) are combined into the single condition (8.14); see Table 8.1.

Matrix completion. Matrix completion recovers a matrix given a subset of its elements, usually, a subset of rows or columns. Typically, the matrix has a low rank, and the missing part is recovered through rank minimization [145–147]. The matrix is modeled as

$$\mathrm{T} = \mathrm{X} + \mathrm{W} \in \mathbb{R}^{N \times L}, \tag{8.15}$$

where X is the true matrix and W is the noise. Assuming we can access a subset of indices, denoted as $\mathcal{M}$, the matrix X is recovered from (8.15) as the solution with the lowest rank:

$$\widehat{\mathrm{X}} \;=\; \arg\min_{\mathrm{X}} \|\mathrm{X}\|_* \,, \tag{8.16}$$

$$\text{subject to} \qquad \|(\mathrm{X} - \mathrm{T})_{\mathcal{M}}\|_2^2 \;\leq\; \epsilon^2; \tag{8.17}$$

this is a special case of (8.6) with $\alpha = \gamma = 0$, $\beta = 1$, $\mathcal{E} = 0$, and conditions (8.7) and (8.8) are combined into the single condition (8.17); see Table 8.1. This also means that the values in the matrix are associated with a graph that is represented by the identity matrix, that is, we do not have any prior information about the graph structure.

Robust principal component analysis. Similarly to matrix completion, robust principal component analysis is used for recovering low-rank matrices. The main difference is the assumption that all matrix elements are measurable but corrupted by outliers [143, 144]. In this setting, the matrix is modeled as

$$\mathrm{T} = \mathrm{X} + \mathcal{E} \in \mathbb{R}^{N \times L}, \tag{8.18}$$

where X is the true matrix, and $\mathcal{E}$ is a sparse matrix of outliers.

The matrix X is recovered from (8.18) as the solution with the lowest rank and fewest outliers:

$$\widehat{\mathrm{X}}, \widehat{\mathcal{E}} \;=\; \arg\min_{\mathrm{X},\mathcal{E}} \beta \|\mathrm{X}\|_* + \gamma \|\mathcal{E}\|_1 \,,$$
$$\text{subject to} \qquad \mathrm{T} = \mathrm{X} + \mathcal{E};$$

this is a special case of (8.6) with $\alpha = \epsilon = 0$, $\mathrm{W} = 0$, and $\mathcal{M}$ contains all the indices; see Table 8.1. Like before, this also means that the matrix is associated with a graph that is represented by the identity matrix, that is, we do not have any prior information about the graph structure.

8.3.1 Graph Signal Inpainting

We now discuss in detail the problem of signal inpainting on graphs. Parts of this section have appeared in [118], and we include them here for completeness.

As discussed in the previous section, signal inpainting (8.13) seeks to recover the missing entries of the signal $\mathbf{x}$ from incomplete and noisy measurements under the assumption that two consecutive signal samples in $\mathbf{x}$ have similar values. Here, we treat $\mathbf{x}$ as a graph signal that is smooth with respect to the corresponding graph. We thus update the signal inpainting prob-

lem (8.13), and formulate the *graph signal inpainting* problem[2] as

$$\widehat{\mathbf{x}} \;=\; \arg\min_{\mathbf{x}} \; \mathrm{S}_2(\mathbf{x}), \tag{8.19}$$

$$\text{subject to} \qquad \|(\mathbf{x}-\mathbf{t})_{\mathcal{M}}\|_2^2 \;\leq\; \epsilon^2; \tag{8.20}$$

this is a special case of (8.6) with $L=1, \beta=\gamma=0$; see Table 8.1.

Solutions. In general, graph signal inpainting (8.19) can be solved by using Algorithm 3. However, in special cases, there exist closed-form solutions that do not require iterative algorithms.

Noiseless inpainting

Suppose that the measurement $\mathbf{t}$ in (8.11) does not contain noise. In this case, $\mathbf{w}=0$, and we solve (8.19) for $\epsilon=0$:

$$\widehat{\mathbf{x}} \;=\; \arg\min_{\mathbf{x}} \; \mathrm{S}_2(\mathbf{x}), \tag{8.21}$$

$$\text{subject to} \qquad \mathbf{x}_{\mathcal{M}} = \mathbf{t}_{\mathcal{M}}.$$

We call the problem (8.21) *graph signal inpainting via total variation minimization* (GTVM) [118].

Let $\widetilde{\mathrm{A}} = (\mathbf{I}-\mathrm{A})^*(\mathbf{I}-\mathrm{A})$. By reordering nodes, write $\widetilde{\mathrm{A}}$ in block form as

$$\widetilde{\mathrm{A}} = \begin{bmatrix} \widetilde{\mathrm{A}}_{\mathcal{M}\mathcal{M}} & \widetilde{\mathrm{A}}_{\mathcal{M}\mathcal{U}} \\ \widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}} & \widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}} \end{bmatrix},$$

and set the derivative of (8.21) to 0; the closed-form solution is

$$\widehat{\mathbf{x}} = \begin{bmatrix} \mathbf{t}_{\mathcal{M}} \\ -\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}}^{-1}\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}} \end{bmatrix}.$$

When $\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}}$ is not invertible, a pseudoinverse should be used.

Unconstrained inpainting

The graph signal inpainting (8.19) can be formulated as an unconstrained problem by merging condition (8.20) with the objective function:

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \; \|(\mathbf{x}-\mathbf{t})_{\mathcal{M}}\|_2^2 + \alpha\, \mathrm{S}_2(\mathbf{x}), \tag{8.22}$$

where the tuning parameter α controls the trade-off between the two parts of the objective function. We call (8.22) the *graph signal inpainting via total variation regularization* (GTVR).

[2]If we build a graph to model a dataset by representing signals or images in the dataset as nodes and the similarities between each pair of nodes as edges, the corresponding labels or values associated with nodes thus form a graph signal, and the proposed inpainting algorithm actually tackles semi-supervised learning with graphs [106].

GTVR is a convex quadratic problem that has a closed-form solution. Setting the derivative of (8.22) to zero, we obtain the closed-form solution

$$\widehat{\mathbf{x}} = \left(\begin{bmatrix} \mathbf{I}_{\mathcal{M}\mathcal{M}} & 0 \\ 0 & 0 \end{bmatrix} + \alpha \widetilde{\mathrm{A}} \right)^{-1} \begin{bmatrix} \mathbf{t}_{\mathcal{M}} \\ \mathbf{0} \end{bmatrix},$$

where $\mathbf{I}_{\mathcal{M}\mathcal{M}}$ is an identity matrix. When the term in parentheses is not invertible, a pseudoinverse should be adopted.

Theoretical analysis. Let $\mathbf{x}^0$ denote the *true* graph signal that we are trying to recover. Assume that $\mathrm{S}_2(\mathbf{x}^0) = \eta^2$ and $\mathbf{x}^0$ satisfies (8.20), so that $\|\mathbf{x}^0_{\mathcal{M}} - \mathbf{t}_{\mathcal{M}}\|_2^2 \leq \epsilon^2$. Similarly to (8.22), we write A in a block form as

$$\mathrm{A} = \begin{bmatrix} \mathrm{A}_{\mathcal{M}\mathcal{M}} & \mathrm{A}_{\mathcal{M}\mathcal{U}} \\ \mathrm{A}_{\mathcal{U}\mathcal{M}} & \mathrm{A}_{\mathcal{U}\mathcal{U}} \end{bmatrix}.$$

The following results, proven in [118], establish an upper bound on the error of the solution to graph signal inpainting (8.19).

Lemma 3. The error $\|\mathbf{x}^0 - \widehat{\mathbf{x}}\|_2$ of the solution $\widehat{\mathbf{x}}$ to the graph signal inpainting problem (8.19) is bounded as

$$\left\|\mathbf{x}^0 - \widehat{\mathbf{x}}\right\|_2 \leq \frac{q}{2} \left\|(\mathbf{x}^0 - \widehat{\mathbf{x}})_{\mathcal{U}}\right\|_2 + p|\epsilon| + |\eta|,$$

where

$$p = \left\| \begin{bmatrix} \mathbf{I}_{\mathcal{M}\mathcal{M}} + \mathrm{A}_{\mathcal{M}\mathcal{M}} \\ \mathrm{A}_{\mathcal{U}\mathcal{M}} \end{bmatrix} \right\|_2, \quad q = \left\| \begin{bmatrix} \mathrm{A}_{\mathcal{M}\mathcal{U}} \\ \mathbf{I}_{\mathcal{U}\mathcal{U}} + \mathrm{A}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \right\|_2,$$

and $\|\cdot\|_2$ for matrices denotes the spectral norm.

Theorem 16. If $q < 2$, then the error on the inaccessible part of the solution $\widehat{\mathbf{x}}$ is bounded as

$$\left\|(\mathbf{x}^0 - \widehat{\mathbf{x}})_{\mathcal{U}}\right\|_2 \leq \frac{2p|\epsilon| + 2|\eta|}{2 - q}.$$

The condition $q < 2$ may not hold for some matrices; however, if A is symmetric, we have $q \leq \|\mathbf{I} + \mathrm{A}\|_2 \leq \|\mathbf{I}\|_2 + \|\mathrm{A}\|_2 = 2$, since $\|\mathrm{A}\|_2 = 1$. Since q is related to the size of the inaccessible part, when we take a larger number of measurements, q becomes smaller, which leads to a tighter upper bound. Also, note that the upper bound is related to the smoothness of the true graph signal and the noise level of the accessible part. A central assumption of any inpainting technique is that the true signal $\mathbf{x}^0$ is smooth. If this assumption does not hold, then the upper bound is large and useless. When the noise level of the accessible part is smaller, the measurements from the accessible part are closer to the true values, which leads to a smaller estimation error.

8.3.2 Graph Signal Matrix Completion

We now consider graph signal matrix completion—another important subproblem of the general graph signal recovery problem (8.3).

Matrix completion seeks to recover missing entries of matrix X from the incomplete and noisy measurement matrix (8.15) under the assumption that X has low rank. Since we view X

as a matrix of graph signals (see (8.1)), we also assume that the columns of X are smooth graph signals. In this case, we update the matrix completion problem (8.16) and formulate the *graph signal matrix completion* problem as

$$\begin{aligned} \widehat{\mathrm{X}} &= \mathbf{argmin}_{\mathrm{X}}\ \mathrm{S}_2(\mathrm{X}) + \beta \left\|\mathrm{X}\right\|_*, \\ \text{subject to} &\quad \left\|(\mathrm{X}-\mathrm{T})_{\mathcal{M}}\right\|_F^2 \ \leq\ \epsilon^2; \end{aligned} \tag{8.23}$$

this is a special case of (8.6) with $\alpha = 1, \gamma = 0$; see Table 8.1.

Solutions. In addition to Algorithm 3 that can be used to solve the graph signal matrix completion problem (8.23), there exist alternative approaches that we discuss next.

Minimization

Here we consider the noise-free case. Suppose the measurement matrix T in (8.15) does not contain noise. We thus solve (8.23) for $\mathrm{W} = 0$ and $\epsilon = 0$,

$$\begin{aligned} \widehat{\mathrm{X}} &= \mathbf{argmin}_{\mathrm{X}}, \mathrm{S}_2(\mathrm{X}) + \beta \left\|\mathrm{X}\right\|_*, \\ \text{subject to} &\quad \mathrm{X}_{\mathcal{M}} = \mathrm{T}_{\mathcal{M}}\,. \end{aligned} \tag{8.24}$$

We call (8.24) *graph signal matrix completion via total variation minimization* (GMCM). This is a constrained convex problem that can be solved by projected generalized gradient descent [154]. We first split the objective function into two components, a convex, differential component, and a convex, nondifferential component; based on these two components, we formulate a proximity function and then solve it iteratively. In each iteration, we solve the proximity function with an updated input and project the result onto the feasible set. To be more specific, we split the objective function (8.24) into a convex, differentiable component $\mathrm{S}_2(\mathrm{X})$, and a convex, nondifferential component $\beta \left\|\mathrm{X}\right\|_*$. The proximity function is then defined as

$$\begin{aligned} \mathrm{prox}_t(\mathrm{X}) &= \arg\min_{\mathrm{Z}} \frac{1}{2t} \left\|\mathrm{X}-\mathrm{Z}\right\|^2 + \beta \left\|\mathrm{X}\right\|_* \\ &= \mathrm{D}_{t\beta}(\mathrm{Z}), \end{aligned}$$

where $\mathrm{D}(\cdot)$ is defined in (8.10). In each iteration, we first solve for the proximity function and project the result onto the feasible set as

$$\mathrm{X} \leftarrow \mathrm{proj}\left(\mathrm{prox}_t\left(\mathrm{X} - t\nabla \mathrm{S}_2(\mathrm{X})\right)\right),$$

where t is the step size that is chosen from the backtracking line search [154], and $\mathrm{proj}(\mathrm{X})$ projects X to the feasible set so that the (n,m)th element of $\mathrm{proj}(\mathrm{X})$ is

$$\mathrm{proj}(\mathrm{X})_{n,m} = \begin{cases} \mathrm{T}_{n,m}, & \text{when} \quad (n,m) \in \mathcal{M}, \\ \mathrm{X}_{n,m}, & \text{when} \quad (n,m) \in \mathcal{U}. \end{cases}$$

For implementation details, see Algorithm 4. The bulk of the computational cost of Algorithm 4 is in the singular value decomposition (8.10) when updating X, which is also involved in the standard implementation of matrix completion.

Algorithm 4 Graph Signal Matrix Completion via Total Variation Minimization

Input T matrix of measurements
$\widehat{\mathrm{X}}$ matrix of graph signals

Function **GMCM**(T)
initialize X, such that $\mathrm{X}_{\mathcal{M}} = \mathrm{T}_{\mathcal{M}}$ holds
while the stopping criterion is not satisfied
Choose step size t from backtracking line search
$\mathrm{X} \leftarrow \mathrm{proj}\left(\mathrm{D}_{t\beta}(\mathrm{X} - 2t\widetilde{\mathrm{A}}\,\mathrm{X})\right)$
end
return $\widehat{\mathrm{X}} \leftarrow \mathrm{X}$

Regularization

The graph signal matrix completion (8.23) can be formulated as an unconstrained problem,

$$\widehat{\mathrm{X}} \;=\; \mathbf{argmin}_{\mathrm{X}}, \|\mathrm{X}_{\mathcal{M}} - \mathrm{T}_{\mathcal{M}}\|_F^2 + \alpha\,\mathrm{S}_2(\mathrm{X}) + \beta\,\|\mathrm{X}\|_*\,. \tag{8.25}$$

We call (8.25) *graph signal matrix completion via total variation regularization* (GMCR). This is an unconstrained convex problem and can be solved by generalized gradient descent. Similarly to projected generalized gradient descent, generalized gradient descent also formulates and solves a proximity function. The only difference is that generalized gradient descent does not need to project the result after each iteration to a feasible set. To be more specific, we split the objective funtion (8.24) into a convex, differentiable component $\|\mathrm{X}_{\mathcal{M}} - \mathrm{T}_{\mathcal{M}}\|_F^2 + \alpha\,\mathrm{S}_2(\mathrm{X})$, and a convex, non-differential component $\beta\,\|\mathrm{X}\|_*$. The proximity function is then defined as

$$\begin{aligned}\mathrm{prox}_t(\mathrm{X}) \;&=\; \arg\min_{\mathrm{Z}} \frac{1}{2t}\,\|\mathrm{X}-\mathrm{Z}\|^2 + \beta\,\|\mathrm{X}\|_* \\ &=\; \mathrm{D}_{t\beta}(\mathrm{Z}),\end{aligned} \tag{8.26}$$

where $\mathrm{D}(\cdot)$ is defined in (8.10). In each iteration, we first solve for the proximity function as

$$\mathrm{X} \leftarrow \mathrm{prox}_t\left(\mathrm{X} - t\nabla\left(\|\mathrm{X}_{\mathcal{M}} - \mathrm{T}_{\mathcal{M}}\|_F^2 + \alpha\,\mathrm{S}_2(\mathrm{X})\right)\right), \tag{8.27}$$

where t is the step size that is chosen from the backtracking line search [154]; for implementation details, see Algorithm 5. The bulk of the computational cost of Algorithm 5 is in the singular value decomposition (8.10) when updating X, which is also involved in the standard implementation of matrix completion.

Theoretical analysis. We now discuss properties of the proposed algorithms. The key in classical matrix completion is to minimize the nuclear norm of a matrix. Instead of considering general matrices, we only focus on graph signal matrices, whose corresponding algorithm is to minimize both the graph total variation and the nuclear norm. We study the connection between graph total variation and nuclear norm of a matrix to reveal the underlying mechanism of our algorithm.

Let X be a $N \times L$ matrix of rank r with singular value decomposition $\mathrm{X} = \mathrm{U}\,\Sigma\,\mathrm{Q}^*$, where $\mathrm{U} = \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_r\end{bmatrix}$, $\mathrm{Q} = \begin{bmatrix}\mathbf{q}_1 & \mathbf{q}_2 & \dots & \mathbf{q}_r\end{bmatrix}$, and Σ is a diagonal matrix with σ_i along the diagonal, $i = 1, \cdots, r$.

Algorithm 5 Graph Signal Matrix Completion via Total Variation Regularization

Input T matrix of measurements
$\widehat{\mathrm{X}}$ matrix of graph signals

Function **GMCR**(T)
initialize X
while the stopping criterion is not satisfied
Choose step size t from backtracking line search
$\mathrm{X} \leftarrow \mathrm{D}_{t\beta}\left(\mathrm{X} - 2t(\mathrm{X}_{\mathcal{M}} - \mathrm{T}_{\mathcal{M}}) - 2\alpha t \widetilde{\mathrm{A}}\,\mathrm{X}\right)$
end
return $\widehat{\mathrm{X}} \leftarrow \mathrm{X}$

Lemma 4.

$$\mathrm{S}_2(\mathrm{X}) = \sum_{i=1}^{r} \sigma_i^2 \, \|(\mathbf{I} - \mathrm{A})\mathbf{u}_i\|_2^2.$$

Proof.

$$\begin{aligned}
\mathrm{S}_2(\mathrm{X}) &= \|\mathrm{X} - \mathrm{A}\,\mathrm{X}\|_F^2 \overset{(a)}{=} \|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\Sigma\,\mathrm{Q}^*\|_F^2, \\
&= \mathrm{Tr}\left(\mathrm{Q}\,\Sigma\,\mathrm{U}^*(\mathbf{I} - \mathrm{A})^*(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\Sigma\,\mathrm{Q}^*\right), \\
&\overset{(b)}{=} \mathrm{Tr}\left(\Sigma\,\mathrm{U}^*(\mathbf{I} - \mathrm{A})^*(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\Sigma\,\mathrm{Q}^*\,\mathrm{Q}\right), \\
&= \|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\Sigma\|_F^2 \overset{(c)}{=} \sum_{i=1}^{r} \sigma_i^2 \, \|(\mathbf{I} - \mathrm{A})\mathbf{u}_i\|_2^2,
\end{aligned}$$

where (a) follows from the singular value decomposition; (b) from the cyclic property of the trace operator; and (c) from Σ being a diagonal matrix. □

From Lemma 4, we see that graph total variation is related to the rank of X; in other words, lower rank naturally leads to smaller graph total variation.

Theorem 17.

$$\mathrm{S}_2(\mathrm{X}) \leq \mathrm{S}_2(\mathrm{U}) \, \|\mathrm{X}\|_*^2 .$$

Proof. From Lemma 4, we have

$$\begin{aligned}
\mathrm{S}_2(\mathrm{X}) &= \|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\Sigma\|_F^2 \overset{(a)}{\leq} \|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\|_F^2 \, \|\Sigma\|_F^2 , \\
&\overset{(b)}{\leq} \|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\|_F^2 \, \|\Sigma\|_*^2 = \|\mathrm{U} - \mathrm{A}\,\mathrm{U}\|_F^2 \, \|\mathrm{X}\|_*^2 ,
\end{aligned}$$

where (a) follows from the submultiplicativity of the Frobenius norm; and (b) from the norm equivalence [84]. □

Graph signal recovery problem	$\widehat{\mathrm{X}}, \widehat{\mathrm{W}}, \widehat{\mathcal{E}} = \arg\min_{\mathrm{X},\mathrm{W},\mathcal{E}\in\mathbb{R}^{N\times L}} \alpha\, \mathrm{S}_2(\mathrm{X}) + \beta \left\|\mathrm{X}\right\|_* + \gamma \left\|\mathcal{E}\right\|_1$, subject to $\left\|\mathrm{W}\right\|_F^2 \leq \epsilon^2, \mathrm{T}_{\mathcal{M}} = (\mathrm{X}+\mathrm{W}+\mathcal{E})_{\mathcal{M}}$.
Signal inpainting	$L = 1, \beta = 0, \gamma = 0$, graph shift is the cyclic permutation matrix.
Matrix completion	$\alpha = 0$, or graph shift is the identity matrix, $\gamma = 0$
Robust principal component analysis	$\alpha = 0$, or graph shift is the identity matrix, $\epsilon = 0$, $\mathcal{M}$ is all the indices in T.
Graph signal inpainting	$L = 1, \beta = 0, \gamma = 0$
Graph signal matrix completion	$\alpha = 1, \gamma = 0$
Anomaly detection	$L = 1, \beta = 0, \mathcal{M}$ is all indices in T
Robust graph signal inpainting	$L = 1, \beta = 0$

Table 8.1: The table of algorithms.

In Theorem 4, we see that the graph total variation is related to two quantities: the nuclear norm of X and the graph total variation of the left singular vectors of X. The first quantity reveals that minimizing the nuclear norm potentially leads to minimizing the graph total variation. We can thus rewrite the objective function (8.23) as

$$\mathrm{S}_2(\mathrm{X}) + \beta \left\|\mathrm{X}\right\|_* \;\leq\; \mathrm{S}_2(\mathrm{U}) \left\|\mathrm{X}\right\|_*^2 + \beta \left\|\mathrm{X}\right\|_*.$$

If the graph shift is built from insufficient information, we just choose a larger β to force the nuclear norm to be small, which causes a small graph total variation in return. The quantity $\mathrm{S}_2(\mathrm{U})$ measures the smoothness of the left singular vectors of X on a graph shift A; in other words, when the left singular vectors are smooth, the graph signal matrix is also smooth. We can further use this quantity to bound the graph total variation of all graph signals that belong to a subspace spanned by the left singular vectors.

Theorem 18. Let a graph signal $\mathbf{x}$ belong to the space spanned by U, that is, $\mathbf{x} = \mathrm{U}\,\mathbf{a}$, where $\mathbf{a}$ is the vector of representation coefficients. Then,

$$\mathrm{S}_2(\mathbf{x}) \leq \mathrm{S}_2(\mathrm{U}) \left\|\mathbf{a}\right\|_2^2$$

Proof.

$$\begin{aligned} \mathrm{S}_2(\mathbf{x}) \;&=\; \left\|\mathbf{x} - \mathrm{A}\,\mathbf{x}\right\|_2^2 \;=\; \left\|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\,\mathbf{a}\right\|_2^2 \\ &\overset{(a)}{\leq}\; \left\|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\right\|_2^2 \left\|\mathbf{a}\right\|_2^2 \;\overset{(b)}{\leq}\; \left\|(\mathbf{I} - \mathrm{A})\,\mathrm{U}\right\|_F^2 \left\|\mathbf{a}\right\|_2^2. \end{aligned}$$

where (a) follows from the submultiplicativity of the spectral norm; and (b) from the norm equivalence [84]. □

Theorem 18 shows that a graph signal is smooth when it belongs to a subspace spanned by the smooth left singular vectors.

8.3.3 Anomaly Detection

We now consider anomaly detection of graph signals, another important subproblem of the general recovery problem (8.3).

Robust principal component analysis seeks to detect outlier coefficients from a low-rank matrix. Here, anomaly detection of graph signals seeks to detect outlier coefficients from a smooth graph signal. We assume that the outlier is sparse and contains few non-zero coefficients of large magnitude. To be specific, the measurement is modeled as

$$\mathbf{t} = \mathbf{x} + \mathbf{e} \in \mathbb{R}^N, \tag{8.28}$$

where $\mathbf{x}$ is a smooth graph signal that we seek to recover, and the outlier $\mathbf{e}$ is sparse and has large magnitude on few nonzero coefficients. The task is to detect the outlier $\mathbf{e}$ from the measurement $\mathbf{t}$. Assuming that $\mathbf{x}$ is smooth, that is, its variation is small, and $\mathbf{e}$ is sparse, we propose the ideal optimization problem as follows:

$$\begin{aligned} \widehat{\mathbf{x}}, \widehat{\mathbf{e}} &= \arg\min_{\mathbf{x},\mathbf{e}} \|\mathbf{e}\|_0 \\ \text{subject to} \quad & \mathrm{S}_2(\mathbf{x}) \leq \eta^2, \\ & \mathbf{t} = \mathbf{x} + \mathbf{e}. \end{aligned} \tag{8.29}$$

To solve it efficiently, instead of dealing with the ℓ_0 norm, we relax it to the ℓ_1 norm and reformulate (8.29) as follows:

$$\widehat{\mathbf{x}}, \widehat{\mathbf{e}} = \arg\min_{\mathbf{x},\mathbf{e}} \|\mathbf{e}\|_1 \tag{8.30}$$

$$\text{subject to} \quad \mathrm{S}_2(\mathbf{x}) \leq \eta^2, \tag{8.31}$$

$$\mathbf{t} = \mathbf{x} + \mathbf{e}; \tag{8.32}$$

this is a special case of (8.6) with $L = 1, \beta = 0, \mathcal{M}$ contains all indices in $\mathbf{t}$, and choosing α properly to ensure that (8.30) holds, see Table 8.1. In Section 8.3.3, we show that, under these assumptions, both (8.29) and (8.30) lead to perfect outlier detection.

Solutions. The minimization problem (8.30) is convex, and it is numerically efficient to solve for its optimal solution.

We further formulate an unconstrained problem as follows:

$$\widehat{\mathbf{e}} = \arg\min_{\mathbf{e}} \mathrm{S}_2(\mathbf{t} - \mathbf{e}) + \beta \|\mathbf{e}\|_1 . \tag{8.33}$$

We call (8.33) *anomaly detection via ℓ_1 regularization* (AD). In (8.33), we merge conditions (8.31) and (8.32) and move them from the constraint to the objective function. We solve (8.33) by using generalized gradient descent, as discussed in Section 8.3.2. For implementation details, see Algorithm 6.

Theoretical analysis. Let $\mathbf{x}^0$ be the true graph signal, represented as $\mathbf{x}^0 = \mathrm{V}\,\mathbf{a}^0 = \sum_{i=0}^{N-1} a_i^0 \mathbf{v}_i$, where V is defined in (2.6), $\mathbf{e}^0$ be the outliers that we are trying to detect, represented as $\mathbf{e}^0 = \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i$, where $\boldsymbol{\delta}_i$ is impulse on the ith node, and $\mathcal{E}$ contains the outlier indices, that is, $\mathcal{E} \subset \{0, 1, 2, \cdots N-1\}$, and $\mathbf{t} = \mathbf{x}^0 + \mathbf{e}^0$ be the measurement.

Lemma 5. Let $\widehat{\mathbf{x}}, \widehat{\mathbf{e}}$ be the solution of (8.29), and let $\widehat{\mathbf{x}} = \mathrm{V}\,\widehat{\mathbf{a}} = \sum_{i=0}^{N-1} \widehat{a}_i \mathbf{v}_i$. Then,

$$\widehat{\mathbf{e}} = \mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}) + \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i.$$

Algorithm 6 Anomaly detection via ℓ_1 regularization

Input $\mathbf{t}$ input graph signals
Output $\widehat{\mathbf{e}}$ outlier signals

Function $\mathbf{AD}(\mathbf{x})$
initialize $\mathbf{e}$
while the stopping criterion is not satisfied
  Choose step size t from backtracking line search
  $\mathbf{e} \leftarrow \Theta_{t\beta}\left(\mathbf{e} - 2t\widetilde{\mathrm{A}}(\mathbf{t}-\mathbf{e})\right)$
end
return $\widehat{\mathbf{e}} \leftarrow \mathbf{e}$

Proof.

$$\widehat{\mathbf{e}} \overset{(a)}{=} \mathbf{t} - \widehat{\mathbf{x}} \overset{(b)}{=} \mathbf{x}^0 + \mathbf{e}^0 - \widehat{\mathbf{x}} \overset{(c)}{=} \mathrm{V}\,\mathbf{a}^0 + \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i - \mathrm{V}\,\widehat{\mathbf{a}},$$

where (a) follows from the feasibility of $\widehat{\mathbf{x}}, \widehat{\mathbf{e}}$ in (8.30); (b) from the definition of $\mathbf{t}$; and (c) from the definitions of $\mathbf{x}^0$ and $\widehat{\mathbf{x}}$. □

Lemma 5 provides another representation of the outliers, which is useful in the following theorem.

Let $\mathrm{K} = (\mathbf{I} - \Lambda)^T \mathrm{V}^T \mathrm{V}(\mathbf{I} - \Lambda)$, the K norm as $\|\mathbf{x}\|_{\mathrm{K}} = \sqrt{\mathbf{x}^T \mathrm{K}\,\mathbf{x}}$, and $\mathcal{K}_\eta = \{\mathbf{a} \in \mathbb{R}^N : \|\mathbf{a}\|_{\mathrm{K}} \leq \eta, \text{for all } \mathbf{a} \neq 0\}$.

Lemma 6. Let $\mathbf{x} = \mathrm{V}\,\mathbf{a}$ satisfy (8.31), (8.32), and $\mathbf{a} \neq 0$. Then, $\mathbf{a} \in \mathcal{K}_\eta$.

Proof.

$$\begin{aligned} \mathrm{S}_2(\mathbf{x}) &= \|\mathbf{x} - \mathrm{A}\,\mathbf{x}\|_2^2 = \|\mathrm{V}\,\mathbf{a} - \mathrm{A}\,\mathrm{V}\,\mathbf{a}\|_2^2, \\ &\overset{(a)}{=} \|\mathrm{V}\,\mathbf{a} - \mathrm{V}\,\Lambda\mathbf{a}\|_2^2 = \mathbf{a}^T(\mathbf{I} - \Lambda)^T \mathrm{V}^T \mathrm{V}(\mathbf{I} - \Lambda)\mathbf{a}, \\ &\overset{(b)}{=} \mathbf{a}^T \mathrm{K}\,\mathbf{a} \overset{(c)}{\leq} \eta^2, \end{aligned}$$

where (a) follows from the eigendecomposition; (b) from the definition of the K norm; and (c) from the feasibility of $\mathbf{x}$. □

Lemma 6 shows that the frequency components of the solution from (8.29) and (8.30) belong to a subspace, which is useful in the following theorem.

Theorem 19. Let $\mathrm{S}_2(\mathbf{x}^0) \leq \eta^2$, $\mathbf{x}^0 \neq 0$, $\|\mathbf{e}^0\|_0 \leq k$, and $\widehat{\mathbf{x}}, \widehat{\mathbf{e}}$ be the solution of (8.29) with $\widehat{\mathbf{x}} \neq 0$. Let $\mathcal{K}_{2\eta}$ have the following property:

$$\|\mathrm{V}\,\mathbf{a}\|_0 \geq 2k + 1 \quad \text{for all } \mathbf{a} \in \mathcal{K}_{2\eta},$$

where $k \geq \|\mathbf{e}^0\|_0$. Then, perfect recovery is achieved,

$$\widehat{\mathbf{x}} = \mathbf{x}^0, \widehat{\mathbf{e}} = \mathbf{e}^0.$$

Proof. Since both $\mathbf{x}^0 = \mathrm{V}\,\mathbf{a}^0$ and $\widehat{\mathbf{x}} = \mathrm{V}\,\widehat{\mathbf{a}}$ are feasible solutions of (8.29), by Lemma 6, we then have that $\mathbf{a}^0, \widehat{\mathbf{a}} \in \mathcal{K}_\eta$. We next bound their difference, $\mathbf{a}^0 - \widehat{\mathbf{a}}$, by the triangle inequality, as $\|\mathbf{a}^0 - \widehat{\mathbf{a}}\|_{\mathrm{K}} \leq \|\mathbf{a}^0\|_{\mathrm{K}} + \|\widehat{\mathbf{a}}\|_{\mathrm{K}} \leq 2\eta$.

If $\mathbf{a}^0 \neq \widehat{\mathbf{a}}$, then $\mathbf{a}^0 - \widehat{\mathbf{a}} \in \mathcal{K}_{2\eta}$, so that $\|\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\|_0 \geq 2k+1$. From Lemma 5, we have $\|\widehat{\mathbf{e}}\|_0 = \left\|\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}) + \sum_{i\in\mathcal{E}} b_i \mathbf{e}_i\right\|_0 \geq k+1$. The last inequality comes from the fact that at most k indices can be canceled by the summation.

On the other hand, $\widehat{\mathbf{e}}$ is the optimum of (8.29), thus, $\|\widehat{\mathbf{e}}\|_0 \leq \|\mathbf{e}^0\|_0 = k$, which leads to a contradiction.

Therefore, $\widehat{\mathbf{a}} = \mathbf{a}^0$ and $\widehat{\mathbf{e}} = \mathbf{e}^0$. □

Theorem 19 shows the condition that leads to perfect outlier detection by following (8.29). The key is that the outliers are sufficiently sparse and the smooth graph signal is not sparse.

Theorem 20. Let $\mathrm{S}_2(\mathbf{x}^0) \leq \eta^2$, $\mathbf{x}^0 \neq 0$, $\|\mathbf{e}^0\|_0 \leq k$, and $\widehat{\mathbf{x}}, \widehat{\mathbf{e}}$ be the solution of (8.30) with $\widehat{\mathbf{x}} \neq 0$. Let $\mathcal{K}_{2\eta}$ have the following property:

$$\|(\mathrm{V}\,\mathbf{a})_{\mathcal{E}^c}\|_1 > \|(\mathrm{V}\,\mathbf{a})_{\mathcal{E}}\|_1 \quad \text{for all } \mathbf{a} \in \mathcal{K}_{2\eta}$$

where $\mathcal{E}^c \cap \mathcal{E}$ is the empty set, $\mathcal{E}^c \cup \mathcal{E} = \{0, 1, 2, \cdots N-1\}$. Then, perfect recovery is achieved,

$$\widehat{\mathbf{x}} = \mathbf{x}^0, \widehat{\mathbf{e}} = \mathbf{e}^0.$$

Proof. From Lemma 5, we have

$$\begin{aligned}
\|\widehat{\mathbf{e}}\|_1 &= \left\|\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}) + \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i\right\|_1 \\
&= \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}^c} + \big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}} + \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i\right\|_1 \\
&= \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}^c}\right\|_1 + \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}} + \sum_{i\in\mathcal{E}} b_i \boldsymbol{\delta}_i\right\|_1
\end{aligned}$$

Denote $(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}))_{\mathcal{E}} = \sum_{i\in\mathcal{E}} d_i \boldsymbol{\delta}_i$, we further have

$$\begin{aligned}
\|\widehat{\mathbf{e}}\|_1 &= \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}^c}\right\|_1 + \left\|\sum_{i\in\mathcal{E}} (d_i + b_i)\boldsymbol{\delta}_i\right\|_1 \\
&= \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}^c}\right\|_1 + \sum_{i\in\mathcal{E}} |d_i + b_i|
\end{aligned}$$

If $\mathbf{a}^0 \neq \widehat{\mathbf{a}}$, then $\mathbf{a}^0 - \widehat{\mathbf{a}} \in \mathcal{K}_{2\eta}$. By the assumption, we have $\|(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}))_{\mathcal{E}^c}\|_1 > \|(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}}))_{\mathcal{E}}\|_1 = \left\|\sum_{i\in\mathcal{E}} d_i \boldsymbol{\delta}_i\right\|_1 = \sum_{i\in\mathcal{E}} |d_i|$. We thus obtain

$$\begin{aligned}
\|\widehat{\mathbf{e}}\|_1 &= \left\|\big(\mathrm{V}(\mathbf{a}^0 - \widehat{\mathbf{a}})\big)_{\mathcal{E}^c}\right\|_1 + \sum_{i\in\mathcal{E}} |d_i + b_i| \\
&> \sum_{i\in\mathcal{E}} (|d_i| + |d_i + b_i|) \geq \sum_{i\in\mathcal{E}} |b_i|.
\end{aligned}$$

On the other hand, $\widehat{\mathbf{e}}$ is the optimum of (8.30), so $\|\widehat{\mathbf{e}}\|_1 \leq \|\mathbf{e}^0\|_1 = \left\|\sum_{i\in\mathcal{E}} b_i\boldsymbol{\delta}_i\right\|_1 = \sum_{i\in\mathcal{E}} |b_i|$, which leads to a contradiction.

Therefore, $\widehat{\mathbf{a}} = \mathbf{a}^0$ and $\widehat{\mathbf{e}} = \mathbf{e}^0$. □

Theorems 19 and 20 show that under appropriate assumptions, (8.29), (8.30) detects the outliers perfectly. Note that the assumptions on $\mathcal{K}$ in Theorems 19 and 20 are related to two factors: the upper bound on smoothness, η, and the eigenvector matrix, V. The volume of $\mathcal{K}_{2\eta}$ is determined by the upper bound on smoothness, η. The mapping properties of V are also restricted by Theorems 19 and 20. For instance, in Theorem 19, the eigenvector matrix should map each element in $\mathcal{K}_{2\eta}$ to a non-sparse vector.

Robust graph signal inpainting. One problem with the graph signal inpainting in Section 8.3.1 is that it tends to trust the accessible part, which may contain sparse, but large-magnitude outliers. Robust graph signal inpainting should prevent the solution from being influenced by the outliers. We thus consider the following optimization problem:

$$\widehat{\mathbf{x}}, \widehat{\mathbf{w}}, \widehat{\mathbf{e}} = \arg\min_{\mathbf{x},\mathbf{w},\mathbf{e}} \alpha\, \mathrm{S}_2(\mathbf{x}) + \gamma \|\mathbf{e}\|_0 , \tag{8.34}$$

$$\text{subject to} \quad \|\mathbf{w}\|_F^2 < \eta^2 \tag{8.35}$$

$$\mathbf{t}_{\mathcal{M}} = (\mathbf{x} + \mathbf{w} + \mathbf{e})_{\mathcal{M}}; \tag{8.36}$$

this is a special case of (8.3) with $L = 1, \beta = 0$; see Table 8.1.

Similarly to (8.6), instead of dealing with the ℓ_0 norm, we relax it to be the ℓ_1 norm and reformulate (8.34) as an unconstrained problem,

$$\widehat{\mathbf{x}}, \widehat{\mathbf{e}} = \arg\min_{\mathbf{x},\mathbf{e}} \|\mathbf{t}_{\mathcal{M}} - (\mathbf{x} + \mathbf{e})_{\mathcal{M}}\|^2 + \alpha\, \mathrm{S}_2(\mathbf{x}) + \gamma \|\mathbf{e}\|_1 . \tag{8.37}$$

We call problem (8.37) the *robust graph total variation regularization* (RGTVR) problem. In (8.37), we merge conditions (8.35) and (8.36) and move them from the constraint to the objective function. Note that (8.37) combines anomaly detection and graph signal inpainting to provide a twofold inpainting. The first level detects the outliers in the accessible part and provides a clean version of the accessible measurement; the second level uses the clean measurement to recover the inaccessible part. We solve (8.37) by using ADMM. For implementation details, see Algorithm 7.

8.4 Experimental Results

We now evaluate the proposed methods on several real-world recovery problems. Further, we apply graph signal inpainting and robust graph signal inpainting to online blog classification and bridge condition identification for indirect bridge structural health monitoring; We apply graph signal matrix completion to temperature estimation and expert opinion combination.

Datasets. We use the following datasets in the experiments:

Algorithm 7 Robust Graph Total Variation Regularization

Input $\mathbf{t}$ input graph signal
Output $\widehat{\mathbf{e}}$ outlier graph signal
$\widehat{\mathbf{x}}$ output graph signal

Function **RGTVR(t)**
while the stopping criterion is not satisfied
$\mathbf{x} \leftarrow \left(\mathbf{I}+2\alpha\eta^{-1}\widetilde{\mathrm{A}}\right)^{-1}(\mathbf{t}-\mathbf{e}-\mathbf{w}-\mathbf{c}-\eta^{-1}\lambda)$
$\mathbf{w} \leftarrow \eta(\mathbf{t}-\mathbf{x}-\mathbf{w}-\mathbf{c}-\eta^{-1}\lambda)/(\eta+2)$
$\mathbf{e} \leftarrow \Theta_{\gamma\eta^{-1}}(\mathbf{t}-\mathbf{x}-\mathbf{e}-\mathbf{c}-\eta^{-1}\lambda)$
$\lambda \leftarrow \lambda-\eta(\mathbf{t}-\mathbf{x}-\mathbf{e}-\mathbf{w}-\mathbf{c})$
$\mathbf{c}_{\mathcal{M}} \leftarrow 0, \mathbf{c}_{\mathcal{U}} \leftarrow (\mathbf{t}-\mathbf{x}-\mathbf{w}-\mathbf{e}-\eta^{-1}\lambda)_{\mathcal{U}}$,
end
return $\widehat{\mathbf{e}} \leftarrow \mathbf{e}$, $\widehat{\mathbf{x}} \leftarrow \mathbf{x}$

Online blogs

We consider the problem of classifying $N = 1224$ online political blogs as either conservative or liberal [116]. We represent conservative labels as $+1$ and liberal ones as -1. The blogs are represented by a graph in which nodes represent blogs, and directed graph edges correspond to hyperlink references between blogs. For a node v_n, its outgoing edges have weights $1/\deg(v_n)$, where $\deg(v_n)$ is the out-degree of v_n (the number of outgoing edges). The graph signal here is the label assigned to the blogs.

Acceleration signals

We next consider the bridge condition identification problem [155, 156]. To validate the feasibility of indirect bridge structural health monitoring, a lab-scale bridge-vehicle dynamic system was built. Accelerometers were installed on a vehicle that travels across the bridge; acceleration signals were then collected from those accelerometers. To simulate the severity of different bridge conditions on a lab-scale bridge, masses with various weights were put on the bridge. We collected 30 acceleration signals for each of 31 mass levels from 0 to 150 grams in steps of 5 grams, to simulate different degrees of damages, for a total of 930 acceleration signals. For more details on this dataset, see [157].

The recordings are represented by an 8-nearest neighbor graph, in which nodes represent recordings, and each node is connected to eight other nodes that represent the most similar recordings. The graph signal here is the mass level over all the acceleration signals. The graph shift A is formed as $\mathrm{A}_{i,j} = \mathrm{P}_{i,j} \,/\, \sum_i \mathrm{P}_{i,j}$, with

$$\mathrm{P}_{i,j} \;=\; \exp\left(\frac{-N^2\,\|\mathbf{f}_i-\mathbf{f}_j\|_2}{\sum_{i,j}\|\mathbf{f}_i-\mathbf{f}_j\|_2}\right),$$

and $\mathbf{f}_i$ is a vector representation of the features of the ith recording. Note that P is a symmetric matrix that represents an undirected graph and the graph shift A is an asymmetric matrix that represents a directed graph, which is allowed by the framework of $\mathrm{DSP_G}$. From the empirical performance, we find that a directed graph provides much better results than an undirected graph.

Temperature data

We consider 150 weather stations in the United States that record their local temperatures [2]. Each weather station has 365 days of recordings (one recording per day), for a total of 54,750 measurements. The graph representing these weather stations is obtained by measuring the geodesic distance between each pair of weather stations. The nodes are represented by an 8-nearest neighbor graph, in which nodes represent weather stations, and each node is connected to eight other nodes that represent the eight closest weather stations. The graph signals here are the temperature values recorded in each weather station.

The graph shift A is formed as $\mathrm{A}_{i,j} = \mathrm{P}_{i,j} \,/\, \sum_i \mathrm{P}_{i,j}$, with

$$\mathrm{P}_{i,j} = \exp\left(-\frac{N^2 d_{i,j}}{\sum_{i,j} d_{i,j}}\right),$$

where $d_{i,j}$ is the geodesic distance between the ith and the jth weather stations. Similarly to the acceleration signals, we normalize P to obtain a asymmetric graph shift, which represents a directed graph, to achieve better empirical performance.

Jester dataset 1

The Jester joke data set [158] contains 4.1×10^6 ratings of 100 jokes from 73,421 users. The graph representing the users is obtained by measuring the ℓ_1 norm of existing ratings between each pair of jokes. The nodes are represented by an 8-nearest neighbor graph in which nodes represent users and each node is connected to eight other nodes that represent similar users. The graph signals are the ratings of each user. The graph shift A is formed as

$$\mathrm{P}_{i,j} \;=\; \exp\left(\frac{-N^2 \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_1}{\sum_{i,j} \left\|\mathbf{f}_i - \mathbf{f}_j\right\|_1}\right),$$

where $\mathbf{f}_i$ is the vector representation of the existing ratings for the ith user. Similarly to acceleration signals, we normalize P to obtain an asymmetric graph shift, which represents a directed graph, to achieve better empirical performance.

Evaluation score. To evaluate the performance of the algorithms, we use the following four metrics: accuracy (ACC), mean square error (MSE), root mean square error (RMSE), and mean absolute error (MAE), defined as

$$\begin{aligned}
\mathrm{ACC} &= \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(x_i = \widehat{x}_i), \\
\mathrm{MSE} &= \frac{1}{N}\sum_{i=1}^{N} (x_i - \widehat{x}_i)^2, \\
\mathrm{RMSE} &= \sqrt{\frac{1}{N}\sum_{i=1}^{N} (x_i - \widehat{x}_i)^2} = \sqrt{\mathrm{MSE}}, \\
\mathrm{MAE} &= \frac{\sum_{i=1}^{N} |x_i - \widehat{x}_i|}{N},
\end{aligned}$$

where x_i is the ground-truth for the ith sample, $\widehat{x}_i$ is the estimate for the ith sample, and $\mathbf{1}$ is the indicator function, $\mathbf{1}(x) = 1$, for $x = 0$, and 0 otherwise.

In the following applications, the tuning parameters for each algorithm are chosen by cross-validation; that is, we split the accessible part into a training part and a validation part. We train the model with the training part and choose the tuning parameter that provides the best performance in the validation part.

Applications of graph signal inpainting. Parts of this subsection have appeared in [118]; we include them here for completeness. We apply the proposed graph signal inpainting algorithm to online blog classification and bridge condition identification. We compare the proposed GTVR (8.22) with another regression model based on graphs, graph Laplacian regularization regression (LapR) [82, 149, 150]. As described in Section 10.1, the main difference between LapR and GTVR is that a graph Laplacian matrix in LapR is restricted to be symmetric and only represents an undirected graph; a graph shift in GTVR can be either symmetric or asymmetric.

Online blog classification

We consider a semi-supervised classification problem, that is, classification with few labeled data and a large amount of unlabeled data [106]. The task is to classify the unlabeled blogs. We adopt the dataset of blogs as described in Section 8.4. We randomly labeled 0.5%, 1%, 2%, 5%, and 10% of blogs, called the *labeling ratio*. We then applied the graph signal inpainting algorithms to estimate the labels for the remaining blogs. Estimated labels were thresholded at zero, so that positive values were set to $+1$ and negative to -1.

Classification accuracies of GTVR and LapR were then averaged over 30 tests for each labeling ratio and are shown in Figure 8.1. We see that GTVR achieves significantly higher accuracy than LapR for low labeling ratios. The failure of LapR at low labeling ratios is because an undirected graph fails to reveal the true structure.

Figure 8.1 also shows that the performance of GTVR saturates at around 95%. Many of the remaining errors are misclassification of blogs with many connections to the blogs from a different class, which violates the smoothness assumption underlying GTVR. Because of the same reason, the performance of a data-adaptive graph filter also saturates at around 95% [159]. To improve on this performance may require using a more sophisticated classifier that we will pursue in future work.

Bridge condition identification

We consider a semi-supervised regression problem, that is, regression with few labeled data and a large amount of unlabeled data [106]. The task is to predict the mass levels of unlabeled acceleration signals. We adopt the dataset of acceleration signals as described in Section 8.4. We randomly assigned known masses to 0.5%, 1%, 2%, 5%, and 10% of acceleration signals and applied the graph signal inpainting algorithms to estimate the masses for remaining nodes.

Figure 8.2 shows MSEs for estimated masses averaged over 30 tests for each labeling ratio. The proposed GTVR approach yields significantly smaller errors than LapR for low labeling ratios. Similarly to the conclusion of online blog classification, a direct graph adopted in GTVR reveals a better structure.

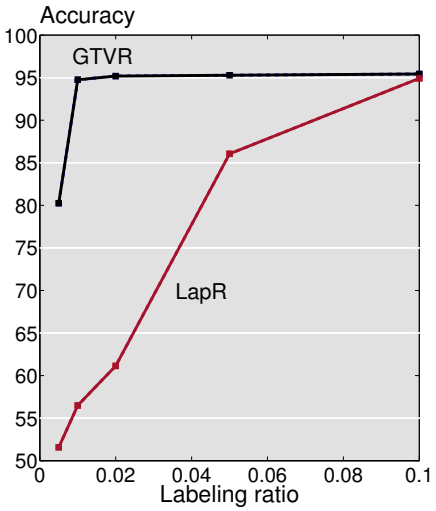


Figure 8.1: Accuracy comparison of online blog classification as a function of labeling ratio.

We observe that the performance of GTVR saturates at 3 in terms of MSE. This may be the result of how we obtain the graph. Here we construct the graph by using features from principal component analysis of the data. Since the data is collected with a real lab-scale model, which is complex and noisy, the principal component analysis may not extract all the useful information from the data, limiting the performance of the proposed method even with larger number of samples.

Applications of graph signal matrix completion. We now apply the proposed algorithm to temperature estimation, recommender systems and expert opinion combination of online blog classification. We compare the proposed GMCR (8.25) with matrix completion algorithms. Those algorithms include SoftImpute [160], OptSpace [147], singular value thresholding (SVT) [161], weighted non-negative matrix factorization (NMF) [162], and graph-based weighted nonnegative matrix factorization (GWNMF) [163]. Similarly to the matrix completion algorithm described in Section 8.3, SoftImpute, OptSpace, and SVT minimize the rank of a matrix in similar, but different ways. NMF is based on matrix factorization, assuming that a matrix can be factorized into two nonnegative, low-dimensional matrices; GWNMF extends NMF by further constructing graphs on columns or rows to represent the internal information. In contrast to the proposed graph-based methods, GWNMF considers the graph structure in the hidden layer. For a fair comparison, we use the same graph structure for GWNMF and GMCM. NMF and GWNMF solve non-convex problems and get local minimum.

Temperature estimation

We consider matrix completion, that is, estimation of the missing entries in a data matrix [145]. The task is to predict missing temperature values in an incomplete temperature data matrix where each column corresponds to the temperature values of all the weather stations from each day. We adopt the dataset of temperature data described in Section 8.4 [2]. In each day of temperature recording, we randomly hide $50\%, 60\%, 70\%, 80\%, 90\%$ measurements and apply the proposed

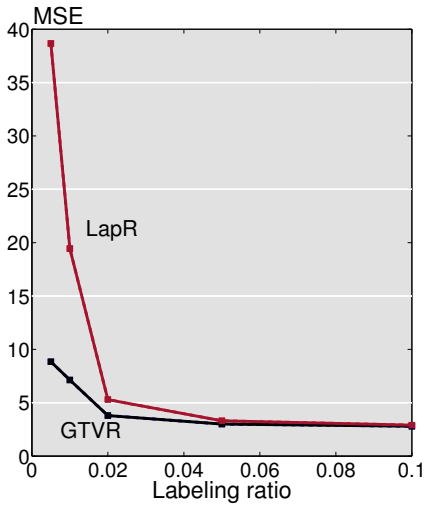


Figure 8.2: MSE comparison for the bridge condition identification as a function of labeling ratio.

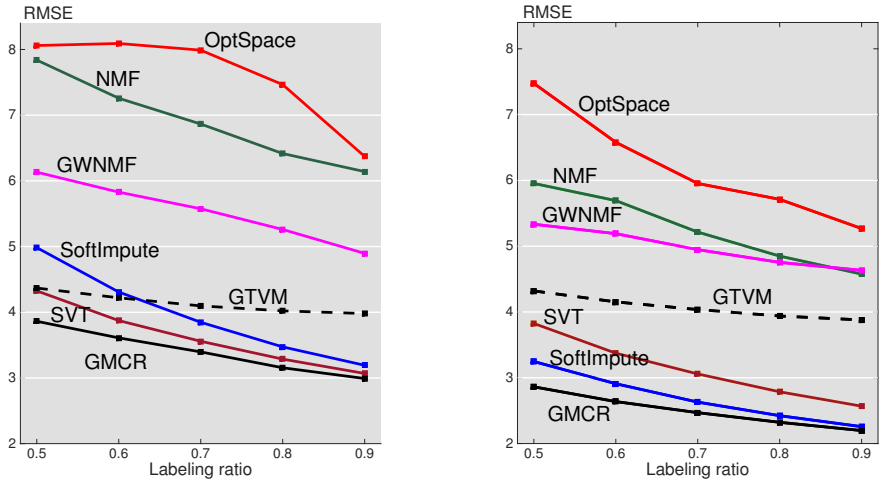
matrix completion methods to estimate the missing measurements. To further test the recovery algorithms with different amount of data, we randomly pick 50 out of 365 days of recording and conduct the same experiment for 10 times. In this case, we have a graph signal matrix with $N = 150$, and $L = 50$, or $L = 365$.

Figures 8.3 and 8.4 show RMSEs and MAEs for estimated temperature values averaged over 10 tests for each labeling ratio. We see that GTVM, as a pure graph-based method (8.21), performs well when the labeling ratio is low. When the labeling ratio increases, the performance of GTVM does not improve as much as the matrix completion algorithms, because it cannot learn from the graph signals. For both evaluation scores, RMSE and MAE, GMCR outperforms all matrix completion algorithms because it combines the prior information on graph structure with the low-rank assumption to perform a twofold learning scheme.

Rating completion for recommender system

We consider another matrix completion problem in the context of recommender systems based on the Jester dataset 1 [158]. The task is to predict missing ratings in an incomplete user-joke rating matrix where each column corresponds to the ratings of all the jokes from each user. Since the number of users is large compared to the number of jokes, following the protocol in [164], we randomly select 500 users for comparison purposes. For each user, we extract two ratings at random as test data for 10 tests. In this case, we have a graph signal matrix with $N = 100$, and $L = 500$.

Figures 8.5 and 8.6 show RMSEs and MAEs, defined in the evaluation score section, for estimated temperature values averaged over 10 tests. We see that graph-based methods (GWNMF and GMCR) take the advantage of exploiting the internal information of users and achieve smaller error. For RMSE, GMCR provides the best performance; for MAE, GWNMF provides the best performance.



(a) 50 recordings. (b) 365 recordings.

Figure 8.3: RMSE of temperature estimation for 50 recordings and 365 recordings.

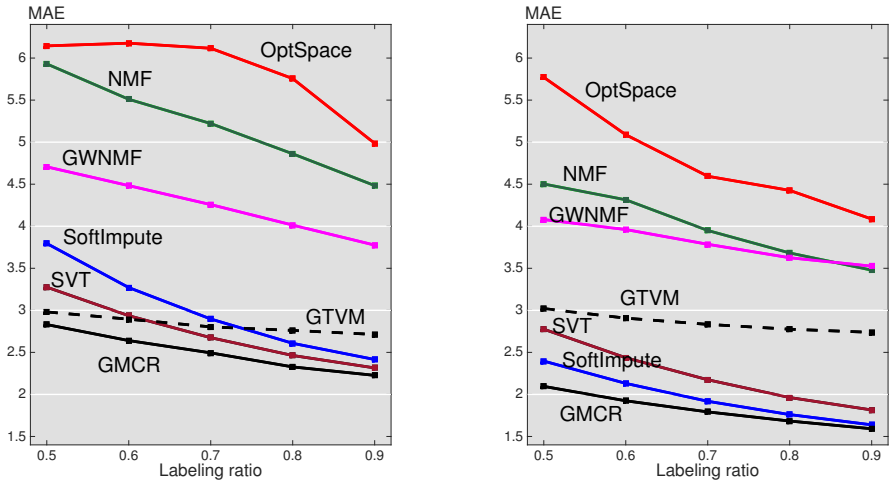
Combining expert opinions

In many real-world classification problems, the opinion of experts determines the ground truth. At times, these are hard to obtain; for instance, when a dataset is too large, obtaining the opinion of experts is too expensive, or experts differ among themselves, which happens, for example, in biomedical image classification [165]. In this case, a popular solution is to use multiple experts, or classifiers, to label dataset elements and then combine their opinions into the final estimate of the ground truth [166]. As we demonstrate here, opinion combining can be formulated and solved as graph signal matrix denoising.

We consider the online blog classification problem. We hide the ground truth and simulate $K = 100$ experts labeling 1224 blogs. Each expert labels each blog as conservative $(+1)$ or liberal (-1) to produce an opinion vector $\mathbf{t}_k \in \{+1, -1\}^{1224}$. Note that labeling mistakes are possible. We combine opinions from all the experts and form an opinion matrix $\mathrm{T} \in \{+1, -1\}^{1224\times 100}$, whose kth column is $\mathbf{t}_k$. We think of T as a graph signal matrix with noise that represents the experts' errors. We assume some blogs are harder to classify than others (for instance, the content in a blog is ambiguous, which is hard to label), we split the dataset of all the blogs into "easy" and "hard" blogs and assume that there is a 90% chance that an expert classifies an "easy" blog correctly and only a 30% chance that an expert classifies a "hard" blog correctly. We consider four cases of "easy" blogs making up 55%, 65%, 75%, and 85% of the entire dataset.

A baseline solution is to average (AVG) all the experts opinions into vector $\mathbf{t}_{\text{avg}} = (\sum_k \mathbf{t}_k)/K$ and then use the sign $\text{sign}(\mathbf{t}_{\text{avg}})$ vector as the labels to blogs. We compare the baseline solution with the GTVR solution (8.22) and GMCR. In GTVR, we first denoise every signal $\mathbf{t}_k$ and then compute the average of denoised signals $\widetilde{\mathbf{t}}_{\text{avg}} = (\sum_k \widetilde{\mathbf{t}}_k)/K$ and use $\text{sign}(\widetilde{\mathbf{t}}_{\text{avg}})$ as labels to blogs.

Using the proposed methods, we obtain a denoised opinion matrix. We average the opinions from all the experts into a vector and use its signs as the labels to blogs. Note that, for GTVR



(a) 50 recordings. (b) 365 recordings.

Figure 8.4: MAE of temperature estimation for 50 recordings and 365 recordings.

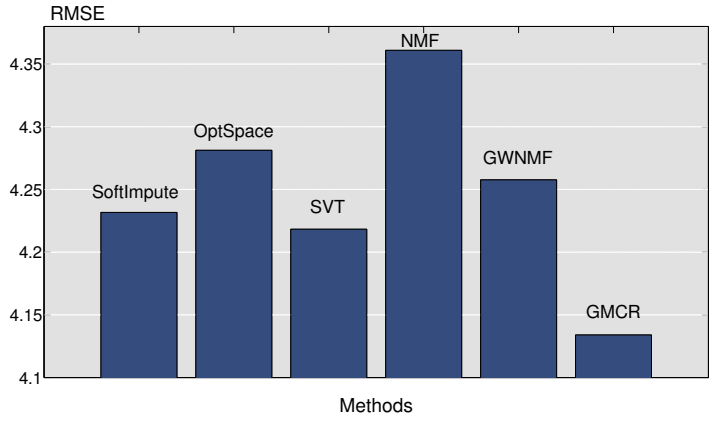


Figure 8.5: RMSE of the rating completion in Jester 1 dataset.

and GMCR, the accessible part is all the indices in the opinion matrix T; since each entry in T can be wrong, no ground-truth is available for cross-validation. We vary the tuning parameter and report the best results. Figure 8.7 shows the accuracy of estimating the ground-truth. We see that, through promoting the smoothness in each column, GTVR improves the accuracy; GMCR provides the best results because of its twofold learning scheme. Note that Figure 8.7 does not show that the common matrix completion algorithms provide the same "denoised" results as the baseline solution.

Applications of robust graph signal inpainting. We now apply the proposed robust graph signal inpainting algorithm to online blog classification and bridge condition identification. In contrast to what is done in the applications of graph signal inpainting, we manually add some outliers to the accessible part and compare the algorithm to common graph signal inpainting algorithms.

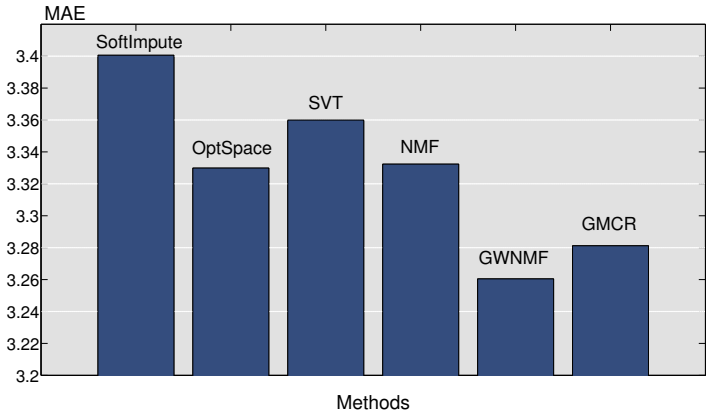


Figure 8.6: MAE of the rating completion in Jester 1 dataset.

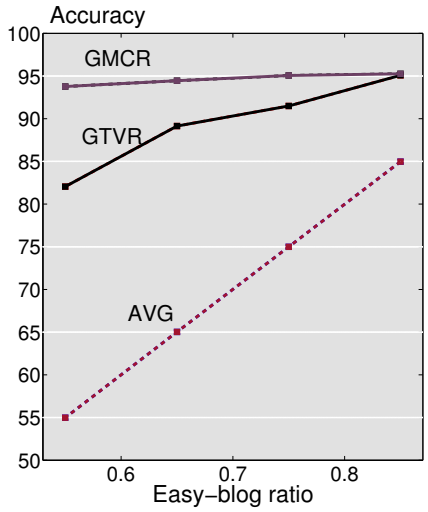


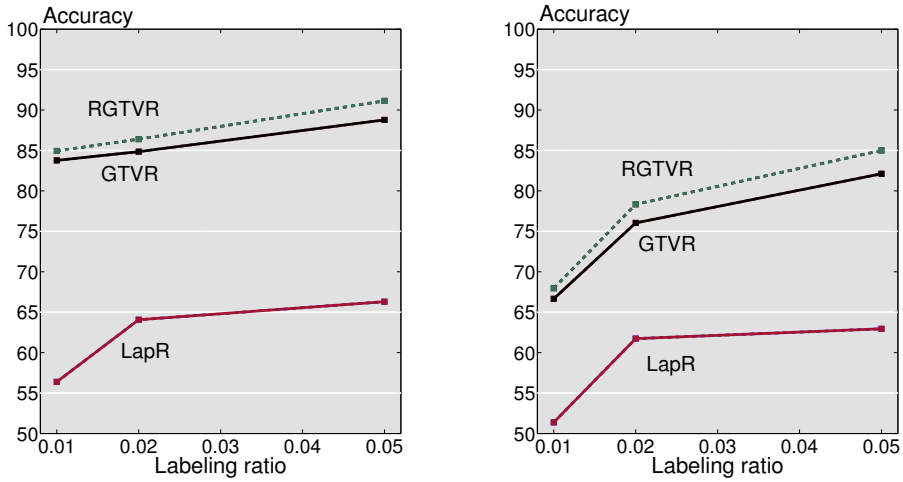
Figure 8.7: Accuracy of combining expert opinions.

Online blog classification

We consider semi-supervised online blog classification as described in Section 8.4. To validate the robustness of detecting outliers, we randomly mislabel a fraction of the labeled blogs, feed them into the classifiers together with correctly labeled signals, and compare the fault tolerances of the algorithms. Figure 8.8 shows the classification accuracies when $1\%, 2\%$, and 5% of blogs are labeled, with 16.66% and 33.33% of these labeled blogs mislabeled in each labeling ratio. We see that, in each case, RGTVR provides the most accurate classification.

Bridge condition identification

We consider a semi-supervised regression problem and adopt the dataset of acceleration signals as described in Section 8.4. To validate the robustness of facing outliers, we randomly mislabel a fraction of labeled acceleration signals, feed them into the graph signal inpainting algorithm



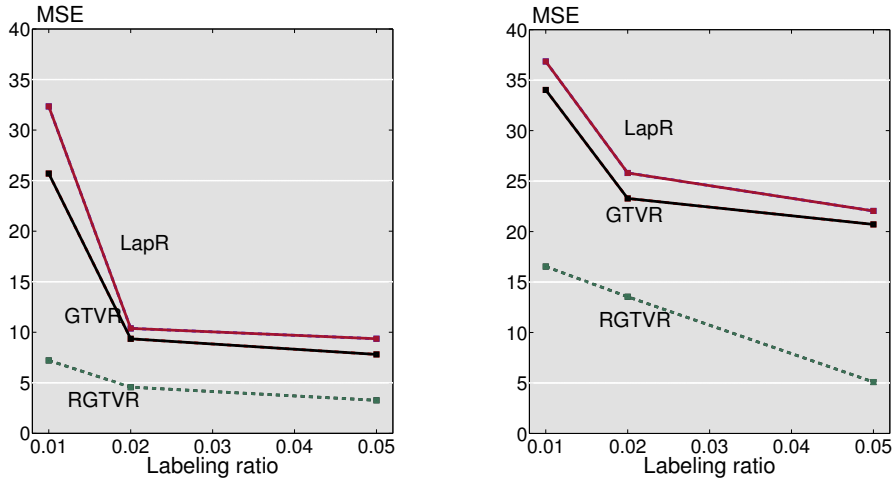
(a) mislabeled ratio: 16.67%. (b) mislabeled ratio: 33.33%.

Figure 8.8: Robustness to mislabeled blogs: accuracy comparison with labeling ratio of $1\%, 2\%$ and 5%, and mislabeling ratio of 16.66% and 33.33% in each labeling ratio.

together with correctly labeled acceleration signals, and compare the fault tolerances of the algorithms. Figure 8.9 shows MSEs when $1\%, 2\%$, and 5% of signals are labeled, with 16.66% and 33.33% of these labeled signals mislabeled in each labeling ratio. We see that, in each case, RGTVR provides the smallest error.

8.5 Conclusions

In this chapter, we formulated graph signal recovery as an optimization problem and provided a general solution by using the alternating direction method of multipliers. We showed that several existing recovery problems, including signal inpainting, matrix completion, and robust principal component analysis, are related to the proposed graph signal recovery problem. We further considered three subproblems, including graph signal inpainting, graph signal matrix completion, and anomaly detection of graph signals. For each subproblem, we provided specific solutions and theoretical analysis. Finally, we validated the proposed methods on real-world recovery problems, including online blog classification, bridge condition identification, temperature estimation, recommender system, and expert opinion combination of online blog classification.



(a) 16.67%. (b) 33.33%.

Figure 8.9: Robustness to mislabeled signals: MSE comparison with labeling ratio of $1\%, 2\%$, and 5%, and mislabeling ratio of 16.66% and 33.33% in each labeling ratio.

Part IV

Detection and Localization of Graph Signals

Overview of Detection and Localization

Detecting and localizing targeted patterns in data are vital tasks with numerous high-impact applications, especially in a big data era. The massive amount of data are generated from various sources, including online social networks, citation networks, biological networks, physical infrastructures and many others. For example, Google, Yahoo!, Microsoft, and other Internet-based companies have data that is measured in exabytes (10^{18} bytes). Social media, such as Facebook, YouTube and Twitter, have exploded beyond anyone's wildest imagination [167]. It makes pattern discovery and detection more challenging and meaningful.

Detection is not a new topic in the signal processing community. The task of finding activated supports of signals/images has been intensely studied in classical signal/image processing from various aspects over the past few decades. For example, impulse detection considers localizing impulses in a noisy signal [3]; change-point detection identifies times when the probability distribution of a stochastic process or time series changes [4]; support recovery of sparse signals localizes sparse activations with a limited number of samples [5, 6]; foreground detection localizes foreground in a video sequence [7]; cell detection and segmentation localizes cells in microscopy images [8] and matched filtering localizes radar signals in the presence of additive stochastic noise [9, 10]. Needless to say, detection is significant; however, detection is not well defined in many tasks. For example, it is debatable to provide a precise definition for anomalies in anomaly detection and communities in community detection.

In this part, we consider detection and localization of graph signals. In general, graph signal detection is to identify whether or not a graph signal contains a structure-related pattern, and graph signal localization is to find the exact supports of a structure-related pattern. They are relevant to many applications including identifying clustered attributes in social networks, special events in the urban traffic networks, unusual brain activity in the brain connectivity networks, and viruses in cyber-physical systems.

Mathematically, we consider a class of graph signals with structure-related patterns as

$$\mathcal{X}_K = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{x} = \mathrm{D}\,\mathbf{a}, \mathbf{a} \in \mathbb{R}^S, \|\mathbf{a}\|_0 \leq K\},$$

where $\mathrm{D} \in \mathbb{R}^{N \times S}$ is a graph dictionary where each atom represents a structure-related pattern. Given a noisy observation of graph signal $\mathbf{y} \in \mathbb{R}^N$, the goal of detection is to test the null against the alternative hypotheses:

$$\begin{aligned} H_0 &: \mathbf{y} \sim f(0, \epsilon), \\ H_1 &: \mathbf{y} \sim f(\mathbf{x}, \epsilon) \text{ with } \mathbf{x} \in \mathcal{X}_K. \end{aligned}$$

where ϵ is noise and $f(\cdot, \cdot)$ is a link function. For example, $f(\mathbf{x}, \epsilon) = \mathbf{x} + \epsilon \in \mathbb{R}^N$ is a deterministic linear mapping; $f(\mathbf{x}, \epsilon) = \mathrm{Bernoulli}(\mathbf{x} + \epsilon) \in \mathbb{R}^N$ is a stochastic mapping, where each element $f(\mathbf{x}, \epsilon)_i$ is a Bernoulli random variable with mean $(\mathbf{x} + \epsilon)_i$. Let this test be a mapping $T(\mathbf{y}) = \{0, 1\}$, where 1 indicates that we reject the null. It is imperative that we control both the probability of false alarm, and the false acceptance of the null. To this end, we define the detection risk to be

$$R(T) = \mathbb{E}_0[T] + \sup_{\mathbf{x} \in \mathcal{X}_k} \mathbb{E}_{\mathbf{x}}[1 - T],$$

where $\mathbb{E}_{\mathbf{x}}$ denotes the expectation with respect to $\mathbf{y} \sim \text{Bernoulli}(\mathbf{x})$. These terms are also known as the probability of type 1 and type 2 error respectively. We will say that H_0 and H_1 are asymptotically distinguishable by a test T, if $\lim_{N\in\infty} R(T) = 0$, where N is the number of nodes in a graph. If such a test exists then H_0 and H_1 are asymptotically distinguishable, otherwise they are asymptotically indistinguishable.

Given a noisy observation of graph signal $\mathbf{y} = f(\mathbf{x}, \epsilon) \in \mathbb{R}^N$, where $\mathbf{x} = \sum_{i=1}^{K} a_{\Omega_i} \mathbf{d}_{\Omega_i} \in \mathcal{X}_K$, the goal of localization is to recover each activated atom, $\mathbf{d}_{\Omega_i}$.

The prototype of designing localization strategies is to solve the following optimization problem

$$\begin{aligned} \mathbf{a}^* \;=\; \arg\min_{\|\mathbf{a}\|_0 \le K} \quad & d(\mathbf{x}', \mathbf{y}), \\ \text{subject to} \quad & \mathbf{x}' = \mathrm{D}\,\mathbf{a}, \end{aligned}$$

where $d(\cdot, \cdot)$ is some evaluation metric. For example, when D is an identity matrix, we localize impulses on graphs; when D is the graph Fourier basis, we localize activation in the graph Fourier domain. This formulation is similar to graph signal approximation; however, the goal is slightly different: localization aims to identify which atoms are activated (recover $\mathbf{a}$), and approximation aims to reconstruct the original graph signal (recover $\mathrm{D}\,\mathbf{a}$). When the size of graph dictionary grows exponentially with the size of graph, the graph dictionary cannot be explicitly expressed, which makes precise localization challenging.

Chapter 9

Detection of Localized Graph Signals

9.1 Introduction

In this chapter, we consider detecting *localized categorical attributes* on a graph. A categorical attribute is defined as a variable that can be put into a countable number of categories or groups. It can be represented by several binary attributes and is widely used in numerous data and graph mining applications [31]. We model a categorical attribute by binary graph signals[1]: when a signal coefficient is one, the corresponding node is activated by the attribute, and vice versa. A localized categorical attribute, or a localized pattern[2], is defined as an attribute whose activated nodes form a subgraph that can be easily separated from the rest of nodes. In practice, detecting a localized pattern is nontrivial because an observed attribute is often corrupted by noise. The goal of localized pattern detection is to identify the localized pattern hidden in a noisy attribute with the aid of graph topology. This task is relevant to many real-world applications including identifying localized attributes in online social networks, activity in the brain connectivity networks and viruses in cyber-physical systems.

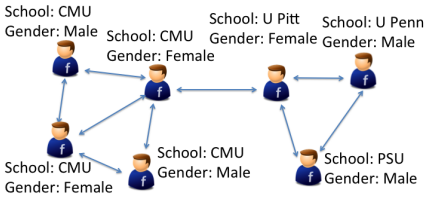
In classical signal processing, a localized signal is constant over local connected regions separated by lower-dimensional boundaries. It is often related to concepts such as impulse function, step function, square wave and Haar basis [84]. Detecting localized signals has been considered from many aspects, such as signal/noise discrimination [168], edge detection [169], pattern matching [10] and support recovery of sparse signals [5, 6]. This paper considers a counterpart problem on graphs. A localized graph signal is constant over a subgraph that is easily separated from the rest of the nodes. Similarly to classical localized signals, a localized graph signal emphasizes fast transitions (corresponding to boundaries) and localization on the graph vertex domain (corresponding to signals being nonzero in a local neighborhood).

This localized pattern detection task is related, yet different from conventional community detection in network science [28, 170, 171]. The goal of community detection is to identify modules and hierarchical organization by using only the information encoded in the graph topology [172]. While a module is vaguely defined, it is usually considered as a node set with dense internal and sparse external connections. The difference between the two is that community de-

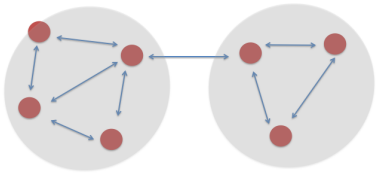
[1] Graph signals and attributes are same in this context.

[2] A localized pattern is used to represent a noiseless localized attribute.

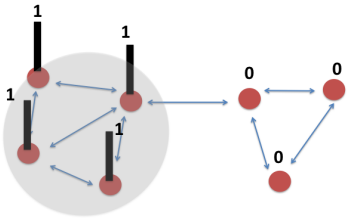
tection considers detecting patterns in graph topology while localized pattern detection considers detecting patterns in an attribute. For example, suppose that we want to identify whether users from Carnegie Mellon University form a localized pattern on Facebook. The binary answer to *Is this user from Carnegie Mellon University?* is a binary attribute on a graph, which activates a subgraph with few external connections. We thus consider that activated users form a localized pattern; see Figure 9.1.



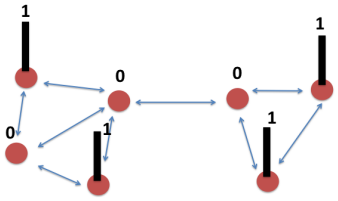
(a) Attribute graph.



(b) Two communities.



(c) Binary attribute *from CMU?*.



(c) Binary attribute *male?*.

Figure 9.1: Detecting localized categorical attributes. Plot (a) shows a graph with two attributes; Plot (b) shows two communities in the graph; Plot (c) shows the attribute *Is this user from Carnegie Mellon University?* forms a localized graph signal; and Plot (d) shows the attribute *Is this user male?* forms a nonlocalized graph signal. The goal of community detection is to identify subgraphs as shown in Plot (b), but the goal of the localized pattern detection is to identify if a binary attribute is localized as shown in Plots (c) and (d).

To describe the localization level of an attribute, we consider external rather than internal connections of the subgraph activated by the attribute, leading to scalable detection algorithms. Specifically, the localization level of a localized pattern is defined as the difficulty of separating the corresponding subgraph from the rest of the nodes, which is quantified by the total variation on graphs. We then formulate hypothesis tests to decide whether a categorical attribute is localized. We propose two statistics: graph wavelet statistic and graph scan statistic. Similarly to detecting transient changes in time-series signals by using wavelet techniques, we design a graph wavelet statistic based on a Haar-like graph wavelet basis. Since the graph wavelet basis is preconstructed, the computational cost is linear with the number of nodes. We also formulate a generalized likelihood test and propose a graph scan statistic, which can be efficiently solved

by a standard graph-cut algorithm. The intuition behind the proposed statistics is to find the underlying localized pattern in a graph, which is equivalent to denoising the given attribute based on the graph structure, and then calculating the statistic values based on the denoised attribute. We validate the proposed hypothesis tests on both simulated and real data. Experimental results demonstrate the effectiveness and robustness of the proposed methods.

This detection problem shares similar characteristics to many detection problems in current graph-related literature, such as detecting a smooth graph signal or a localized graph signal under a specific noise model. For example, [173, 174] detects a cluster in a lattice graph that exhibits unusual behavior; [175] constructs a generalized likelihood test to detect smooth graph signals; [176] considers a general graph-structured normal means test; in [78], the uniform spanning tree wavelet statistic is constructed to approximate epsilon scan statistic; and in [177, 178], the Lovasz extended scan statistic and spectral relaxation are considered as the relaxations of the combinatorial scan statistic.

The uniform spanning tree wavelet statistic and the Lovasz extended scan statistic lay a foundation for this paper. Here, we extend the noise model from the Gaussian to the Bernoulli model; that is, we deal with binary values instead of real values. This extension is needed because categorical attributes are widely used in numerous tasks in data/graph mining. Intuitively, we can treat our input, a binary value, as the outcome of a real value after hard thresholding. Because hard thresholding causes information loss, real-valued attributes are more informative than binary-valued attributes; in other words, handling binary-valued attributes is a nontrivial task.

Our detection problem is also related to community detection. As one of the key topics in network science and graph mining, community detection aims to extract tightly connected subgraphs in networks, also known as graph partitioning and graph clustering [22, 28, 179]. The traditional community detection algorithms focus on the graph structure only [170, 180]. Some recent studies tried to combine the knowledge of both graph structure and node attributes [181]. Useful node attributes not only improve the accuracy in community detection, but also provide the interpretation of detected communities. However, as some attributes may be irrelevant for some communities, community detection accuracy may suffer. It is also computational inefficient to include a large number of attributes in the training phase [182]. In the paper, we aim to find useful attributes for improving community detection and interpretation.

9.2 Problem Formulation

In the task of detecting a localized graph signal, we aim to identify whether a noisy graph signal has a localized pattern. Mathematically, given a noisy binary graph signal (or attribute) $\mathbf{y}$, the general statistical testing formulation is:

$$\begin{aligned} H_0 &: \mathbf{y} \sim f(0, \epsilon_0), \\ H_1 &: \mathbf{y} \sim f(\mathbf{s}, \epsilon) \text{ with } \mathbf{s} \in \mathcal{S}, \end{aligned} \tag{9.1}$$

where $\mathcal{S}$ is a predefined class of localized patterns, $f(\cdot, \cdot)$ is a link function and $0 < \epsilon,\ \epsilon_0 < 1$ are noise levels. The null hypothesis represents a scattered pattern and the alternative hypothesis represents a localized pattern. Two key factors about the statistical testing (9.1) are the noise

model and the localized pattern, and we can make independent assumptions on these two. For example, noise can follow Gaussian or Bernoulli distribution and the localization level can be described by small cut costs or cliques [26].

Let this test be a mapping $T(\mathbf{y}) = \{0, 1\}$, where 1 indicates that we reject the null hypothesis. It is imperative that we control both the probability of false alarm and the false acceptance of the null hypothesis. To this end, we define the risk to be

$$R(T) = \mathbb{E}_0[T] + \sup_{\mathbf{s}\in\mathcal{S}} \mathbb{E}_{\mathbf{s}}[1 - T],$$

where $\mathbb{E}$ denotes the expectation. These two terms are also known as the probability of type-1 and type-2 errors, respectively. We will say that H_0 and H_1 are asymptotically distinguishable by a test T, if $\lim_{N\to\infty} R(T) = 0$, where N is the number of nodes. In other words, when the number of nodes goes to infinity and the detection risk goes to zero, H_0 and H_1 are asymptotically distinguishable; otherwise they are asymptotically indistinguishable.

Bernoulli noise model. In this paper, we are particularly concerned with a specific version of (9.1):

$$\begin{aligned} H_0 &: \mathbf{y} \sim \text{Bernoulli}(\epsilon_0 \mathbf{1}_{\mathcal{V}}), \\ H_1 &: \mathbf{y} \sim \text{Bernoulli}(\mu \mathbf{1}_C + \epsilon \mathbf{1}_{\overline{C}}) \text{ for all } \|\Delta \mathbf{1}_C\|_p \leq \rho, \end{aligned} \tag{9.2}$$

where μ is signal strength, ϵ is noise level, $\epsilon_0 = \epsilon + |C|(\mu - \epsilon)/N$, and ρ is the number of external edges of a localized pattern, which reflects the shapes of candidate localized patterns and characterizes the alternative hypothesis H_1. Here, the signals under both H_0 and H_1 have the same mean. The link function follows from a Bernoulli noise model

$$f(\mathbf{s}, \epsilon) = \text{Bernoulli}(\mathbf{s} + \epsilon \mathbf{1}_{\mathcal{V}}) \in \mathbb{R}^N,$$

where each element $f(\mathbf{s}, \epsilon)_i$ is a Bernoulli random variable with mean $(\mathbf{s} + \epsilon)_i$ and the class of localized graph signals is

$$\mathcal{S} = \left\{ \mathbf{s} : \mathbf{s} = (\mu - \epsilon)\mathbf{1}_C, C \in \mathcal{C} \right\}$$

with $\mathcal{C} = \left\{ C \subseteq \mathcal{V} : \|\Delta \mathbf{1}_C\|_p \leq \rho \right\}$, where $0 < \mu - \epsilon < 1$ controls the signal strength and ρ, p control the cut cost of the activated node set. The class $\mathcal{C}$ specifies the localized patterns that the user is testing for. Quantity ρ is a user-defined parameter: when ρ is large, we consider a relaxed scenario where all candidate localized patterns are allowed to have any number of external edges, and the test will always succeed; when ρ is small, we consider a constrained scenario where all candidate localized patterns have few external edges. Note that the Bernoulli model here is similar to the setting in community detection with categorical attributes [31, 181]. For example, suppose that we want to identify whether users who graduated from Carnegie Mellon University form social communities on Facebook. The binary value *Is this user from Carnegie Mellon University?* is an attribute on Facebook and forms a binary graph signal. When this attribute leads to a community, we should find a subgraph such that (1) most nodes are activated within the subgraph and few nodes are activated outside the subgraph; (2) the connection between this subgraph and its complement is weak. We describe a binary attribute by the Bernoulli noise model and a localized pattern by a graph signal with small total-variation.

9.3 Methodology

In this section, we propose two statistics for hypothesis testing (9.2): graph wavelet statistic and graph scan statistic. The first statistic is based on a graph wavelet basis; when a given attribute has large graph wavelet coefficients, this attribute agrees with the graph structure and is localized. The second statistic is based on matching all the possible node sets to a given attribute via solving an optimization problem; when we find such a feasible node set, this attribute is localized. The first method is more efficient as the graph wavelet basis is pre-constructed, while the second is more accurate as we adaptively search for localized patterns.

9.3.1 Graph Wavelet Statistic

In classical signal processing, we detect transient changes in a time-series signal by projecting the signal on the wavelet basis [84]. When a coefficient in a high-frequency band is large, there exists a transient change. Similarly, we construct a Haar-like graph wavelet basis to detect the boundary between the activated nodes and nonactivated nodes. When the boundary exists, we reject the null hypothesis.

The construction of the graph wavelet basis is shown in Chapter 4. The main idea is to recursively partition a local parent set into two disjoint local child sets with similar sizes. Each partition generates two basis vectors that represent the two local child sets. We start from the entire node set $\mathcal{V}$, corresponding to the coarsest resolution in the graph vertex domain. In the end, each local set is either an individual node or an empty set, corresponding to the finest resolution in the graph vertex domain. The construction of the graph wavelet basis is similar in spirit to the construction of the standard Haar wavelet basis, which is generated by using a general dyadic splitting scheme [183]. We simply split the graph into two subgraphs with roughly equal sizes, irrespective of the connections between them; we do not assume that the graph contains communities with balanced sizes. This construction is multiresolution in nature, as the graph wavelet basis vectors have self-similarity relations and have supports of different sizes.

To ensure the detection property of the graph wavelet basis, we impose three requirements on each partition: (1) the two local child sets are disjoint; (2) the union of two local child sets is the local parent set; and (3) the cardinalities of the two local child sets are as close as possible. The first two requirements lead to the orthogonality of the graph wavelet basis and the third requirement promotes the sparsity for all graph signals with small ℓ_0-norm-based total variation. In general, any algorithm that satisfies these three requirements can be used to generate a graph wavelet basis; [93] introduces three such algorithms. Theorem 9 will show that such basis has similar detection performance.

Recall that Theorem 7 shows the graph wavelet basis is orthornormal shows that the graph wavelet basis preserves the energy of any input graph signal. Theorem 8 shows that the total decomposition level T is crucial for the upper bound on sparsity, which corresponds to the worst case in a sparse representation. We consider the worst case because the graph wavelet basis is constructed before obtaining any graph signal and we need to ensure that the graph it works for arbitrary graph signals. Wavelet representation pushes the energy of a graph signal into a few wavelet coefficients. Let W denote the graph wavelet basis (we use $\mathrm{W}_{\mathrm{LSPC}}$ in Chapter 4). We

have

$$\left\|\mathrm{W}_{(-1)}^T\,\mathbf{y}\right\|_\infty^2 \overset{(a)}{\geq} \frac{\left\|\mathrm{W}_{(-1)}^T\,\mathbf{y}\right\|_2^2}{\left\|\mathrm{W}_{(-1)}^T\,\mathbf{y}\right\|_0} \overset{(b)}{\geq} \frac{\|\mathbf{y}\|_2^2 - \left(\frac{1}{\sqrt{N}}\mathbf{1}_{\mathcal{V}}^T\mathbf{y}\right)^2}{1+\|\Delta\mathbf{y}\|_0\,T},$$

where $\mathrm{W}_{(-1)} \in \mathbb{R}^{N\times(N-1)}$ is W without the first constant column. The first constant column is removed since it only calculates the mean of $\mathbf{y}$, which is not informative for detection. The inequality (a) follows from the basic norm inequality, and (b) from Theorems 7 and 8 will show that the largest nontrivial wavelet coefficient is an important metric in distinguishing whether $\mathbf{y}$ is a localized attribute or not. We see that the lower bound on the largest nontrivial wavelet coefficient is related to the total decomposition level. To lift the largest nontrivial wavelet coefficient up, we need to minimize the total decomposition level T, which, in turn, is minimized when we partition each local set evenly, with $T = O(\log_2 N)$.

We use the graph wavelet basis to detect the localized patterns; similarly to classical image processing, the graph wavelet basis is only activated by the boundaries in a graph signal. For a localized graph signal with a small ℓ_0-norm-based total variation, the corresponding graph wavelet coefficients are sparse and the energy of the original graph signal concentrates in a few graph wavelet coefficients. However, for a noisy graph signal with a large ℓ_0-norm-based total variation, the energy of the original graph signal spreads over all the graph wavelet coefficients.

Each graph wavelet basis vector compares the absolute difference between the average values of the input graph signal in two local sets. For example, $\mathbf{y}^T\left(\mathbf{1}_{S_1}/|S_1| - \mathbf{1}_{S_2}/|S_2|\right)$ is the difference of the average values of $\mathbf{y}$ in the node sets S_1 and S_2. When one local set captures significantly larger average value than the other local set, that local set detects an activated region. Because of the multiresolution construction, the graph wavelet basis searches for activated regions of various sizes. Thus, the maximum value of the graph wavelet coefficient dentifies whether the original graph signal has a structure-correlated pattern.

Mathematically, given a noisy observation $\mathbf{y}$, the *graph wavelet statistic* is the maximum absolute value of the nontrivial graph wavelet coefficients,

$$\widehat{w} = \left\|\mathrm{W}_{(-1)}^T\,\mathbf{y}\right\|_\infty. \tag{9.3}$$

Remember that we remove the first constant column from W whose corresponding wavelet coefficient is trivially the mean of $\mathbf{y}$. When $\widehat{w}$ is larger than some threshold, we detect the activated node set and reject the null hypothesis. To analyze the detection error, we use the following theorem.

Theorem 21. Let the graph wavelet statistic be $\widehat{w}$ in (9.3) and let

$$\mu - \epsilon \geq \frac{\sqrt{1+\rho\log N}\left(\sqrt{\log N} + 2\sqrt{2\log(\frac{2}{\delta})}\right)}{\sqrt{|C|\left(1-\frac{|C|}{N}\right)}}. \tag{9.4}$$

Under the statistical test (9.2) with $p = 0$, rejecting the null hypothesis for all $\widehat{w} > \tau$, with $\tau = \sqrt{\log N} + \sqrt{2\log(2/\delta)}$, implies that, under H_0, $\mathbb{P}\{\text{reject}\} \leq \delta$, and under H_1, $\mathbb{P}\{\text{reject}\} \geq (1-\delta)^4$.

See the proof of Theorem 21 in Appendix 14.5. The main idea is to show that under the null hypothesis, each graph wavelet coefficient is a sub-Gaussian random variable, which means that its statistical performance is similar to a Gaussian random variable [184]; and under the alternative hypothesis, the maximum value of the graph wavelet coefficients is large because the energy of the original graph signal concentrates in a few graph wavelet coefficients.

Since the distribution of graph wavelet statistic does not have an analytical form, it is hard to calculate the exact p-value. Instead, we can use the sub-Gaussianity to provide an upper bound of the p-value. Given graph wavelet statistic $\widehat{w} = \left\|\mathrm{W}_{(-1)}^{T}\,\mathbf{y}\right\|_{\infty}$, the upper bound of p-value is $e^{-\frac{1}{2}\left(\widehat{w}-\sqrt{\log N}\right)^2}$. Let our test be level α. When α is larger than the upper bound of p-value, α is definitely larger than the exact p-value. Thus, we reject the null hypothesis for all $\alpha \geq e^{-\frac{1}{2}\left(\widehat{w}-\sqrt{\log N}\right)^2}$. The threshold is only related to the size of graph N.

The assumption (9.4) shows that the key to detect the activation is the difference between μ and ϵ. Such a difference is related to the properties of the ground-truth activated node set. When the size of the ground-truth activated node set is larger and the ground-truth activated node set has a small ℓ_0-norm-based total variation, it is easier for graph wavelet statistic to detect the activation.

Corollary 9. H_0 and H_1 are asymptotically distinguishable by the graph wavelet statistic when

$$\sqrt{|C|}(\mu - \epsilon) \;=\; O\left(\sqrt{\rho}\log N\right).$$

The sufficient condition is $|C| \geq O(\rho \log^2 N)$ assuming μ, ϵ constant.

Corollary 9 builds a mathematical relationship between the size of a localized pattern and asymptotic distinguishability; it is easier to detect a larger localized pattern with small ρ. The cut cost ρ is the number of external edges of a localized pattern, which determines the shape of candidate localized patterns we are testing for. When ρ is small, we consider a constrained scenario where all candidate localized patterns have few external edges. When ρ is large, we need to search over more candidate localized patterns and requires greater $|C|$ to increase the signal-to-noise ratio, which is $(\mu - \epsilon)$ here.

The computational bottleneck of constructing the graph wavelet basis is the graph partition algorithm as shown in Figure 4.2. Let the computational cost of the graph partition algorithm be of the order $O(h(N))$, where $h(\cdot)$ is a polynomial function. The total computational cost to construct a graph wavelet basis behaves as $O\left(\sum_{i=0}^{\log N} 2^i h(N/2^i)\right)$; for example, when the cost of a graph partition algorithm is of the order $O(N \log N)$, the total computational cost to construct a graph wavelet basis behaves as $O(N \log^2 N)$. Since the graph wavelet basis is constructed only based on the graph structure, the construction is performed only once and works for any graph signal supported on this graph. The total computational cost to obtain the graph wavelet statistic only involves a matrix-vector multiplication and a search for the maximum value. The graph wavelet statistic is thus scalable to large-scale graphs.

9.3.2 Graph Scan Statistic

Previously, we construct a graph wavelet statistic to test if a given graph signal has a structure-correlated pattern. The graph wavelet statistic is efficient, but the construction of the graph

wavelet basis is independent from the given graph signal. Now we aim to propose a data-adaptive approach, which scans all feasible node sets based on the given graph signal.

For the simplicity of derivation, we assume that $\epsilon_0 = \epsilon$ in (9.2). This implies that the average value under H_1 is larger than the average value under H_0. A naive approach is to use the average of the observation as the statistic. Here we propose a graph scan statistic that works better than this naive approach. The intuition behind the proposed statistics is that given the noisy observation y, we search for the activated node set C. If we can find such C, we reject the null hypothesis, and vice versa.

If we knew the true activated node set $C \in \mathcal{C}$, an intuitive idea is to test the null hypothesis $H_0 : \mathbf{s} = 0$ against the alternative $H_1 : \mathbf{s} = \mu \mathbf{1}_C$ by using the likelihood ratio test. Given the observation $\mathbf{y}$, based on the Bernoulli noise model, the likelihood is

$$\mathbb{P}(\mathbf{y}|H_1) \;=\; \prod_{i\in C} \mu^{y_i}(1-\mu)^{1-y_i} \prod_{i\in \overline{C}} \epsilon^{y_i}(1-\epsilon)^{1-y_i}$$

and the maximum likelihood estimator is $\widehat{\mu} = \mathbf{1}_C^T \mathbf{y}/|C|$. The likelihood ratio is

$$\begin{aligned} & \frac{\prod_{i\in\mathcal{V}} \epsilon^{y_i}(1-\epsilon)^{1-y_i}}{\prod_{i\in C} \widehat{\mu}^{y_i}(1-\widehat{\mu})^{1-y_i} \prod_{i\in\overline{C}} \epsilon^{y_i}(1-\epsilon)^{1-y_i}} \\ = \;& \prod_{i\in C} \left(\frac{\epsilon}{\widehat{\mu}}\right)^{y_i} \left(\frac{1-\epsilon}{1-\widehat{\mu}}\right)^{1-y_i}. \end{aligned}$$

The log likelihood ratio is

$$\begin{aligned} & \sum_{i\in C} y_i \log\left(\frac{\epsilon}{\widehat{\mu}}\right) + \sum_{i\in C}(1-y_i)\log\left(\frac{1-\epsilon}{1-\widehat{\mu}}\right) \\ = \;& |C| \left(\widehat{\mu}\log\left(\frac{\epsilon}{\widehat{\mu}}\right) + (1-\widehat{\mu})\log\left(\frac{1-\epsilon}{1-\widehat{\mu}}\right)\right) \\ = \;& -|C|\,\mathrm{KL}(\widehat{\mu}||\epsilon). \end{aligned}$$

In practice, however, the true active node set C is unknown. In this case, we are concerned with the generalized likelihood ratio

$$\begin{aligned} \widehat{g} \;=\; & \max_C |C|\,\mathrm{KL}\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}||\epsilon\right) \\ & \text{subject to } \|\Delta \mathbf{1}_C\|_1 \le \rho. \end{aligned} \tag{9.5}$$

When $\mathbf{1}_C^T\mathbf{y}/|C|$ is much larger than the background randomness ϵ, the node set C is activated. We call $\widehat{g}$ *graph scan statistic*. This is the Bernoulli version of [177, 178]. To maximize the objective in (9.5), the activated region C should tradeoff between its size and the average value inside: when $|C|$ is large, the average value inside C tend to be small; on the other hand, when C fits the activated nodes in $\mathbf{y}$, the average value is large; however, due to the cut cost constraint, C can only fit a few scattered nodes and $|C|$ is small. The goal of the graph scan statistic is to search a node set with both large cardinality and large average value.

When $\widehat{g}$ is larger than some threshold, we detect the activated node set and reject the null hypothesis. To analyze the detection error, we have the following theorem.

Theorem 22. Let the graph scan statistic be $\widehat{g}$ in (9.5) and let

$$\begin{aligned}\mu - \epsilon \;\geq\; & \frac{2\sqrt{2}}{\sqrt{|C|}}\left(\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\right. \\ & \left.+\sqrt{2\log 2}+\sqrt{2\log(1/\delta)}\right)+\sqrt{\frac{\log(2/\delta)}{2|C|}}.\end{aligned} \tag{9.6}$$

Under the statistical test (9.2) with $p=1$, rejecting the null hypothesis for all $\widehat{g} > \tau$, with

$$\begin{aligned}\tau \;=\; & 8\left(\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\right. \\ & \left.+\sqrt{2\log 2}+\sqrt{2\log(1/\delta)}\right)^2,\end{aligned}$$

implies that, under H_0, $\mathbb{P}\{\text{reject}\} \leq \delta$, and under H_1, $\mathbb{P}\{\text{reject}\} \geq 1-\delta$.

See the proof of Theorem 22 in Appendix 14.6. The main idea is to show that under the null hypothesis, $\mathbf{1}_C^T(\mathbf{y}-\epsilon)/\sqrt{|C|}$ is a sub-Gaussian random variable; and under the alternative hypothesis, the maximum likelihood estimator $\mathbf{1}_C^T\mathbf{y}/|C|$ is close to μ with high probability. Similarly to the graph wavelet statistic, the assumption (9.6) shows that the key to detect the activation is related to the properties of the ground-truth activated node set. When the size of the ground-truth activated node set is larger and the ground-truth activated node set has a small ℓ_1-norm-based total variation, it is easier for graph scan statistic to detect the activation.

Similarly to the graph wavelet statistic, given graph scan statistic $\widehat{g}$, the upper bound of p-value is $e^{-\frac{1}{2}\left(\sqrt{\frac{\widehat{g}}{8}}-2\log 2-\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\right)^2}$. Let our test be level α. We reject the null hypothesis at all $\alpha \geq e^{-\frac{1}{2}\left(\sqrt{\frac{\widehat{g}}{8}}-2\log 2-\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\right)^2}$. The threshold is only related to the size of graph N and the cut cost ρ, which is a user-defined parameter.

There are two advantages of graph scan statistic over graph wavelet statistic: graph scan statistic is data adaptive and is flexible to consider edge weights. Instead of using a pre-constructed graph wavelet basis, graph scan statistic actively searches for the activated node set. Thus, graph scan statistic not only detects whether the structure-correlated activated node set exists, but also localizes these regions. Graph scan statistic also considers the edge weights by using the ℓ_1-norm-based total variation. It is more general compared to ℓ_0-norm-based total variation used in graph wavelet statistic. Note that the ℓ_1-norm-based total variation and the ℓ_0-norm-based total variation are the same when we only consider binary edge weights. Thus, all the results based on the ℓ_1-norm can be directly applied to the ℓ_0-norm.

Corollary 10. H_0 and H_1 are asymptotically distinguishable by the graph scan statistic when

$$\sqrt{|C|}(\mu-\epsilon) \;=\; O\left(\max(\sqrt{\rho},\sqrt{\log N})\sqrt{\log N}\right).$$

The sufficient condition is $|C| \geq O(\max(\rho, \log N)\log N)$ assuming μ, ϵ constant.

As mentioned earlier, a naive statistic is the average value of the observation. The naive statistic requires $|C| \geq O(\sqrt{N})$, which is asymptotically much worse than the proposed graph scan statistic. Similarly to the graph wavelet statistic, for the graph scan statistic, it is easier to detect a larger localized pattern with small ρ.

Practical algorithms. In the previous analysis, we used the global optimum of (9.5), $\widehat{g}$; however, we will show that this global optimum is hard to obtain because the optimization problem is combinatorial. We consider two practical methods to compute the graph scan statistic. The first method obtains a local optimum of the original optimization problem and the second method obtains a global optimum of a relaxed optimization problem.

In the first method, we reformulate (9.5) and solve

$$\begin{aligned} \widehat{g} &= \max_t \max_{\mathbf{x}} t \, \mathrm{KL}\left(\frac{\mathbf{x}^T\mathbf{y}}{t} || \epsilon\right) \\ &\quad \text{subject to } \mathbf{x} \in \{0,1\}^N, \|\Delta\mathbf{x}\|_1 \leq \rho, \mathbf{1}^T\mathbf{x} \leq t, \end{aligned} \tag{9.7}$$

where $\mathbf{x}$ is an auxiliary graph signal to denote $\mathbf{1}_C$ and t denotes $|C|$. Since ϵ is a small constant, for each t, we optimize over $\mathbf{x}$ to make $\mathbf{x}^T\mathbf{y}/t$ as far away from ϵ as possible, which is equivalent to maximizing $\mathbf{x}^T\mathbf{y}$ within the feasible region.[3] We thus solve

$$\begin{aligned} \mathbf{x}_t^* &= \arg\min_{\mathbf{x}} -\mathbf{x}^T\mathbf{y}, \\ &\quad \text{subject to } \mathbf{x} \in \{0,1\}^N, \|\Delta\mathbf{x}\|_1 \leq \rho, \mathbf{1}^T\mathbf{x} \leq t. \end{aligned} \tag{9.8}$$

The corresponding Lagrangian function is

$$L(\eta_1, \eta_2, \mathbf{x}) = -\mathbf{x}^T\mathbf{y} + \eta_1(\mathbf{1}^T\mathbf{x} - t) + \eta_2\left(\|\Delta\mathbf{x}\|_1 - \rho\right).$$

The Lagrange dual function is

$$\begin{aligned} Q(\eta_1, \eta_2) &= \min_{\mathbf{x}\in\{0,1\}^N} L(\eta_1, \eta_2, \mathbf{x}) \\ &= \min_{\mathbf{x}\in\{0,1\}^N} \left(-\mathbf{x}^T\mathbf{y} + \eta_1\mathbf{1}^T\mathbf{x} + \eta_2\|\Delta\mathbf{x}\|_1\right) - \eta_1 t - \eta_2\rho \\ &= q(\eta_1, \eta_2) - \eta_1 t - \eta_2\rho. \end{aligned}$$

For given η_1, η_2, the function $q(\eta_1, \eta_2)$ can be efficiently solved by s-t graph cuts [179, 185]. We then maximize $Q(\eta_1, \eta_2)$ by using the simulated annealing and obtain $\mathbf{x}_t^*$ as the optimum of (9.8). Finally, we optimize over t by evaluating each pair of t and $\mathbf{x}_t^*$ in the objective function (9.7). Since $\mathbf{x}$ takes only binary values, the optimization problem (9.8) is not convex. However, because optimizing $q(\eta_1, \eta_2)$ is a standard graph-cut problem, many previous works show that even the local minimum provides decent results [185] and the computation is remarkably efficient. We call the solution *local graph scan statistic* because it is a local optimum of the original optimization problem (9.7) by using the graph cuts.

[3] We implicitly assume that $\mathbf{1}_C^T\mathbf{y}/|C| > \epsilon$.

In the second method, we compute the graph scan statistic in a convex fashion by relaxing the original combinatorial optimization problem (9.7) as follows:

$$\begin{aligned} \widehat{r} &= \max_t \max_{\mathbf{x}} t \, \mathrm{KL}\left(\frac{\mathbf{x}^T\mathbf{y}}{t} || \epsilon\right) \\ &\quad \text{subject to } \mathbf{x} \in [0,1]^N, \|\Delta\mathbf{x}\|_1 \leq \rho, \mathbf{1}^T\mathbf{x} \leq t. \end{aligned} \tag{9.9}$$

The only difference between (9.7) and (9.9) is that we relax the the feasible set of $\mathbf{x}$ from $\{0,1\}^N$ to be a convex set $[0,1]^N$. We call $\widehat{r}$ *convex graph scan statistic*. When $\widehat{r}$ is larger than some threshold, we detect the activated node set and reject the null hypothesis. To analyze the property of the convex relaxation, we have the following theorem.

Theorem 23. Let the graph scan statistic be $\widehat{r}$ in (9.9) and let

$$\begin{aligned} \mu - \epsilon \;\geq\; & \frac{2\sqrt{2}}{\sqrt{|C|}}\Bigg(\frac{\log 2N + 1}{\sqrt{\left(\sqrt{\rho} + \sqrt{\frac{1}{2}\log N}\right)^2 \log N}} + \sqrt{2\log 2} \\ & + 2\sqrt{\left(\sqrt{\rho} + \sqrt{\frac{1}{2}\log N}\right)^2 \log N + \sqrt{2\log(1/\delta)}}\Bigg) \\ & + \sqrt{\frac{\log(2/\delta)}{2|C|}}. \end{aligned} \tag{9.10}$$

Under the statistical test (9.2) with $p = 1$, rejecting the null hypothesis for all $\widehat{r} > \tau$, with

$$\begin{aligned} \tau \;=\; & 8\Bigg(\left(\sqrt{\rho} + \sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)} \\ & + \sqrt{2\log 2} + \sqrt{2\log(1/\delta)}\Bigg)^2, \end{aligned}$$

implies that, under H_0, $\mathbb{P}\{\text{reject}\} \leq \delta$, and under H_1, $\mathbb{P}\{\text{reject}\} \geq 1 - \delta$.

See the proof of Theorem 23 in Appendix 14.7. The main idea is to show that under the null hypothesis, $\mathbf{x}^T(\mathbf{y} - \epsilon)/\sqrt{\mathbf{1}^T\mathbf{x}}$ is a sub-Gaussian random variable with mean zero; and under the alternative hypothesis, the maximum likelihood estimator $\mathbf{1}_C^T\mathbf{y}/|C|$ is a sub-Gaussian random variable with mean μ. Similarly to the graph wavelet statistic and the original graph scan statistic, the assumption (9.10) shows that the key to detect the activation is related to the properties of the ground-truth activated node set.

Corollary 11. H_0 and H_1 are asymptotically distinguishable by the convex graph scan statistic when

$$\sqrt{|C|}(\mu - \epsilon) \;=\; O\left(\max(\sqrt{\rho}, \sqrt{\log N})\sqrt{\log N}\right).$$

The sufficient condition is $|C| \geq O(\max(\rho, \log N)\log N)$ assuming μ, ϵ constant.

To compute the convex graph scan statistic, for a given t, we solve

$$\begin{aligned} \mathbf{x}_t^* &= \arg\min_{\mathbf{x}} -\mathbf{x}^T\mathbf{y}, \\ &\quad \text{subject to } \mathbf{x} \in [0,1]^N, \|\Delta\mathbf{x}\|_1 \leq \rho, \mathbf{1}^T\mathbf{x} \leq t. \end{aligned} \tag{9.11}$$

The objective function is linear and all the constraints are convex, so (9.11) can be easily solved by a convex optimization solver. In the earlier method, the search over η_1, η_2 is nonconvex and only guarantees a local minimum, but here the search is convex. Finally, we optimize over t by evaluating each pair of t and $\mathbf{x}_t^*$ in the objective function (9.9). Because of the convex relaxation, the final solution of $\mathbf{x}$ is not binary and a higher value of x_i indicates a higher confidence that the ith node is activated.

We summarize the above two methods to compute the graph scan statistic in Algorithm 8. In practice, the convex graph scan statistic outperforms the local graph scan statistic, but the local graph scan statistic is more appealing to deal with a large-scale graph.

Algorithm 8 Graph Scan Statistic

Input $\mathbf{y}$ input graph signal
ϵ noise level
Output $\mathbf{x}^*$ activated local set

Function
For a given t,
solve (9.7) by the graph cuts (local graph scan statistic)
or solve (9.9) by the convex optimization solver (convex graph scan statistic)
end
search over t, return the largest $t\,\mathrm{KL}\left(\frac{\mathbf{x}_t^{*T}\mathbf{y}}{t}||\epsilon\right)$ and $\mathbf{x}_t^*$ as $\mathbf{x}^*$

9.3.3 Discussion

Here we compare the proposed statistics.

- From an intuitive perspective, graph wavelet statistic selects a feature by projecting given attributes to a pre-constructed graph wavelet basis and is a data-independent and discriminative approach, which works only for detection.[4] Graph scan statistic searches over graphs and localizes the activated region and is a data-dependent and generative approach, which works for both detection and localization.
- From a statistical perspective, graph wavelet statistic and graph scan statistic require that the size of activated region be larger than $O(\rho \log^2 N)$, $O(\max(\rho, \log N) \log N)$, respectively, as shown in Corollaries 9, 10 and 11. Recall that we have two noise levels ϵ_0 and ϵ in (9.2). In general, we set $\epsilon_0 = \epsilon + |C|\,(\mu - \epsilon)\,/N$ to make sure that both hypotheses have the same mean. To derive the graph scan statistic, we need the equality assumption, that is, $\epsilon_0 = \epsilon$. However, the graph scan statistic empirically works well when ϵ_0 and ϵ are not equal, which will be shown in Section 9.4.

[4] It is also possible to use graph wavelet basis to do nonlinear approximation to localize the activated region, but this is beyond the scope of this paper.

- From a computational perspective, graph wavelet statistic is the cheapest to compute. Graph scan statistic can be implemented by two methods: the local graph scan statistic is also efficient by using efficient graph cuts and the convex graph scan statistic costs the most because it needs to solve a series of convex optimization problems.
- From the empirical performance, the convex graph scan statistic usually provides the best performance. Graph wavelet statistic provides slightly worse results. Because the graph cuts only provide a local solution, the computation of the local graph scan statistic is sensitive to the parameters and the initial conditions.

9.4 Experimental Results

We now evaluate our proposed methods on three datasets. We first study how detection performance changes according to parameters on a simulated dataset. Our observation is that the size of the ground-truth activated node set is crucial to the detection, which is consistent with Theorems 21, 22 and 23. We also show that our proposed methods can be applied to real-world problems of different domains. Here we consider two problems: air pollution detection and attribute ranking for community detection. Experimental results validate the efficiency and effectiveness of our proposed methods.

9.4.1 Simulations

We generate simulated data on the Minnesota road graph [97] and study how the parameters, including the signal strength μ, the noise level ϵ and the activation size $|C|$, influence the detection performance. The Minnesota road graph is a standard dataset including 2642 nodes and 3304 undirected edges [97]. We generate two binary graph signals as follows: we randomly choose one node as a node center and assign all other nodes that are within k steps to the community center to an activated node set, where k varies from 6 to 12. Figures 9.2 (a) and (b) show these two binary graph signals, where the nodes in yellow indicates the activated nodes and the nodes in blue indicates the nonactivated nodes. Using these two binary graph signals as templates, we then generate two classes of random graph signals: for attributes under H_1, each node inside the activated region is activated with probability μ and each node outside the activated region is activated with probability ϵ. Both μ and ϵ vary from 0.05 to 0.95 with interval of 0.1. For each combination of μ and ϵ, we generate corresponding attributes under H_0, the activation probability for each node is $\epsilon_0 = (\mu|C| + \epsilon(N - |C|))/N$. We run 100 random tests to compute the statistics and quantify the performance by the area under the receiver operating characteristic curve (AUC) [117]. To construct the graph wavelet basis, we compare three graph partition algorithms proposed in [93]. Since three algorithms provide similar performances, we only report the results given by the balanced partition in the maximum spanning tree [93].

Figures 9.2 (c), (e) and (g) show AUCs of the graph wavelet statistic, the local graph scan statistic (LGSS) and the convex graph scan statistic (CGSS) for the small activated region, where the step $k = 6$. For example, each block in Figure 9.2 (c) corresponds to the AUC of the graph wavelet statistic given a pair of μ and ϵ. A whiter block indicates a higher AUC and a better performance. Note that when μ is smaller than ϵ, we did not run the experiments and directly set

the corresponding AUC to zero. We see that the graph wavelet statistic has a similar performance with the convex graph scan statistic and both outperform the local graph scan statistic. Figures 9.2 (d), (f) and (h) show AUCs of the graph wavelet statistic, the local graph scan statistic and the convex graph scan statistic for the large activated region, where the step $k = 12$. We see that the convex graph scan statistic perform the best and the graph wavelet statistic has a slightly better performance than the local graph scan statistic. Comparing the results from two activated regions (left column versus right column in Figure 9.2), we see that all the methods perform better when the activated region is large. For example, both Figures 9.2 (c) and (d) use graph wavelet statistic. Given a fixed pair of μ and ϵ, a large activated region has a larger AUC, indicating higher probability to be detection.

To have a clearer understanding of how the proposed statistics work, we set the signal strength $\mu = 0.35$ and the noise level $\epsilon = 0.15$. Figures 9.3 (b) and (c) show the attribute under H_1 and H_0 given the ground-truth activated region in Figure 9.3 (a). When we compare the attributes under H_1 and H_0, it is clear that distinguishing H_1 from H_0 is not trivial.

Figures 9.3 (d), (f) and (h) compare the activated regions detected by graph wavelet basis, local graph scan statistic and convex graph scan statistic under H_1. The graph wavelet basis compares the average values between the nodes in yellow and the nodes in blue. When the difference is large, the activated region is detected. Ideally, we want all the nodes in blue are activated and all the nodes in yellow are nonactivated. Considering the graph wavelet basis is designed before obtaining any data, it captures the activated region fairly well. As expected, local graph scan statistic and convex graph scan statistic perform similarly and capture the activated region well. We also show the noisy attribute and the activated regions detected by graph wavelet basis, local graph scan statistic and convex graph scan statistic under H_0 in Figures 9.3 (c), (e), (g) and (i). The graph wavelet basis, local graph scan statistic and convex graph scan statistic cannot detect regions that are close to the true activated region from the pure noisy attribute.

	Attribute under H_0 Figure 9.3 (b)	Attribute under H_1 Figure 9.3 (c)
Modularity	1.5763	−3.5837
Cut cost	872	901
Wavelet	1.118	1.8688
LGSS	409.2793	553.5842
CGSS	416.4342	561.4883

Table 9.1: Facts about the data in Figures 9.3 (b) and (c).

Table 9.1 shows some facts about data in Figures 9.3 (b) and (c), including modularity, the number of cuts, graph wavelet statistic, graph scan statistic and convex graph scan statistic. Modularity is a popular metric to measure the strength of communities [26]. Networks with high modularity have dense connections within communities but sparse connections in different communities. Mathematically, the modularity of a binary attribute $\mathbf{1}_C \in \mathbb{R}^N$ is

$$\text{Modularity} = \sum_{(i,j)\in\mathcal{E}} \left(\mathrm{A}_{i,j} - \frac{d_i d_j}{M} \right) \frac{(\mathbf{1}_C)_i (\mathbf{1}_C)_j}{E},$$

where d_i is the degree of the ith node, $E = \sum_i d_i$ is the total number of edges, and $(\mathbf{1}_C)_i = 1$ when the ith node is activated; otherwise, $(\mathbf{1}_C)_i = 0$. A large modularity means the activated nodes are strongly connected.

Graph cuts measure the cost to separate a community from the other nodes. Mathematically, the number of cuts of the binary attribute $\mathbf{1}_C \in \mathbb{R}^N$ is $\text{Cut} = \|\Delta \mathbf{1}_C\|_0)$. A small cut cost means the activated nodes are easily separated from the nonactivated nodes.

We expect that under H_1, modularity is larger, indicating dense internal connections, and the number of cuts is smaller, indicating few external connections. From Table 9.1, however, we see that attributes under H_0 and H_1 contain similar number of activated nodes, the modularity under H_1 is smaller than the modularity under H_0, indicating the activated nodes under H_0 have even stronger internal connections. The number of cuts under H_0 is smaller than the number of cuts under H_1, indicating the activated nodes under H_0 are easier to be separated from the nonactivated nodes. It is clear that both modularity and number of cuts fail when the noise level is high. On the other hand, the graph wavelet statistic, local graph scan statistic and convex graph scan statistic under H_1 are much higher than those under H_0, indicating these three proposed statistics succeed even when the noise level is high. Graph wavelet statistic is robust because it selects a useful feature by using the graph wavelet basis. The graph scan statistic also is robust is because it localizes the true activated region first, which is equivalent to denoise the attribute based on the graph structure. Based on the denoised attribute, we compute the statistic values and the results are more robust. In other words, graph wavelet statistic extracts features from original attributes and is a discriminative approach to detect and graph scan statistic recovers a denoised attributes and is a generative approach.

In terms of the computational complexity, for each random test, it takes around 30 seconds to construct the graph wavelet basis, around 0.01 seconds to calculate the graph wavelet statistic, around 5 seconds to calculate the local graph scan statistic, around 10 seconds to calculate the convex graph scan statistic. Overall, the proposed statistics provide efficient and effective performances.

9.4.2 High Air Pollution Detection

Our first real-world example is on air pollution detection; we are particularity interested in the particle pollution as indicated by the fine particulate matter (PM 2.5), particles that are 2.5 micrometers in diameter or smaller, and can only be seen with an electron microscope. These tiny particles are produced from all types of combustion, including motor vehicles, power plants, residential wood burning, forest fires, agricultural burning, and some industrial processes. High PM 2.5 causes an increasing mortality rate for patients suffering from heart and lung diseases [186]. We aim to provide an efficient and effective approach of detecting high PM 2.5 regions, which can guide authorities in designing remedial measures.

The dataset comes from [187] and includes 756 operating sensors that record the daily average at various locations; Figure 9.4 (a) shows the daily distribution of PM 2.5 on July 1st, 2014, in the mainland U.S. We construct a ten-nearest neighbor graph where each sensor is a node and connects to ten neighboring sensors. In Figure 9.4 (b), the sensors whose measurements are above 15 are marked in yellow, which are relatively high-pollution cities.

When high-pollution cities are spread, the high pollution may cause by random events or

measurement failures, which makes the detection sensitive to noise. When high-pollution cities are clustered together, it indicates that an area consistently suffers from high pollution, which makes the detection robust. The meaning of using detection algorithms here is to answer whether the high-pollution cities are clustered and further provide a more robust high-pollution detector. In Figure 9.4 (b), the high-pollution cities seem to concentrate in the middle-east part of the U.S. Now we verify if these high-pollution cities are clustered together, by performing the graph wavelet statistic, graph scan statistic and convex graph scan statistic on this data. To make a comparison, we also simulate $1,000$ graph signals that have the same number of high-pollution cities, but are scattered across the U.S. Figures 9.4 (e), (f) and (g) show the statistical values. For each plot, the red dashed line shows the statistical value for the real pollution graph signal as shown in Figure 9.4 (b) and the black curves show the empirical histogramsof statistical values under $1,000$ random trials. We see that the statistical values of the real graph signals are always much larger than those of the scattered simulated graph signals for all three statistics, which means that it is easy to reject the null hypothesis and confirm that the high-pollution cities in Figure 9.4 (b) form a local cluster. Figures 9.4 (c) and (d) show the activated regions detected by the graph scan statistic and the convex graph scan statistic, respectively. We see that these two detected activated regions are similar and confirm that the mid-east region has a relatively high pollution.

We did the same experiments for the data collected on December 1st, 2014, as shown in Figure 9.5 (a). The dataset includes 837 operating sensors (the operating sensors are different every day) and we still construct a ten-nearest neighbor graph. We see that the high-pollution cities are more scattered, but some of them seems to cluster in the northwestern corner. Figures 9.5 (e), (f) and (g) show that the statistical values of the real data are still larger than those of the scattered simulated graph signals for all three statistics most of the time, which confirms that the high-pollution cities cluster together. Again, the two detected activated regions in Figures 9.5 (c) and (d) confirm that the northwestern corner has a relatively high pollution.

9.4.3 Ranking Attributes for Community Detection

Relevant node attributes improve the accuracy of community detection and interpreting the detected communities, but irrelevant attributes may harm the accuracy and cause computational inefficiency. By using proposed statistics, we can quantify the usefulness of each attribute. As a localized attribute tends to be related to a community structure, our methods can serve as a filter to select out those most useful attributes to benefit community detection.

We use the IEEE Xplore database to find working collaborations among scholars. The raw files were downloaded from [188]. We first construct three bipartite networks: a network of papers and journals, a network of papers and authors, and a network of papers and keywords (keywords are automatically assigned by IEEE). We focus on papers in ten journals: IEEE Transactions on Magnetics, Information Theory, Nuclear Science, Signal Processing, Electron Devices, Communications, Applied Superconductivity, Automatic Control, Microwave Theory and Techniques and Antennas and Propagation.

We project the bipartite network of papers and authors onto authors to create a co-authorship network where two authors are connected when they co-author at least four papers. We keep the largest connected component of the co-authorship network. As explained in more detail below,

we project the network of papers and journals and the network of papers and authors onto the authors in the the largest connected component to create the author-journal matrix, where rows denote authors and columns denote journals. We project the network of papers and keywords and a network of papers and authors onto the authors in the largest connected component to create the author-keyword matrix, where rows denote authors and columns denote keywords.

The entire dataset includes the co-authorship network with 7,330 authors and 108,719 co-authorships, the author-journal matrix with ten journals, and the author-keyword matrix with 3,596 keywords. In other words, we have a graph with 7,330 nodes, 108,719 edges and 3,596 corresponding attributes. We want to detect different academic communities based on graph structure and attributes. The academic communities can be defined based on the journals. Thus, ten journals correspond to ten ground-truth communities: when authors publish at least ten papers in a same journal, we assign them to a community. For example, authors who publish at least ten papers in the IEEE Transactions on Signal Processing form the signal processing community. In this community, some keywords are frequently used, such as 'filtering', 'Fourier transform' and 'wavelets'. By using those keywords, the quality of community detection may be improved.

The goal is to rank all keywords based on their contribution to community detection, where the magnitude or value (if negative is meaningful) of the corresponding statistic is used to determine the rank. We consider four ranking methods: graph wavelet statistic based ranking, local graph scan statistic based ranking, modularity-based ranking and cut-based ranking. We did not use the convex graph scan statistic due to computational complexity. For the first two ranking methods, we compute the statistical value and rank the keywords according to the statistical value in a descending order. This is because a larger statistic means a larger probability that this keyword forms a community. For the modularity-based ranking, we compute the modularity of each keyword and rank the keywords according to modularity in a descending order. For the cut-based ranking, we compute the cut cost of each keyword and rank the keywords according to the number of cuts in a ascending order.

To quantify the real community detection power of keywords, we compare each keyword to the ground-truth community and compute the correspondence by using the average F1 score, suggested in [180, 181]. Mathematically, let C^* be a set of the ground-truth communities and $\widehat{C}$ be a set of the activated node sets provided by the node attributes. Each node set $\widehat{C}_i \in \widehat{C}$ collects the nodes that have the same attribute. The average F1 score is

$$\frac{1}{2|C^*|}\sum_{C_i \in C^*} F1(C_i, \widehat{C}_{g(i)}) + \frac{1}{2|\widehat{C}|}\sum_{\widehat{C}_i \in \widehat{C}} F1(\widehat{C}_{g'(i)}, \widehat{C}_i),$$

where the best matching g and g' is defined as follows:

$$g(i) \;=\; \arg\max_j F1(C_i, \widehat{C}_j) \text{ and } g'(i) = \arg\max_j F1(C_j, \widehat{C}_i),$$

where $F1(C_i, \widehat{C}_j)$ is the harmonic mean of precision and recall. A large average F1 score means that the community induced by a keyword agrees with the community induced by journal papers. We also compute the average F1 score of each keyword and rank the keywords according to the average F1 scores in a descending order, which is the ground-truth ranking. We compare the four estimated rankings with the ground-truth ranking by using the Spearman's rank correlation

coefficient [189]. The Spearman correlation coefficient is defined as the Pearson correlation coefficient between the ranked variables,

$$\text{correlation} \ = \ 1 - \frac{6 \sum |p_i - q_i|^2}{N(N^2 - 1)},$$

where $p_i - q_i$ is the difference between two rankings. The Spearman correlation coefficients of modularity-based ranking, cut-based ranking, graph wavelet statistic based ranking and local graph scan statistic based ranking are shown in Figure 9.6. We see that the graph wavelet statistic and local graph scan statistic outperform other methods. Cut-based ranking performs poorly because it may rank infrequent keywords higher. To account for this effect, we also consider the average modularity, which is the modularity divided by the number of activated authors, and the average cuts, which are the number of cuts divided by the number of activated authors. We see that the average cuts perform much better than the total cuts.

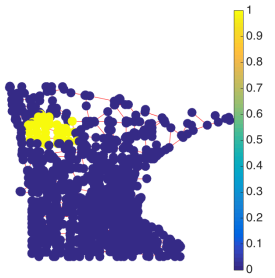
Figure 9.7 compares the average F1 scores as a function of individual keywords ordered by the six ranking methods. The x-axis is the ranking provided by the proposed ranking methods and the y-axis is the average F1 score of the corresponding keyword. For example, since the local graph scan statistic ranks *Maximum likelihood detection* in the first place, we put the corresponding average F1 score as the first element on the red curve (leftmost). We expect that the curve goes down as the rank grows because a good ranking method ranks the important keywords higher. We also use cluster affiliation model for big networks (BIGCLAM, shown in black), a large-scale overlapping community detection algorithm to provide a baseline [180]. We see that the local graph scan statistic is slightly better than the graph wavelet statistic and both of them outperform the other methods, which is consistent with the results given by the Spearman correlation coefficients in Figure 9.6. Average cuts rank important keywords lower, causing the F1 score to increase as the rank decreases. The average cuts fail because small average cuts may come from just a few activations. For example, a keyword activating only one author has a small cut number, although this keyword is actually trivial. Surprisingly, using high ranking keywords selected by LGSS and wavelets works even better than BIGCLAM on the task of community detection. The reason may be that in this co-authorship network, only some keywords are informative and strongly related to the journals, which are the ground-truth communities, while most keywords may provide trivial or misleading information.

9.5 Conclusions

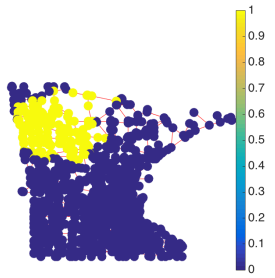
In this chapter, we formulated hypothesis tests to decide whether the observations activate a community in a graph that is corrupted by Bernoulli noise. We modeled the observations as a binary graph signal: when the signal coefficient is positive, we say that the corresponding node is activated; when the activated nodes form a cluster, we say that the graph signal contains a structure-correlated activation. We proposed two statistics for testing: graph wavelet statistic and graph scan statistic. Both are shown to be efficient and statistically effective to detect activations. The intuition behind these statistics is to localize the underlying localized pattern first, which is equivalent to denoising a given attribute based on graph structure. Theorems 21, 22 and 23 show that the key to distinguishing the activation is the difference between the signal strength and the

noise level, and structural properties of the signal. Specifically, when the size of the ground-truth activated node set is larger and the ground-truth activated node set is easier to be separated from the other nodes, it is less difficult to detect the activation.

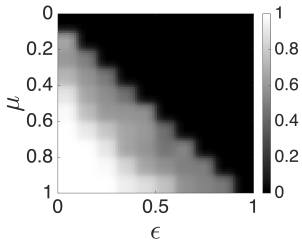
We validated the effectiveness and robustness of the proposed methods on simulated data first; experimental results match the theorems well. We further validated the proposed methods on two real-world applications: high air pollution detection and attribute ranking; experimental results show the proposed statistics are effective and efficient.



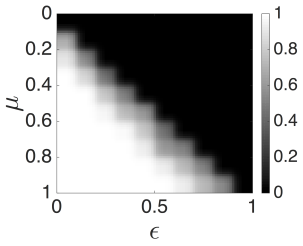
(a) Activation (small).



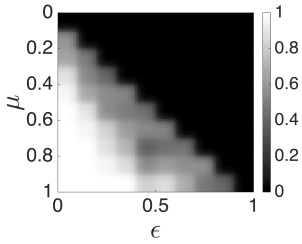
(b) Activation (large).



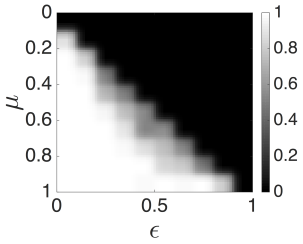
(c) Graph wavelet statistic (small).



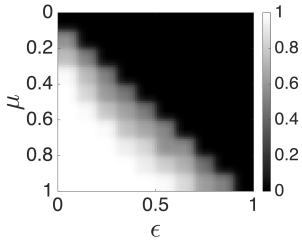
(d) Graph wavelet statistic (large).



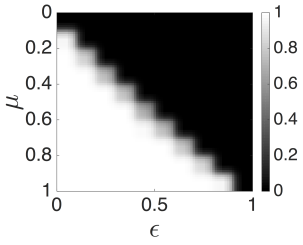
(e) LGSS (small).



(f) LGSS (large).

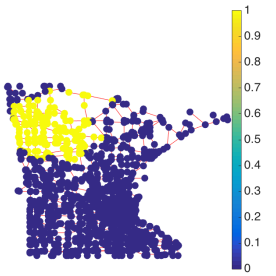


(g) CGSS (small).

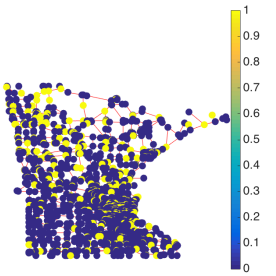


(h) CGSS (large).

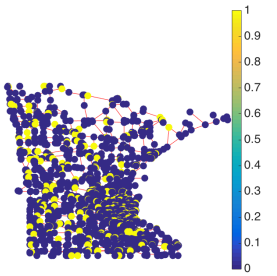
Figure 9.2: Comparison of graph wavelet statistic and graph scan statistic on the simulated dataset. The left column shows the results for a small activated region and the right column shows the results for a large activated region. All methods perform better when an activated region is larger. For a same activated region, all methods perform better when $\mu - \epsilon$ is larger.



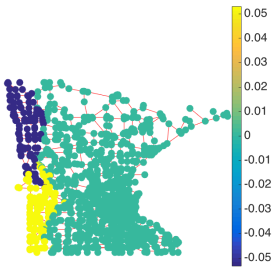
(a) Original attribute.



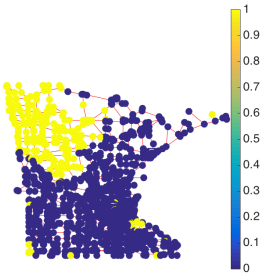
(b) Attribute under H_1.



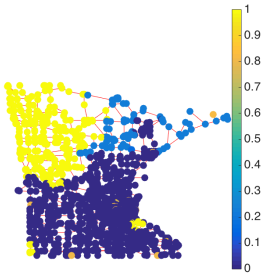
(c) Attribute under H_0.



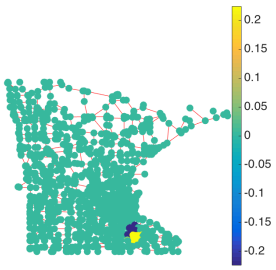
(d) Activated wavelet basis vector under H_1



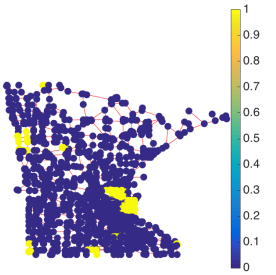
(e) Activation (LGSS) under H_1



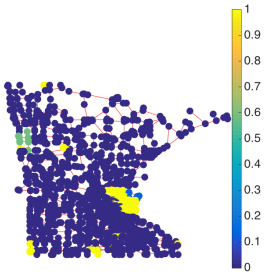
(f) Activation (CGSS) under H_1



(g) Activated wavelet basis vector under H_0

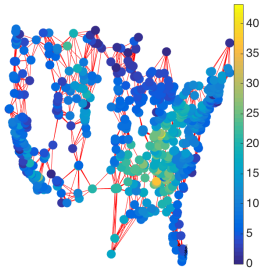


(h) Activation (LGSS) under H_0

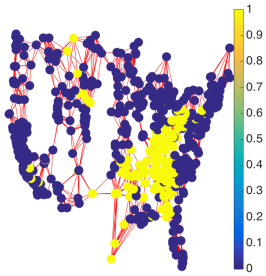


(i) Activattion (CGSS) under H_0

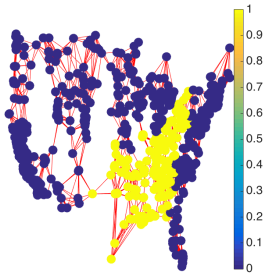
Figure 9.3: Illustration of how the proposed statistics work. Under H_1, the graph wavelet statistic, local graph scan statistic and convex graph scan statistic denoise the given attribute and localize the true community. Under H_0, the graph wavelet basis, local graph scan statistic and convex graph scan statistic cannot localize the true community. The denoising procedure is the key to robustness. Graph wavelet statistic extracts features from original attributes and is a discriminative approach. (d) shows a graph wavelet basis vector corresponding to the maximum absolute value of the graph wavelet coefficients. Graph scan statistic recovers denoised attributes and is a generative approach. (e) and (f) show the activation recovered by graph scan statistics. For CGSS, due to the convex relaxation, the recovered activated region is not binary. A higher value of x_i indicates a higher confidence that the ith node is activated.



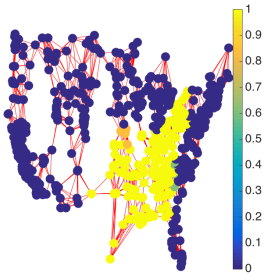
(a) Original data.



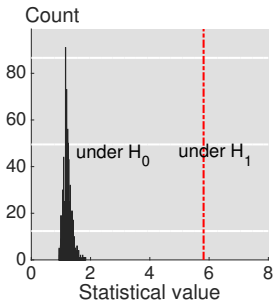
(b) Thresholding (> 15).



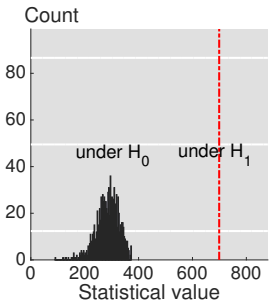
(c) Activation detected by LGSS.



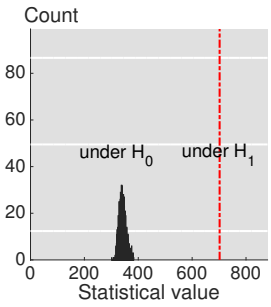
(d) Activation detected by CGSS.



(e) Graph wavelet statistic.

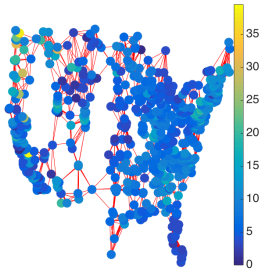


(f) LGSS.

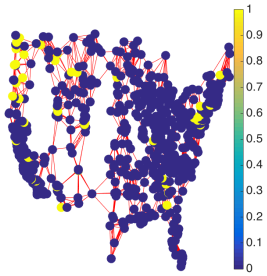


(g) CGSS.

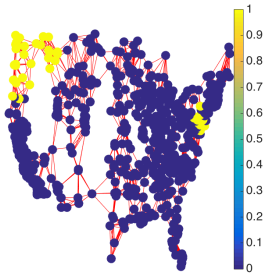
Figure 9.4: Detecting the high-pollution region on July 1st, 2014. The proposed statistics aim to answer whether the high-pollution cities are clustered and further provide high-pollution regions. Plot (b) shows high-pollution cities. Plots (c) and (d) show the high-pollution regions recovered by the graph scan statistics. Plots (e), (f) and (g) show that graph wavelet statistic and graph scan statistic successfully detect the high-pollution regions from random attributes. For each plot, the red dashed line shows the statistical value for the real pollution graph signal as shown in Plot (b) and the black curves show the empirical histograms of statistical values under $1,000$ random trials. Note that graph wavelet statistic only answers whether the high-pollution regions exist, but cannot localize where these regions are.



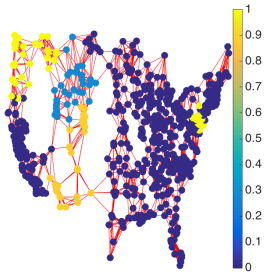
(a) Original data.



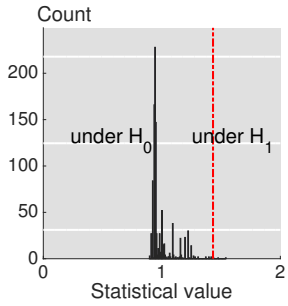
(b) Thresholding (> 15).



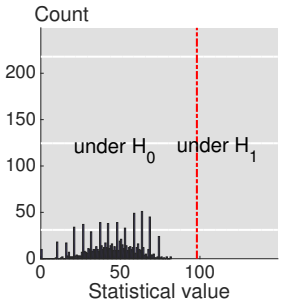
(c) Activation detected by LGSS.



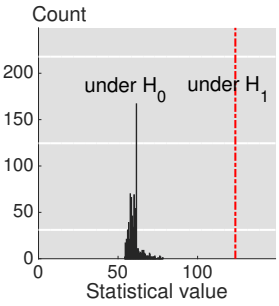
(d) Activation detected by CGSS.



(e) Graph wavelet statistic.



(f) LGSS.



(g) CGSS.

Figure 9.5: Detecting the high-pollution region on December 1st, 2014. Plot (b) shows high-pollution cities. Plots (c) and (d) show the high-pollution regions recovered by the graph scan statistics. Plots (e), (f) and (g) show that graph wavelet statistic and graph scan statistic detect the high-pollution regions from random attributes with high probabilities. For each plot, the red dashed line shows the statistical value for the real pollution graph signal as shown in Plot (b) and the black curves show the empirical histograms of statistical values under $1,000$ random trials. From the gaps between the statistical values under two hypotheses, we see that CGSS provides the most decisive conclusion.

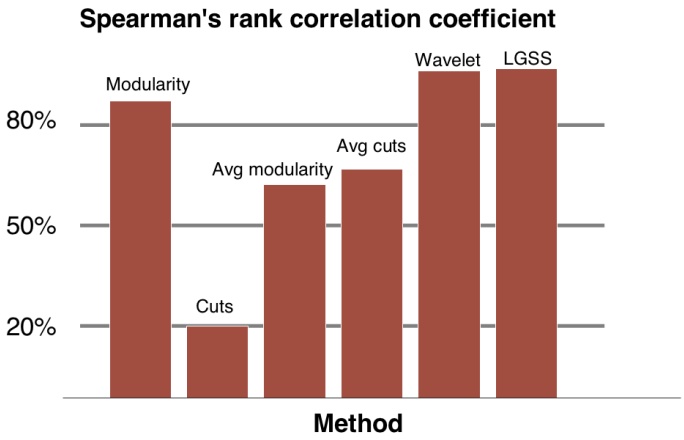


Figure 9.6: Comparison of Spearman's rank correlation coefficients. A larger Spearman's rank correlation coefficient means higher correlation to the ground-truth ranking.

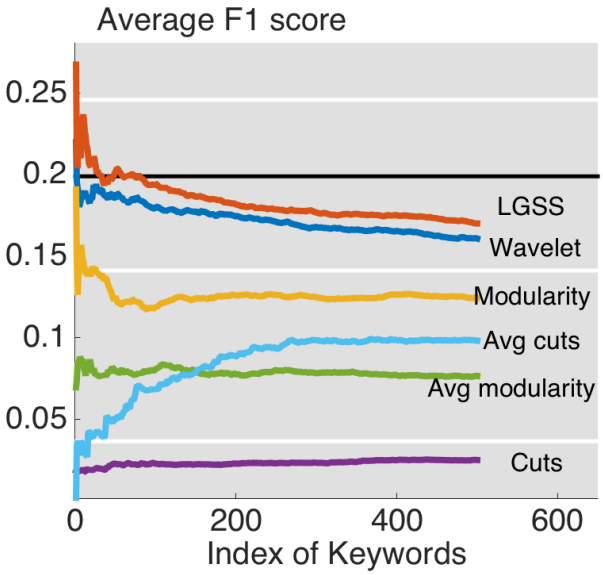


Figure 9.7: Comparison of the average F1 score as a function of the top k ranked keywords. A higher average F1 score means better detection performance by using each individual keyword. The black horizontal line shows the performance of BIGCLAM, a large-scale overlapping community detection algorithm.

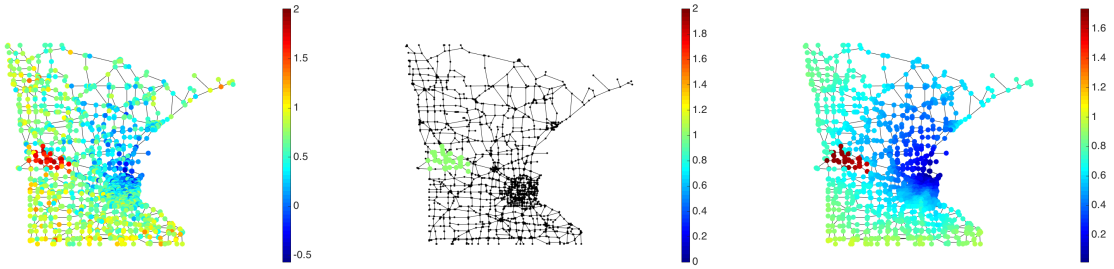
Chapter 10

Localization of Localized Graph Signals

10.1 Introduction

The task of finding activated supports of signals/images has been intensely studied in classical signal/image processing from various aspects over the past few decades. For example, impulse detection considers localizing impulses in a noisy signal [3]; support recovery of sparse signals considers localizing sparse activations with a limited number of samples [5, 6]; foreground detection considers localizing foreground in a video sequence [7]; cell detection and segmentation considers localizing cells in microscopy images [8] and matched filtering consider localizing radar signals in the presence of additive stochastic noise [9, 10].

In this chapter, we consider a counterpart problem on graphs, localizing activated supports of signals in a large-scale graph. The aim of graph signal localization is to identify one set of connected nodes where the graph signal switches values—we call an *activated piece*. For example, given a graph signal in Figure 10.1(a), we are looking for an underlying activated piece as in Figure 10.1(b). As the original signal is noisy, this task is related but not equivalent to denoising. Denoising aims to obtain a noiseless graph signal (see Figure 10.1(c)), which is not necessarily localized. In this paper, we focus on localizing activated pieces in noisy piecewise-constant signals in which case localization is in fact equivalent to denoising.



(a) Signal. (b) Activated piece. (c) Noiseless signal.

Figure 10.1: Signal localization on graphs. Given a signal (a), the aim is to identify an activated piece (b) while denoising aims to obtain a noiseless signal (c). When smooth background is ignored, localization is equivalent to denoising.

Note that most of the work in the graph signal processing area happens in the graph frequency domain; we here choose to work in the graph vertex domain because it provides better localization in the graph vertex domain and is easier for visualization and explanation than the graph frequency domain. For example, in classical image processing, most methods for edge detection and image segmentation work in the space domain, instead of the frequency domain.

This task is relevant to many real-world applications from localizing virus attacks in cyber-physical systems to activity in brain connectivity networks, to traffic events in city road networks. Similarly to classical localization problems, the key issue is to separate localized activated patterns from background noise on graphs; in other words, we aim to find the supports of localized activated patterns on graphs.

In the previous literature, signal localization has been considered from many aspects. From a signal processing perspective, people studied signal/noise discrimination [168], matched filtering [9, 10] and support recovery of sparse signals [5, 6]; from a image processing perspective, people studied object detection in natural and biomedical images [169], and foreground detection in a video sequence [7]; and from a data mining perspective, people studied anomaly detection [190, 191]. Within the context of graphs, much of the literature considers detecting smooth or piecewise-constant graph signals from signals corrupted by Gaussian noise [78, 173, 175, 177, 178]. A recent work [192] considers detecting localized graph signals corrupted by Bernoulli noise.

10.2 Problem Formulation

Consider localizing an activated piece $C \in \mathcal{C}$ (where $\mathcal{C}$ is the set containing all the pieces) in a noisy, piecewise-constant graph signal

$$\mathbf{x} \;=\; \mu \mathbf{1}_C + \epsilon, \tag{10.1}$$

where $\mathbf{1}_C$ is the indicator function, μ is the signal strength and $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$ is Gaussian noise. We consider two cases: $\mu = 1$ and μ arbitrary.

Previous works formulate this as a detection problem via a scan statistic searching for a most probable anomaly set, which is similar to localizing an activated piece. However, such a method is either computationally inefficient or hindered by strong assumptions. For example, in [173], the authors analyze the theoretical performance of detecting paths, blobs and spatial temporal sets by exhaustive search, which is clearly inefficient. In [177, 178, 192], the authors aim to detect a node set with a specific cut number, resulting in a computationally efficient algorithm, but limited by strong assumptions.

Our goal is to develop an algorithm to efficiently localize an activated piece with a general shape. The maximum likelihood estimator of the one-piece-localization problem under Gaussian noise is

$$\min_{\mu, C} \|\mathbf{x} - \mu \mathbf{1}_C\|_2^2, \quad \text{subject to } C \in \mathcal{C}. \tag{10.2}$$

10.3 Methodology

10.3.1 Localization with unit magnitude

Consider first (10.1) with a unit-magnitude signal, that is, $\mu = 1$. Then (10.2) reduces to

$$\min_{C} \|\mathbf{x} - \mathbf{1}_C\|_2^2, \quad \text{subject to } C \in \mathcal{C}. \tag{10.3}$$

This optimization problem is at the core of this paper. In the following context, we first consider a hard thresholding algorithm as a baseline and then we study two typical and complementary classes of a connected component, including ball-shaped class and elongated-shaped class, where we have fast algorithms to solve (10.3). In the end, we will combine both solvers to obtain a general solver.

Hard thresholding based localization. The difficulty in solving (10.3) comes from the constraint, which forces the activated nodes to form a connected subgraph (piece). Since we do not restrict the size and shape of a piece, searching over all the pieces to find the global optimum is an NP-complete problem. We consider a simple algorithm that solves (10.3) in two steps: we first optimize the objective function and then project the solution onto the feasible set in (10.3). This simple algorithm guarantees to satisfy the constraints, but is not necessarily the optimal solution. The objective function can be formulated as

$$\min_{C} \|\mathbf{x} - \mathbf{1}_C\|_2^2 = \min_{\mathbf{t}\in\{0,1\}^N} \|\mathbf{x} - \mathbf{t}\|_2^2 = \min_{t_i\in\{0,1\}} \sum_{i=1}^{N} (x_i - t_i)^2,$$

where $\mathbf{t} \in R^N$ is a vector representation of a node set C. Since each t_i is independent, we can optimize each individual element to obtain $t_i^* = 1$, when $\mathbf{x}_i > 1/2$; and 0, otherwise. In other words, the optimum of the objective function is

$$\{v_i \in \mathcal{V} \mid x_i > \frac{1}{2}\} \;=\; \min_{C} \|\mathbf{x} - \mathbf{1}_C\|_2^2 .$$

This step is nothing but global hard thresholding. We then project this solution onto the feasible set; that is, we solve

$$\begin{aligned} C_{\text{thr}}^* &= \mathrm{P}_{\mathcal{C}}\left(\arg\min_{C} \|\mathbf{x} - \mathbf{1}_C\|_2^2\right) \\ &= \mathrm{P}_{\mathcal{C}}\left(\{v_i \in \mathcal{V} \mid x_i > \frac{1}{2}\}\right), \end{aligned} \tag{10.4}$$

where the projection operator $\mathrm{P}_{\mathcal{C}}(C)$ extracts the largest connected component (piece) in a node set C. Thus, this solver simply performs hard thresholding and then finds a connected component among the nodes with nonzero elements on it.

Cut-based localization for ball-shaped class. The key idea in (10.4) is to partition all the nodes into two categories according to a given graph signal: activated nodes and inactivated nodes. This is similar to graph cuts: cutting a series of edges with minimum cost and obtaining two isolated components. The difference is that one optimizes edge weights in graph cuts,

while (10.3) optimizes signal coefficients. Inspired by [177, 192], we add the number of edges connecting activated nodes and inactivated nodes to the objective function to induce a connected component.

A signal $\Delta \mathbf{1}_C \in \mathbb{R}^M$ records the first-order differences within $\mathbf{1}_C$. The number of edges connecting C and $\overline{C}$ is

$$\|\Delta \mathbf{1}_C\|_0 = \|\mathrm{A}\,\mathbf{1}_C\|_1 - \mathbf{1}_C^T\,\mathrm{A}\,\mathbf{1}_C,$$

where A is the adjacency matrix, $\|\mathrm{A}\,\mathbf{1}_C\|_1$ is the sum of the degrees of the nodes in C and $\mathbf{1}_C^T\,\mathrm{A}\,\mathbf{1}_C$ is the inner connection of nodes within C. When there are no edges connecting the activated nodes, $\|\Delta \mathbf{1}_C\|_0 = \|\mathrm{A}\,\mathbf{1}_C\|_1$. When all the activated nodes are well connected and form a piece, $\mathbf{1}_C^T\,\mathrm{A}\,\mathbf{1}_C$ is often large and $\|\Delta \mathbf{1}_C\|_0 \ll \sum_{i\in C} d_i$ with d_i the degree of the ith node. Thus, minimizing $\|\Delta \mathbf{1}_C\|_0$ induces C to be a piece. We thus solve

$$C_\lambda = \mathrm{P}_{\mathcal{C}}\left(\arg\min_C \|\mathbf{x} - \mathbf{1}_C\|_2^2 + \lambda\,\|\Delta \mathbf{1}_C\|_0\right), \tag{10.5a}$$

$$C^*_{\text{cut}} = \arg\min_{C_\lambda} \|\mathbf{x} - \mathbf{1}_{C_\lambda}\|_2^2. \tag{10.5b}$$

Given a λ, (10.5a) solves the regularized optimization problem and extracts the largest connected component. In (10.5b), we optimize over λ and obtain a solution. The previous hard thresholding solution (10.4) is a subcase of (10.5b) when we force $\lambda = 0$. Computationally, we can solve the regularized optimization problem in (10.5a) by using the Boykov-Kolmogorov graph cuts algorithm [179, 185]. We call this *cut-based localization*.

To understand the regularized problem (10.5a) better, we have

$$\begin{aligned}
&\min_C \|\mathbf{x} - \mathbf{1}_C\|_2^2 + \lambda\,\|\Delta \mathbf{1}_C\|_0 \\
= &\min_{\mathbf{t}\in\{0,1\}^N} \|\mathbf{x} - \mathbf{t}\|_2^2 + \lambda\,\|\Delta \mathbf{t}\|_0 \\
= &\min_{t_i\in\{0,1\}} \sum_i \left((x_i - t_i)^2 + \lambda \sum_{j\in \text{Nei(i)}} |t_i - t_j|\right),
\end{aligned}$$

where $\mathrm{Nei}(i)$ is the neighborhood of the ith node. That is, we have $t_i^* = 1$ when

$$(x_i - 1)^2 + \lambda \sum_{j\in\mathrm{Nei}(i)} |x_j - 1| < x_i^2 + \lambda \sum_{j\in\mathrm{Nei}(i)} |x_j|.$$

The solution is

$$(\mathbf{t}^*_{\text{cut}})_i = \begin{cases} 1, & x_i > \frac{1}{2} + \frac{\lambda}{2}\,(\alpha_0 - \alpha_1)\,; \\ 0, & \text{otherwise}, \end{cases} \tag{10.6}$$

where α_1 counts the neighbors of x_i equaling to 1 and α_0 counts the neighbors of x_i equal to 0. From the above derivation, we see that (10.6) is nothing but an adaptive local thresholding. In other words, the value of each element depends on the values of its neighbors. This explains why cut-based localization outperforms hard thresholding.

Path-based localization for enlongated-shaped class. Cut-based localization is an efficient solver to localize activated nodes that can be easily separated from inactivated ones. The

difficulty of separating these two classes is described by the cut number, which is good at promoting ball-shaped pieces, but not good at promoting elongated-shaped pieces. For example, to solve (10.3), we introduced $\|\Delta \mathbf{1}_C\|_0$ in (10.5a) to promote fewer connections between activated nodes and inactivated ones. This is because when C is a piece, $\mathbf{1}_C^T \mathrm{A}\, \mathbf{1}_C$ is large. When nodes in C are fully connected, C forms a ball-shaped piece (see Figures 10.2 (a)–(b)) and $\mathbf{1}_C^T \mathrm{A}\, \mathbf{1}_C = |C|(|C|-1)/2$; however, when nodes in C are weakly connected, for example, when C forms an elongated-shaped piece, $\mathbf{1}_C^T \mathrm{A}\, \mathbf{1}_C = |C| - 1$. In other words, (10.5a) is better suited to capturing ball-shaped pieces than elongated-shaped pieces.

To address elongated-shaped pieces, we consider two methods. The first method promotes an elongated-shaped piece based on convex optimization. We first restrict $\|\mathrm{A}\, \mathbf{1}_C\|_\infty$, the maximum degree of the nodes in the activated piece to be at most 2, pushing a piece has an elongated shape, and then solve the following optimization problem,

$$\min_C \|\mathbf{x} - \mathbf{1}_C\|_2^2, \quad \text{subject to} \quad \|\mathrm{A}\, \mathbf{1}_C\|_\infty \leq 2.$$

In other words, the constraint requires that each activated node connect to at most two activated nodes, which promotes elongated-shaped pieces. To solve this combinatorial problem efficiently, we relax it to a convex problem,

$$\begin{aligned} \mathbf{t}_{p_1}^* &= \arg\min_{\mathbf{t}\in\mathbb{R}^N} \|\mathbf{x} - \mathbf{t}\|_2^2, \\ &\text{subject to } \|\mathrm{A}\, \mathbf{t}\|_\infty \leq 2 \quad \text{and} \quad 0 \leq \mathbf{t} \leq 1. \end{aligned}$$

We then set various thresholds for $\mathbf{t}_{p_1}^*$, extract the largest connected component, optimize over the threshold to obtain a solution,

$$\begin{aligned} C_\lambda &= \mathrm{P}_{\mathcal{C}}\left(\mathbf{1}_{\mathbf{t}_{p_1}^* \geq \lambda}\right), \\ C_{p_1}^* &= \arg\min_{C_\lambda} \|\mathbf{x} - \mathbf{1}_{C_\lambda}\|_2^2. \end{aligned} \tag{10.7}$$

The solution promotes an elongated shape, but does not restrict to an exact path.

The second method searches for an exact path based on the shortest path algorithm. We consider the following optimization problem.

$$C_{p_2}^* = \arg\min_{C \text{ is a path}} \|\mathbf{x} - \mathbf{1}_C\|_2^2 + \lambda|C|, \tag{10.8}$$

where the regularization parameter $\lambda > 2x_{\max} - 1$ with $x_{\max}$ the maximum element in $\mathbf{x}$, guaranteeing that shortest path algorithm will work later. The optimization problem (10.8) can be implemented in two steps as follows

$$\min_{\text{all pairs of } s,t} \left(\min_{C \text{ connects } s,t} \|\mathbf{x} - \mathbf{1}_C\|_2^2 + \lambda|C| \right).$$

In the first step, we fix two end points s and t and find the optimal path C connecting s and t; in the second step, we enumerate all pairs of end points and find the globally optimal path. We will show that the first step can be solved exactly by the shortest path algorithm.

The objective function of (10.8) can be rewritten as

$$\begin{aligned} & \|\mathbf{x}-\mathbf{1}_C\|_2^2+\lambda|C| \\ = \; & \mathbf{x}^T\mathbf{x}-2\mathbf{1}_C^T\mathbf{x}+(1+\lambda)\mathbf{1}_C^T\mathbf{1} \\ = \; & \mathbf{x}^T\mathbf{x}+2\mathbf{1}_C^T\mathbf{y}, \end{aligned}$$

where $\mathbf{y}=\frac{1+\lambda}{2}\mathbf{1}-\mathbf{x}$. Since C is the only variable, minimizing (10.8) is equivalent to minimizing $\mathbf{1}_C^T\mathbf{y}$.

We consider a graph with edge weight

$$\mathrm{W}_{i,j} \; = \; \begin{cases} \frac{y_i+y_j}{2}, & (i,j)\in\mathcal{E}; \\ \infty, & \text{otherwise.} \end{cases}$$

Let C be a path connecting two end points s and t. The path weight of C is

$$\begin{aligned} \sum_{i,j\in C}\mathrm{W}_{i,j} \; &= \; \sum_{k\in C}y_k-\frac{y_s+y_t}{2} \\ &= \; \mathbf{1}_C^T\mathbf{y}-\frac{y_s+y_t}{2}. \end{aligned}$$

This indicates that given two end points s and t, minimizing $\mathbf{1}_C^T\mathbf{y}$ is equivalent to minimizing the path weight. Since $\lambda>2x_{\max}-1$, all the edge weights are non-negative. We can then use the shortest path algorithm to efficiently compute the short paths between all pairs of nodes [16]. Graph coarsening techniques can be used to reduce the number of nodes and boost the speed [193, 194].

Finally, the elongated-shaped piece is obtained by choosing the one with a smaller objective value in (10.3).

$$C^*_{\mathrm{path}} \; = \; \arg\min_{C\in\{C^*_{p_1},C^*_{p_1}\}}\|\mathbf{x}-\mathbf{1}_C\|_2^2\,. \tag{10.9}$$

We call this *path-based localization*.

Combined solvers. We combine the cut-based localization C^*_{cut} and the path-based localization C^*_{path} by choosing the one with a smaller objective value. The final solution to localize a general-shape piece is

$$C^* \; = \; \mathrm{Loc}_1(\mathbf{x}) \; = \; \arg\min_{C\in\{C^*_{\mathrm{cut}},C^*_{\mathrm{path}}\}}\|\mathbf{x}-\mathbf{1}_C\|_2^2\,, \tag{10.10}$$

where $\mathrm{Loc}_1(\cdot)$ is the operator that finds an activated piece with unit magnitude. This solution is not the global optimum of (10.3); it considers two typical and complementary subsets in the feasible set, and combines a graph-cut solver, a convex programming solver and a shortest-path solver, all of which are efficient. Note that detection or localization techniques usually need to set thresholds; however, this localization solver (10.10) is parameter-free and directly output the supports of a localized pattern. In Section 10.4, we validate it empirically.

10.3.2 Localization with unknown magnitude

We next consider a more general case where the magnitude of the activated piece is unknown, (10.1), with the associated optimization problem (10.2).

We iteratively solve C and μ until convergence; that is, given C, we optimize over μ and then given μ, we optimize over C. In the kth iteration,

$$\begin{aligned} \mu^{(k)} &= \min_{\mu} \|\mathbf{x} - \mu \mathbf{1}_{C^{(k)}}\|_2^2 = \frac{\mathbf{x}^T \mathbf{1}_{C^{(k)}}}{\mathbf{1}_{C^{(k)}}^T \mathbf{1}_{C^{(k)}}}, \\ C^{(k)} &= \mathrm{Loc}_1(\frac{1}{\mu}\mathbf{x}). \end{aligned}$$

We obtain a pair of local optima by alternatively minimizing these two variables. We denote this solver by

$$\mu^*, C^* = \mathrm{Loc}_{\mathrm{unknown}}(\mathbf{x}). \tag{10.11}$$

10.4 Experimental Results

We test our localization solver on two graphs: the Minnesota road network and the Manhattan street network. On each graph, we consider localizing two classes of simulated graph signals with different activated sizes under various noise levels.

10.4.1 Minnesota road network

We model the Minnesota road network as a graph with intersections as nodes and road segments between intersections as edges. The graph includes $2,642$ nodes and $3,342$ undirected edges. We simulate two classes of one-piece graph signals: a ball-shaped class and an elongated-shaped class.

Ball-shaped class. To generate a ball-shaped piece, we randomly choose one node as a node center and assign all the other nodes that are within k steps of the node center to an activated node set, where the radius k is either 5 or 10. Figures 10.2(a)–(b) show examples of ball-shaped pieces with radii $k = 5$ and 10 (left and right columns, respectively). The nodes in green (lighter) indicate the activated nodes and the nodes in black (darker) indicate the inactivated nodes. We aim to localize the activated pieces from noisy one-piece graph signals. We vary the noise variance σ^2 from 0.1 to 1 with interval 0.1 and at each randomly generate $1,000$ noisy one-piece graph signals.

We measure the quality of the localization with two measures: the Hamming distance and the F_1 score. The *Hamming distance* is equivalent to the Manhattan distance between two binary graph signals, that is, it counts the total number of mismatches between the two,

$$d_{\mathrm{H}}(\widehat{C}, C) = \left\|\mathbf{1}_{\widehat{C}} - \mathbf{1}_C\right\|_1 = |C \cup \widehat{C}| - |C \cap \widehat{C}|,$$

where C is the ground truth for the activated piece and $\widehat{C}$ is the localized piece. The lower the Hamming distance the better the quality: the Hamming distance reaches its minimum value at $d_{\mathrm{H}}(\widehat{C}, C) = 0$.

The F_1 *score* is the harmonic mean of the precision and recall, and measures the matching accuracy,

$$F_1(\widehat{C}, C) \;=\; 2\frac{\text{precision}(\widehat{C}, C)\;\text{recall}(\widehat{C}, C)}{\text{precision}(\widehat{C}, C) + \text{recall}(\widehat{C}, C)},$$

where precision is the fraction of retrieved instances that are relevant (true positives over the sum of true positives and false positives), while recall is the fraction of relevant instances that are retrieved (true positive over the sum of true positives and false negatives) [195]. Thus, with true positives $= |C \cap \widehat{C}|$, sum of true positives and false positives $= |\widehat{C}|$ and the sum of the true positives and false negatives $= |C|$, we have

$$F_1(\widehat{C}, C) \;=\; 2\frac{|C \cap \widehat{C}|}{|C| + |\widehat{C}|} \;=\; \frac{2|C \cap \widehat{C}|}{d_{\text{H}}(\widehat{C}, C) + |C \cap \widehat{C}|}.$$

The higher the F_1 score the better the quality: the F_1 score reaches its maximum value at $F_1 = 1$ and minimum value at $F_1 = 0$.

The Hamming distance emphasizes the difference between two sets while the F_1 score emphasizes their agreement; the higher the F_1 score, or the lower the Hamming distance, the better the localization quality.

Figures 10.2(c)–(d) show the F_1 scores and Figures 10.2(e)–(f) the Hamming distances of localizing ball-shaped pieces with radii $k = 5$ and 10 (left and right columns, respectively) as functions of the noise variance. Each figure compares four methods: hard thresholding (10.4) (hard, grey-solid line), path-based localization (10.9) (path, yellow-circle line), cut-based localization (10.5b) (cut number, purple-square line), combined localization (10.10) (combine, red-solid line), local-set-based piecewise-constant dictionary (LSPC, green-square line) [93], trend filtering on graphs (TF) [196] and graph Laplacian denoising (gLap) [1]. LSPC is a predesigned graph dictionary and we use the matching pursuit algorithm to choose one atom to localize an activated piece. Trend filtering on graphs and graph Laplacian denoising are denoising algorithms, which solve the following two optimization problems, respectively,

$$\begin{aligned} \text{TF}(\lambda): &\quad \min_{\mathbf{t}} \|\mathbf{x} - \mathbf{t}\|_2^2 + \lambda \|\Delta \mathbf{t}\|_1, \\ \text{gLap}(\lambda): &\quad \min_{\mathbf{t}} \|\mathbf{x} - \mathbf{t}\|_2^2 + \lambda \mathbf{t}^T \,\mathrm{L}\, \mathbf{t}, \end{aligned}$$

where λ is a tuning parameter, Δ is the graph incidence matrix (2.5) and L is the graph Laplacian matrix. $\|\Delta \mathbf{t}\|_1$ promotes localized adaptivity and $\mathbf{t}^T \,\mathrm{L}\, \mathbf{t}$ promotes smoothness. When λ is small, both solutions are close to the noisy graph signal; when λ is large, both solutions are regularized to be either localized or smooth. We expect that a small λ works better in a noiseless case and a large λ works better in a noisy case. After obtaining the denoised solution $\mathbf{t}^*$, we implement hard thresholding with threshold $\mu/2 = 0.5$ to localize the activated piece. To test the sensitivity to the tuning parameter, we also vary λ as either 0.5 or 5 in our experiments. Trend filtering on graphs and graph Laplacian denoising involve two tuning parameters: regularization parameter λ and threshold; however, both methods do not provide a principle to choose those parameters in practice. On the other hand, the proposed localization method (10.10) is parameter-free.

We observe that: (1) Cut-based localization provides the most robust performances; hard thresholding is a special case of cut-based localization, path-based localization is designed for capturing elongated-shaped pieces, LSPC is a data-independent dictionary and cannot adapt the shape of its atom to the given piece, Graph Laplacian denoising with careful parameter selection works well for large activated pieces, but it still fails to localize small activated pieces as the smooth assumption cannot capture localized variation. Trending filtering on graphs works well in denoising of a piecewise-constant graph signal; however, it may not work well in localization as the output is real and the threshold is hard to decide. Here the threshold is $\mu/2 = 0.5$, which sometimes works well and sometimes rules out the entire output. We also see that both graph Laplacian denoising and trending filtering on graphs are sensitive to the tuning parameter λ. Comparing to other methods, the advantages of cut-based localization are parameter-free, scalable and robust to noise. (2) The noise level influences the localization performance; as it grows, the F_1 score decreases and the Hamming distance increases. (3) The size of the activated piece influences the localization; localizing a piece with radius 10 is easier than localizing a piece with radius 5; this was also observed in [192].

Elongated-shape class. To generate an elongated path, we randomly choose two nodes as starting and ending nodes and compute the shortest path between these two. We look at two categories of path lengths: shorter than 15 and longer than 80 [1]. Figures 10.3(a)–(b) show examples of paths with lengths 15 and 91 (left and right columns, respectively). We aim to localize the activated pieces from noisy path graph signals. We vary the noise variance σ^2 from 0.1 to 1 with interval 0.1 and at each randomly generate $1,000$ noisy path graph signals at each noise level. We measure the quality of the localization by the Hamming distance and F_1 score.

Elongated-shape class. To generate an elongated path, we randomly choose two nodes as starting and ending nodes and compute the shortest path between these two. We look at two categories of path lengths: the first class has length between 10 to 15 and the second class has length longer than 80 [2]. Figures 10.3(a)–(b) show examples of paths with lengths 15 and 91 (left and right columns, respectively). We aim to localize activated paths with activated magnitude $\mu = 1$ from noisy path graph signals. We vary the noise variance σ^2 from 0.1 to 1 with interval 0.1 and at each randomly generate $1,000$ noisy path graph signals at each noise level. We measure the quality of the localization by the Hamming distance and F_1 score.

Figures 10.3(c)–(d) show the F_1 scores and Figures 10.3(e)–(f) the Hamming distances of localizing elongated-shaped pieces with lengths 15 and 91 (left and right columns, respectively) as functions of the noise variance. We compare the same six methods as in Figure 10.2: hard thresholding (grey-solid line), path-based localization (10.9) (yellow-circle line), cut-based localization (10.5b) (purple-square line), combined localization (10.10) (red-solid line), local-set-based piecewise-constant (LSPC, green-square line), trending filtering on graphs (TF($\lambda = 0.5$) light-blue-circle line, TF($\lambda = 5$) dark-blue-dashed line) and graph Laplacian denoising (gLap($\lambda = 0.5$) light-brown-circle line, gLap($\lambda = 5$) dark-brown-dashed line).

We observe that: (1) Path-based localization performs the best, which is also parameter-free, scalable and robust to noise. (2) When the path is short, cut-based localization and path-based localization provide similar performance; when the path is long, path-based localization

[1] The path length is measured by the geodesic distance between the two end nodes.

[2] The path length is measured by the geodesic distance between the two end nodes.

significantly outperforms cut-based localization. (3) The noise level influences the localization performance; as it grows, the F_1 score decreases and the Hamming distance increases. (4) The path length influences the localization performance. Similarly to what we observed in the ball-shaped case, it is much harder to localize a short path because a small activation provides less information.

10.4.2 Manhattan street network

We model the Manhattan street network as a graph with intersections as nodes and city streets as edges. The graph includes $13,679$ nodes and $17,163$ undirected edges. We generate two classes of one-piece graph signals: Central Park and 5th Avenue. Central Park is considered a ball-shaped piece and 5th Avenue is considered a path. We then compare the results with those from the Minnesota road network to validate our conclusions.

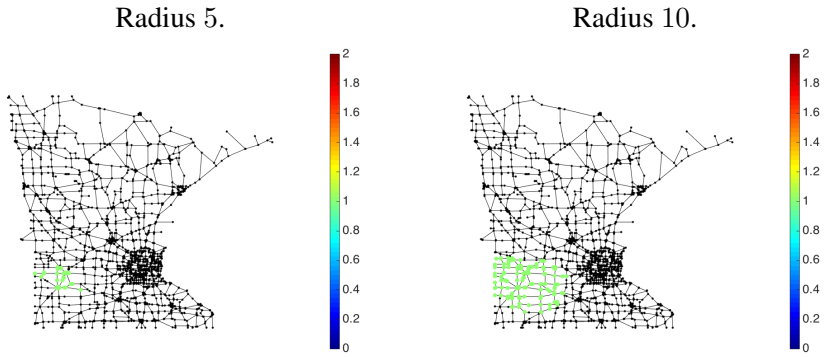
Central Park. Figures 10.4(a)–(d) show a piece activating the nodes in Central Park, a noisy version with noise variance $\sigma^2 = 1$, the activated piece provided by cut-based localization and the activated piece provided by path-based localization, respectively. We see that, even when the graph signal is corrupted by a high level of noise, cut-based localization still provides accurate localization, while path-based localization fails to localize Central Park. Figures 10.4(e)–(f) show the F_1 score and Hamming distance when localizing Central Park as a function of the noise level, respectively. The results are averaged over $1,000$ runs. We see that, when the noise level is low, cut-based localization significantly outperforms the other methods, which is consistent with what we observed with the Minnesota road network. LSPC is less sensitive to noise and slightly outperforms the other methods when the noise level is high. Since LSPC is data-independent, it chooses the most relevant predesigned atom to fit a noisy signal; on the other hand, as a data-adaptive method, cut-based localization designs an atom from the noisy signal and it easily fits noisy data. Overall, the localization performance of LSPC highly depends on the shape of its atoms: when LSPC has a predesigned atom matching the ground truth, it is robust to noise and provides effective localization performance; when LSPC does not have a predesigned atom matching the ground truth, it fails to localize well.

5th Avenue. Figures 10.5(a)–(d) show a piece activating the nodes along 5th Avenue, a noisy version with noise variance $\sigma^2 = 1$, the activated piece provided by cut-based localization and the activated piece provided by path-based localization, respectively. When the graph signal is corrupted by a high level of noise, both cut-based localization and path-based localization fail to localize 5th Avenue. Figures 10.5(e)–(f) show the F_1 score and Hamming distance when localizing 5th Avenue as a function of the noise level, respectively. The results are averaged over $1,000$ runs. We see that the path-based localization significantly outperforms all the other localization methods under various noise levels. This is similar to what we observed with the Minnesota road network.

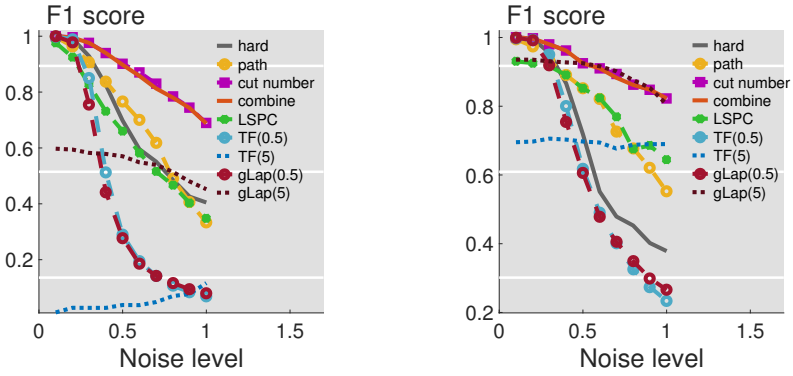
10.5 Conclusions

We aimed to find the supports of localized activated patterns on graphs. Its counterpart problems in classical signal/image processing, such as impluse detection, foreground detection and

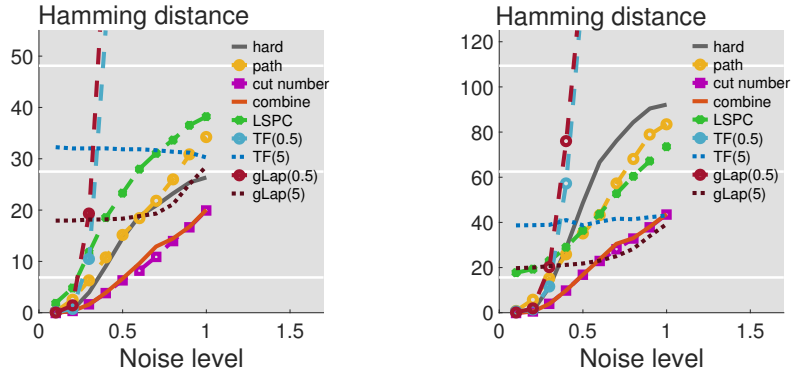
wavelet construction, are intensely studied over the past few decades. We modeled localized patterns by piecewise-constant graph signals, where each piece indicates a localized pattern that exhibits homogeneous internal behavior and the number of pieces indicates the number of localized patterns. We proposed a specific graph signal model, an optimization problem and a computationally efficient solver. The proposed solvers directly find the the supports of arbitrary localized activated patterns without tuning any threshold, which is a notorious issue in many localization problems. We then conducted an extensive empirical study to validate the proposed methods on simulated data. The results show the effectiveness and robustness of the proposed methods.



(a)–(b) Ball-shaped piece.

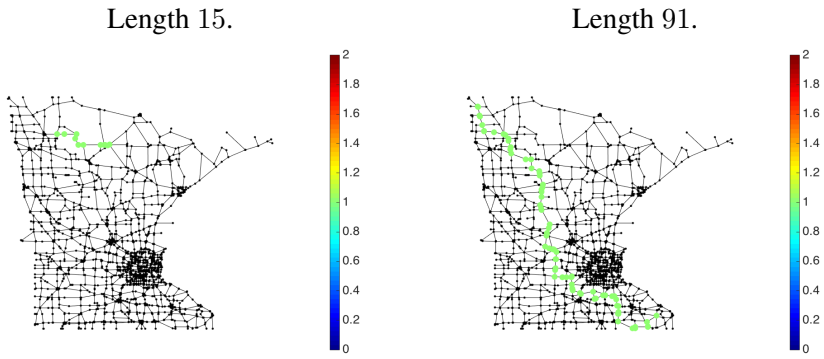


(c)–(d) F_1 score.

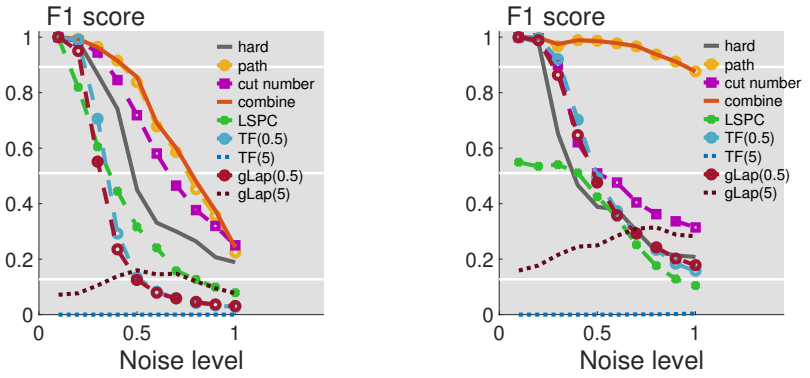


(e)–(f) Hamming distance.

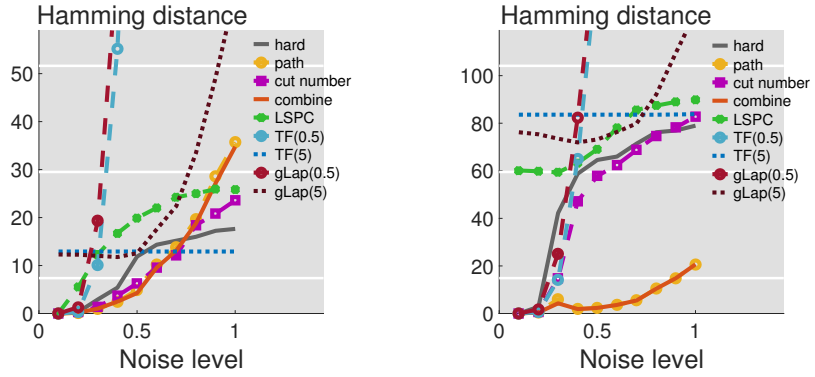
Figure 10.2: Localizing ball-shaped pieces in the Minnesota road network as a function of the noise level using hard thresholding (blue dashed line), cut-based localization (red solid line), path-based localization (yellow-circle line) and local-set-based piecewise-constant (purple-square line). Cut-based localization provides the best performance.



(a)–(b) Path.

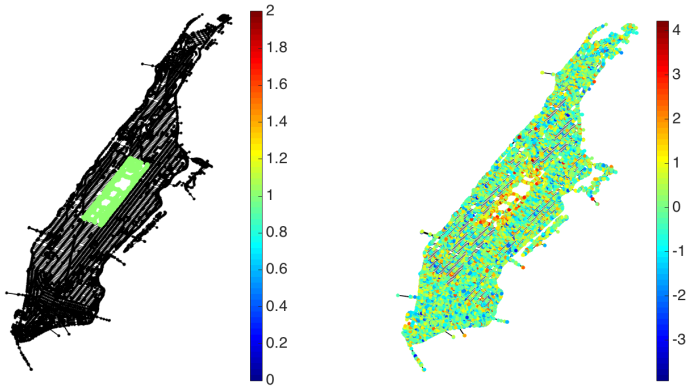


(c)–(d) F_1 score.



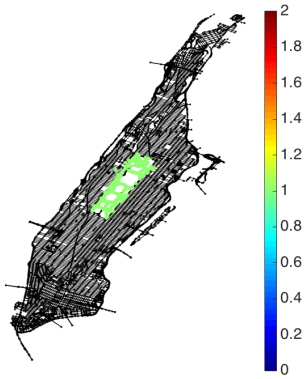
(e)–(f) Hamming distance.

Figure 10.3: Localizing paths in the Minnesota road network as a function of noise level using hard thresholding (blue dashed line), cut-based localization (red solid line), path-based localization (yellow-circle line) and local-set-based piecewise-constant (purple-square line). Path-based localization provides the best performance.

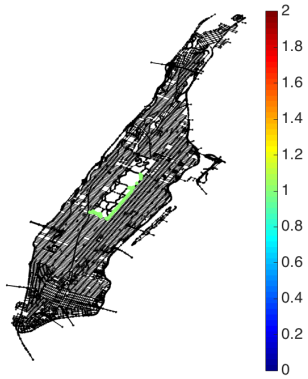


(a) Signal.

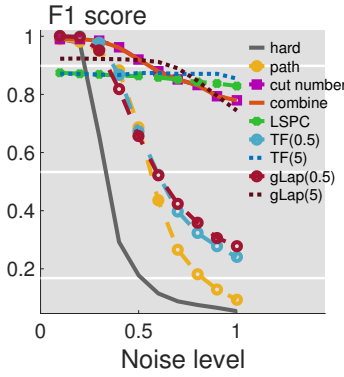
(b) Noisy signal.



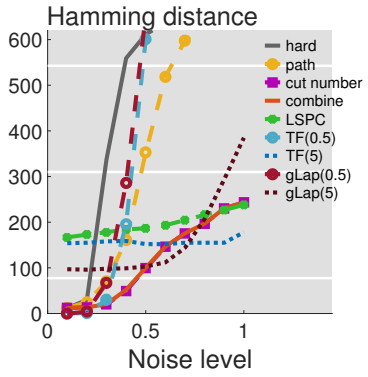
(c) Activated piece (cut).



(d) Activated piece (path).

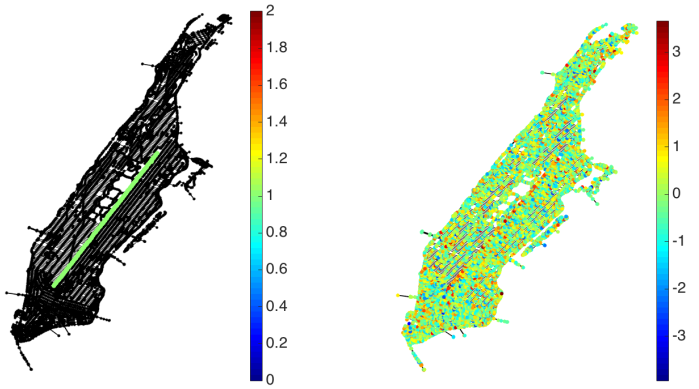


(e) F1 score.



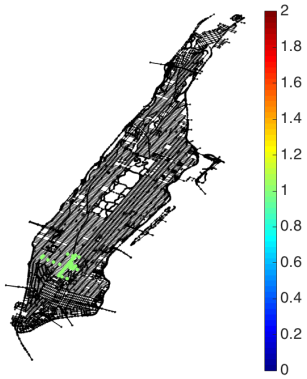
(f) Hamming distance.

Figure 10.4: Localizing (a) Central Park in the Manhattan street network as a function of the noise level from (b) its noisy version. (c) Activated piece obtained by cut-based localization with $F_1 = 0.81$ and $d_{\mathrm{H}} = 214$. (d) Activated piece obtained by path-based localization with $F_1 = 0.24$ and $d_{\mathrm{H}} = 586$. Cut-based localization outperforms path-based localization.

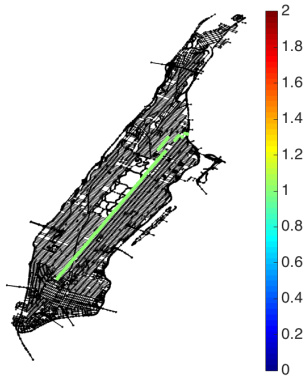


(a) Signal.

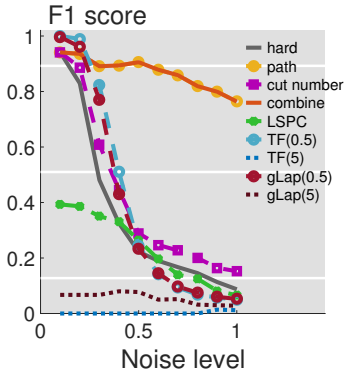
(b) Noisy signal.



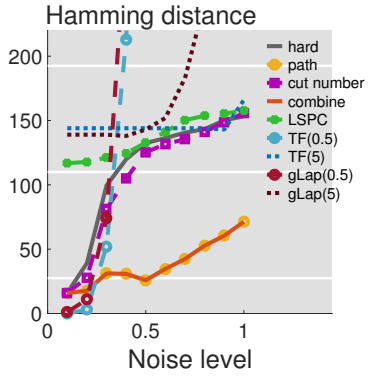
(c) Activated piece (cut).



(d) Activated piece (path).



(e) F1 score.



(f) Hamming distance.

Figure 10.5: Localizing (a) 5th Avenue in the Manhattan street network as a function of the noise level from (b) its noisy version. (c) Activated piece obtained by cut-based localization with $F_1 = 0.24$ and $d_{\mathrm{H}} = 144$. (d) Activated piece obtained by path-based localization with $F_1 = 0.85$ and $d_{\mathrm{H}} = 43$. Path-based localization outperforms cut-based localization.

Part V

Case Studies

Chapter 11

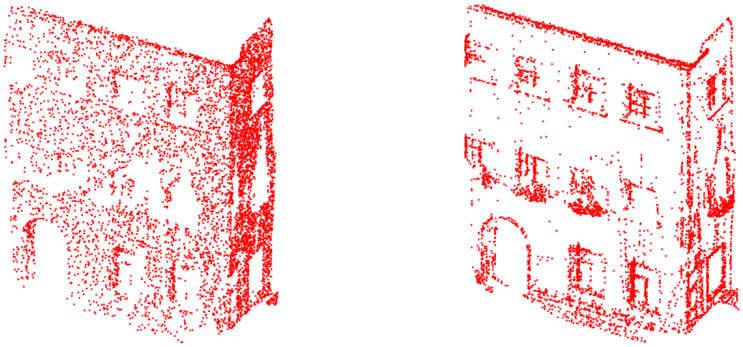
3D Point Cloud Processing

11.1 Introduction

With the recent development of 3D sensing technologies, 3D point clouds have become an important and practical representation of 3D objects and surrounding environments in many applications, such as virtual reality, mobile mapping, scanning of historical artifacts, 3D printing and digital elevation models [197]. A large number of 3D points on an object's surface measured by a sensing device are called a *3D point cloud*. Other than 3D coordinates, a 3D point cloud may also comprise some attributes, such as color, temperature and texture. Based on storage order and spatial connectivity between 3D points, there are two types of point clouds: *organized* point clouds and *unorganized* point clouds [198]. 3D points collected by a camera-like 3D sensor or a 3D laser scanner are typically arranged on a grid, like pixels in an image; we call those point clouds organized. For complex objects, we need to scan these objects from multiple view points and merge all collected points, which intermingles the order of sensing; we call those point clouds unorganized. It is easier to process an organized point cloud than an unorganized point cloud as the underlying grid produces a natural spatial connectivity and reflects the order of sensing. To make it general, we consider unorganized point clouds in this paper.

3D point cloud processing has become an important component in many 3D imaging and vision systems. It broadly includes compression [199–202], visualization [203, 204], surface reconstruction [205, 206], rendering [207, 208], editing [209, 210] and feature extraction [211–215]. A challenge in 3D point cloud processing is how to handle a large number of incoming 3D points [216, 217]. In many applications, such as digital documentation of historical buildings and terrain visualization, we need to store billions of incoming 3D points; additionally, real-time sensing systems generate millions of data points per second. A large-scale point cloud makes storage and subsequent processing inefficient.

To solve this problem, an approach is to consider efficient data structures to represent 3D point clouds. For example, [218, 219] partitions the 3D space into voxels and discretizes point clouds over voxels; a drawback is that to achieve a fine resolution, a dense grid is required, which causes space inefficiency. [220, 221] presents an octree representation of point clouds, which is space efficient, but suffers from discretization errors. [222, 223] presents a probabilistic generative model to model the distribution of point clouds; drawbacks are that those parametric models



(a) Uniform resampling. (b) Contour-enhanced resampling.

Figure 11.1: Proposed resampling strategy enhances contours of a point cloud. Plots (a) and (b) resamples 2% points from a 3D point cloud of a building containing $381,903$ points. Plot (b) is more visual-friendly than Plot (a). Note that the proposed resampling strategy is able to to enhance any information depending on users' preferences.

may not capture the true surface, and it is inefficient to infer parameters in the probabilistic generative model.

Another approach is to consider reducing the number of points through mesh simplification. The main idea is to construct a triangular or polygonal mesh for 3D point clouds, where nodes are 3D points (need not be from the input points) and edges are connectivities between those points respecting certain restrictions (e.g., belonging to a manifold). The mesh is simplified by reducing the number of nodes or edges; that is, several nodes are merged into one node with local structure preserved. Surveys of many such methods can be found in [224–226]. Drawbacks of this approach are that mesh construction requires costly computation, and mesh simplification changes the positions of original points, which causes distortion.

In this chapter, we consider resampling 3D point clouds; that is, we design application-dependent resampling strategies to preserve application-dependent information. For example, conventional contour detection in 3D point clouds requires careful and costly computation to obtain surface normals and classification models [223, 227]. We efficiently resample a small subset of points that is sensitive to the required contour information, making the subsequent processing cheaper without losing accuracy; see Figure 11.1 for an example. Since the original 3D point cloud is sampled from an object, we call this task *resampling*. This approach reduces the number of 3D points without changing the locations of original 3D points. After resampling, we unavoidably lose information in the original 3D point cloud. Our goal here is to design application-dependent resampling strategies to preserve application-dependent information.

We use a graph to capture local dependencies among points, representing a discrete version of the surface of an original object. The advantage of using a graph is to capture both local and global structure of point clouds. Each of the 3D coordinates and other attributes associated with 3D points is a graph signal indexed by the nodes of the underlying graph. We thus formulate a

resampling problem as graph signal sampling. However, graph sampling methods usually select samples in a deterministic fashion, which solve nonconvex optimization problems to obtain samples sequentially and require costly computation [45, 62, 90, 228]. To reduce the computational cost, we propose an efficient randomized resampling strategy to select a subset of points. The main idea is to generate subsamples according to a non-uniform resampling distribution, which is both fast and provably preserves application-dependent information in the original 3D point cloud.

11.2 Problem Formulation

In this section, we start with formulating a task of resampling a 3D point cloud. We then introduce graph signal processing, which lays a foundation for our proposed methods.

11.2.1 Resampling a Point Cloud

We consider a matrix representation of a point cloud with N points and K attributes,

$$\mathrm{X} \;=\; \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_K \end{bmatrix} \;=\; \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} \;\in\; \mathbb{R}^{N\times K}, \tag{11.1}$$

where $\mathbf{s}_i \in \mathbb{R}^N$ represents the ith attribute and $\mathbf{x}_i \in \mathbb{R}^K$ represents the ith point. Depending on sensing device, attributes can be 3D coordinates, RGB colors, textures, and many others. To distinguish 3D coordinates and other attributes, $\mathrm{X}_{\mathrm{c}} \in \mathbb{R}^{N\times 3}$ represents 3D coordinates and $\mathrm{X}_{\mathrm{o}} \in \mathbb{R}^{N\times(K-3)}$ represents other attributes.

The number of points N is usually large. For example, a 3D scan of a building usually needs billions of 3D points. It is challenging to work with a large-scale point cloud from both storage and data analysis perspectives. In many applications, however, we are interested in a subset of 3D points with particular properties, such as key points in point cloud registration and contour points in contour detection. To reduce the storage and computational cost, we consider resampling a subset of representative 3D points from the original 3D point cloud to reduce the scale. The procedure of resampling is to resample M $(M < N)$ points from a point cloud, or select M rows from the point cloud matrix X. The resampled point cloud is

$$\mathrm{X}_{\mathcal{M}} \;=\; \Psi\,\mathrm{X} \;\in \mathbb{R}^{M\times K}, \tag{11.2}$$

where $\mathcal{M} = (\mathcal{M}_1, \dots, \mathcal{M}_M)$ denotes the sequence of resampled indices, called *resampled set*, $\mathcal{M}_i \in \{1, \dots, N\}$ with $|\mathcal{M}| = M$ and the resampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$ as defined in (5.1).

The efficiency of the proposed resampling strategy is critical. Since we work with a large-scale point cloud, we want to avoid expensive computation. To implement resampling in an efficient way, we consider a randomized resampling strategy. It means that the resampled indices

are chosen according to a resampling distribution. Let $\{\pi_i\}_{i=1}^N$ be a series of resampling probabilities, where π_i denotes the probability to select the ith sample in each random trial. Once the resampling distribution is chosen, it is efficient to generate samples. The goal here is to find a resampling distribution that preserves information in the original point cloud.

The invariant property of the proposed resampling strategy is also critical. When we shift, rotate or scale a point cloud, the intrinsic distribution of 3D points does not changed and the proposed resampling strategy should not change.

Definition 22. A resampling strategy is shift-invariant when a sampling distribution π is designed for a point cloud, $\mathrm{X} = \begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}$, then the same sampling distribution π is designed for its shifted point cloud, $\begin{bmatrix}\mathrm{X_c} + \mathbf{1}\mathbf{a}^T & \mathrm{X_o}\end{bmatrix}$, where $\mathbf{a} \in \mathbb{R}^3$.

Definition 23. A resampling strategy is rotation-invariant when a sampling distribution π is designed for a point cloud, $\mathrm{X} = \begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}$, then the same sampling distribution π is designed for its rotated point cloud, $\begin{bmatrix}\mathrm{X_c}\,\mathrm{R} & \mathrm{X_o}\end{bmatrix}$, where $\mathrm{R} \in \mathbb{R}^{3\times 3}$ is a 3D rotation matrix.

Definition 24. A resampling strategy is scale-invariant when a sampling distribution π is designed for a point cloud, $\mathrm{X} = \begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}$, then the same sampling distribution π is designed for its rotated point cloud, $\begin{bmatrix}c\,\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}$, where constant $c > 0$.

Our aim is to guarantee that the proposed resampling strategy is shift, rotation and scale invariant.

11.3 Methodology

11.3.1 Resampling based on Feature Extraction

During resampling, we reduce the number of points and unavoidably lose information in a point cloud. Our goal is to design an application-dependent resampling strategy, preserving selected information depending on particular needs. Those information are described by features. When detecting contours, we usually need careful and intensive computation, such as calculating surface normals and classifying points [223,227]. Instead of working with a large number of points, we consider efficiently sampling a small subset of points that captures the required contour information, making the subsequent computation much cheaper without losing contour information. We also need to guarantee that the proposed resampling strategy is shift/rotation/scale-invariant for robustness. We will show that some features naturally provide invariance and other may not. We will handle the invariance by considering a general objective function.

Feature-Extraction based Formulation

Let $f(\cdot)$ be a feature-extraction operator that extracts targeted information from a point cloud according to particular needs; that is, the features $f(\mathrm{X}) \in \mathbb{R}^{N\times K}$ are extracted from a point cloud $\mathrm{X} \in \mathbb{R}^{N\times K}$[1]. Depending on an application, those features can be edges, key points and flatness [212–215, 229]. In this section, we consider feature-extraction operator at an abstract level and use graph filters to implement a feature-extraction operator in the next section.

[1] For simplicity, we consider the number of features to be the same as the number of attributes. The proposed method also works when the number of features and the number of attributes are different.

To evaluate the performance of a resampling operator, we quantify how much features are lost during resampling; that is, we sample features, and then interpolate to get back original features. The features are considered to reflect the targeted information contained in each 3D point. The performance is better when the recovery error is smaller. Mathematically, we resample a point cloud M times. At the jth step, we independently choose a point $\mathcal{M}_j = i$ with probability π_i. Let $\Psi \in \mathbb{R}^{M\times N}$ be the resampling operator (5.1) and $\mathrm{S} \in \mathbb{R}^{N\times N}$ be a diagonal rescaling matrix with $\mathrm{S}_{i,i} = 1/\sqrt{M\pi_i}$. We quantify the performance of a resampling operator as follows:

$$D_{f(\mathrm{X})}(\Psi) \;=\; \left\|\mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - f(\mathrm{X})\right\|_2^2, \tag{11.3}$$

where $\|\cdot\|_2$ is the spectral norm. $\Psi^T\Psi \in \mathbb{R}^{N\times N}$ is a zero-padding operator, which a diagonal matrix with diagonal elements $(\Psi^T\Psi)_{i,i} > 0$ when the ith point is sampled, and 0, otherwise. The zero-padding operator $\Psi^T\Psi$ ensures the resampled points and the original point cloud have the same size. S is used to compensate non-uniform weights during resampling. $\mathrm{S}\,\Psi^T$ is the most naive interpolation operator that reconstructs the original feature $f(\mathrm{X})$ from its resampled version $\Psi f(\mathrm{X})$ and $\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})$ represents the preserved features after resampling in a zero-padding form. Lemma 7 shows that S aids to provide an unbiased estimator.

Lemma 7. Let $f(\mathrm{X}) \in \mathbb{R}^{N\times K}$ be features extracted from a point cloud X. Then,

$$\begin{aligned}\mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right) &\propto \pi \odot f(\mathrm{X}),\\ \mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right) &= f(\mathrm{X}),\end{aligned}$$

where $\mathbb{E}_{\Psi\sim\pi}$ means the expectation over samples, which are generated from a distribution Π independently and randomly, and $\odot$ is row-wise multiplication.

See Appendix 14.8 for the proof of these results.

The evaluation metric $D_{f(\mathrm{X})}(\Psi)$ measures the reconstruction error; that is, how much feature information is lost after resampling without using sophisticated interpolation operator. When $D_{f(\mathrm{X})}(\Psi)$ is small, preserved features after resampling are close to the original features, meaning that little information is lost. The expectation $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ is the expected error caused by resampling and quantifies the performance of a resampling distribution π. Our goal is to minimize $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ over π to obtain an optimal resampling distribution in terms of preserving features $f(\mathrm{X})$. We now derive the mean square error of the objective function (11.3).

Theorem 24. The mean square error of the objective function (11.3) is

$$\mathbb{E}_{\Psi\sim\pi} D_{f(\mathrm{X})}(\Psi) \;=\; \mathrm{Tr}\left(f(\mathrm{X})\,\mathrm{Q}\,f(\mathrm{X})^T\right), \tag{11.4}$$

where $\mathrm{Q} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with $\mathrm{Q}_{i,i} = 1/\pi_i - 1$.

See Appendix 14.9 for the proof of these results.

We now consider the invariance property of resampling. The sufficient condition for the shift/rotation/scale-invariance of a resampling strategy is that the evaluation metric (11.3) be shift/ rotation/scale-invariance. Recall that a 3D point cloud is $\mathrm{X} = \begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}$, where $\mathrm{X_c} \in \mathbb{R}^{N\times 3}$ represent 3D coordinates and $\mathrm{X_o} \in \mathbb{R}^{N\times(K-3)}$ to represent other attributes.

Definition 25. A feature-extraction operator $f(\cdot)$ is shift-invariant when the features extracted from a point cloud and its shifted version are same; that is, $f(\begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}) = f(\begin{bmatrix}\mathrm{X_c} + \mathbf{1}\mathbf{a}^T & \mathrm{X_o}\end{bmatrix})$ with shift $\mathbf{a} \in \mathbb{R}^3$.

Definition 26. A feature-extraction operator $f(\cdot)$ is rotation-invariant when the features extracted from a point cloud and its rotated version are same; that is, $f(\begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}) = f(\begin{bmatrix}\mathrm{X_c\,R} & \mathrm{X_o}\end{bmatrix})$ with $\mathrm{R} \in \mathbb{R}^{3\times 3}$ is a 3D rotation matrix.

Definition 27. A feature-extraction operator $f(\cdot)$ is scale-invariant when features extracted from a point cloud and its scaled version are same; that is, $f(\begin{bmatrix}\mathrm{X_c} & \mathrm{X_o}\end{bmatrix}) = f(\begin{bmatrix}c\,\mathrm{X_c} & \mathrm{X_o}\end{bmatrix})$ with constant $c > 0$.

When $f(\cdot)$ is shift/rotation/scale-invariant, (11.3) does not change through shifting, rotating or scaling, leading to a shift/rotation/scale-invariant resampling strategy and it is sufficient to minimize $\mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right)$ to obtain a resampling strategy; however, when $f(\cdot)$ is shift/rotation/scale-variance, (11.3) may change through shifting, rotating or scaling, leading to a shift/rotation/scale-variant resampling strategy.

To handle shift variance, we can recenter a point cloud to the origin before processing; that is, we normalize the mean of 3D coordinates to zeros. To handle scale variance, we can normalize the magnitude of the 3D coordinates before processing; that is, we normalize the spectral norm $\|\mathrm{X}_c\|_2 = c$ with constant $c > 0$. The choice of c depends on users' preference and we will show that c is a trade-off between 3D coordinates and the values of other attributes. From now on, we first recenter a point cloud to the origin and then normalize its magnitude to guarantee the shift/scale invariance of any 3D point cloud.

To handle rotation variance of $f(\cdot)$, we consider the following evaluation metric:

$$\begin{aligned} D_f(\Psi) &= \max_{\mathrm{X}'_c:\|\mathrm{X}'_c\|_2=c} D_{f\left(\begin{bmatrix}\mathrm{X}'_c & \mathrm{X_o}\end{bmatrix}\right)}(\Psi) \\ &= \max_{\mathrm{X}'_c:\|\mathrm{X}'_c\|_2=c} \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right) f\left(\begin{bmatrix}\mathrm{X}'_c & \mathrm{X_o}\end{bmatrix}\right)\right\|_F^2, \end{aligned} \tag{11.5}$$

where constant $c = \|\mathrm{X}_c\|_2$ is the normalized spectral norm of 3D coordinates.

Unlike $D_{f(\mathrm{X})}(\Psi)$ (11.3), to remove the influence of rotation, the evaluation metric $D_f(\Psi)$ considers the worst possible reconstruction error caused by rotation. In (11.5), we consider 3D coordinates as variables due to rotation. We constrain the spectral norm of 3D coordinates because a rotation matrix is orthornormal and the spectral norm of 3D coordinates does not change during rotation. We then minimize $\mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right)$ to obtain a rotation-invariant resampling strategy even when $f(\cdot)$ is rotation-variant.

For simplicity, we perform derivation for only linear feature-extraction operators. A linear feature-extraction operator $f(\cdot)$ is of the form of $f(\mathrm{X}) = \mathrm{F\,X}$, where X is a 3D point cloud and $\mathrm{F} \in \mathbb{R}^{N\times N}$ is a feature-extraction matrix.

Theorem 25. Let $f(\cdot)$ be a rotation-varying linear feature-extraction operator, where $f(\mathrm{X}) = \mathrm{F\,X}$ with $\mathrm{F} \in \mathbb{R}^{N\times N}$. The exact form of $\mathbb{E}_{\Psi\sim\pi} D_f(\Psi)$ is

$$\mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right) \;=\; c^2\mathrm{Tr}\left(\mathrm{F\,Q\,F}^T\right) + \mathrm{Tr}\left(\mathrm{F\,X_o\,Q(F\,X_o)}^T\right), \tag{11.6}$$

where $\mathrm{Q} \in \mathbb{R}^{N\times N}$ is a diagonal matrix with $\mathrm{Q}_{i,i} = 1/\pi_i - 1$.

See Appendix 14.10 for the proof of these results.

Optimal Resampling Distribution

We now derive the optimal resampling distributions by minimizing the reconstruction error. For a rotation-invariant feature-extraction operator, we minimize (11.4).

Theorem 26. Let $f(\cdot)$ be a rotation-invariant feature-extraction operator. The corresponding optimal resampling strategy π^* is,

$$\pi_i^* \propto \|f_i(\mathrm{X})\|_2\,, \tag{11.7}$$

where $f_i(\mathrm{X}) \in \mathbb{R}^K$ is the ith row of $f(\mathrm{X})$.

See Appendix 14.11 for the proof of these results.

We see that the optimal resampling distribution is proportional to the magnitude of features; that is, points associated with high magnitudes have high probability to be selected, while points associated with small magnitudes have small probability to be selected. The intuition is that the response after the feature-exaction operator reflects the information contained in each 3D point and determines the resampling probability of each 3D point.

For a rotation-variant linear feature-extraction operator, we minimize (11.6).

Theorem 27. Let $f(\cdot)$ be a rotation-variant linear feature-extraction operator, where $f(\mathrm{X}) = \mathrm{F}\,\mathrm{X}$ with $\mathrm{F} \in \mathbb{R}^{N\times N}$. The corresponding optimal resampling strategy π^* is,

$$\pi_i^* \propto \sqrt{c^2\,\|\mathrm{F}_i\|_2^2 + \|(\mathrm{F}\,\mathrm{X}_\mathrm{o})_i\|_2^2}, \tag{11.8}$$

where constant $c = \|\mathrm{X}_\mathrm{c}\|_2$, F_i is the ith row of F and $(\mathrm{F}\,\mathrm{X}_\mathrm{o})_i$ is the ith row of $\mathrm{F}\,\mathrm{X}_\mathrm{o}$.

See Appendix 14.12 for the proof of these results.

We see that the optimal resampling distribution is also proportional to the magnitude of features. The feature comes from two sources: 3D coordinates and the other attributes. The tuning parameter c in (11.8) is the normalized spectral norm used to remove the scale variance. The choice of c trade-offs the contribution from 3D coordinates and the other attributes.

11.3.2 Resampling based on Graph Filtering

The previous section studied resampling based on an arbitrary feature-extraction operator. In this section, we design graph filters to efficiently extract features from a point cloud. Let features extracted from a point cloud X be

$$f(\mathrm{X}) \;=\; h(\mathrm{A})\,\mathrm{X} \;=\; \sum_{\ell=0}^{L-1} h_\ell\,\mathrm{A}^\ell\,\mathrm{X},$$

which follows from the definition of graph filters (2.10). Since a graph filter is a linear operator, the corresponding optimal resampling distribution follows from the results in Theorems 26 and 27 by replacing $\mathrm{F} = \sum_{\ell=0}^{L-1} h_\ell\,\mathrm{A}^\ell$. All graph filtering-based feature-extraction operators are scale-variant due to linearity. As discussed earlier, we can normalize the spectral norm of a 3D coordinates to handle this issue. We thus will not discuss scale invariance in this section. We will see that by carefully using the graph shift operator A and filter coefficients h_is, a graph filtering-based feature-extraction operator may be shift or rotation varying.

Similarly to filter design in classical signal processing, we design a graph filter either in the graph vertex domain or in the graph spectral domain. In the graph vertex domain, for each point, a graph filter averages the attributes of its local points. For example, the output of the ith point, $f_i(\mathrm{X}) = \sum_{\ell=0}^{L-1} h_\ell \left(\mathrm{A}^\ell \mathrm{X}\right)_i$ is a weighted average of the attributes of points that are within L hops away from the ith point. The ℓth graph filter coefficient, h_ℓ, quantifies the contribution from the ℓth-hop neighbors. We design the filter coefficients to change the weights in local averaging.

In the graph spectral domain, we first design a graph spectrum distribution and then use graph filter coefficients to fit this distribution. For example, a graph filter with length L is

$$\begin{aligned} h(\mathrm{A}) \;&=\; \mathrm{V}\, h(\Lambda)\, \mathrm{V}^{-1} \\ &=\; \mathrm{V} \begin{bmatrix} \sum_{\ell=0}^{L-1} h_\ell \lambda_1^\ell & 0 & \cdots & 0 \\ 0 & \sum_{\ell=0}^{L-1} h_\ell \lambda_2^\ell & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{\ell=0}^{L-1} h_\ell \lambda_N^\ell \end{bmatrix} \mathrm{V}^{-1}, \end{aligned}$$

where V is the graph Fourier basis and λ_i are graph frequencies (2.6). When we want the response of the ith graph frequency to be c_i, we set

$$h(\lambda_i) \;=\; \sum_{\ell=0}^{L-1} h_\ell \lambda_i^\ell \;=\; c_i,$$

and solve a set of linear equations to obtain the graph filter coefficients h_ℓ. It is also possible to use the Chebyshev polynomial to design graph filter coefficients [37]. We now consider some special cases of graph filters.

All-pass Graph Filtering

Let $h(\lambda_i) = 1$; that is, $h(\mathrm{A}) = \mathbf{I}$ is an identity matrix with $h_0 = 1$ and $h_i = 0$ for $i = 1, \ldots, L-1$. The intuition behind this setting is that the original point cloud is trustworthy and all points are uniformly sampled from an object without noise, reflecting the true geometric structure of the object. We want to preserve all the information and the features are thus the original attributes themselves. Since $f(\mathrm{X}) = \mathrm{X}$, the feature-extraction operator $f(\cdot)$ is rotation-variant. Based on Theorem 27, the optimal resampling strategy is

$$\pi_i^* \propto \sqrt{c^2 + \|(\mathrm{X_o})_i\|_2^2}. \tag{11.9}$$

Here the feature-extraction matrix F in (11.7) is an identity matrix and the norm of each row of F is 1. When we only preserve 3D coordinates, we ignore the term of $\mathrm{X_o}$ and obtain a constant resampling probability for each point, meaning that uniform resampling is the optimal resampling strategy to preserve the overall geometry information.

High-pass Graph Filtering

In image processing, a high-pass filter is used to extract edges and contours. Similarly, we use a high-pass graph filter to extract contours in a point cloud. Here we only consider the 3D

coordinates as attributes ($\mathrm{X} = \mathrm{X}_{\mathrm{c}} = \mathbb{R}^{N\times 3}$), but the proposed method can be easily extended to other attributes.

A critical question is how to define contours in a 3D point cloud. We consider that contour points break the trend formed by its neighboring points and bring innovation. Many previous works need sophisticated geometry-related computation, such as surface normal, to detect contours [227]. Instead of measuring sophisticated geometry properties, we describe the possibility of being a contour point by the local variation on graphs, which is the response of high-pass graph filtering. The corresponding local variation of the ith point is

$$f_i(\mathrm{X}) = \| \left(h(\mathrm{A})\,\mathrm{X}\right)_i \|_2^2, \tag{11.10}$$

where $h(\mathrm{A})$ is a high-pass graph filter. The local variation $f(\mathrm{X}) \in \mathbb{R}^N$ quantifies the energy of response after high-pass graph filtering. The intuition behind this is that when the local variation of a point is high, its 3D coordinates cannot be well approximated from the 3D coordinates of its neighboring points; in other words, this point bring innovation by breaking the trend formed by its neighboring points and has a high possibility of being a contour point.

The following theorem shows that in general the local variation is rotation invariant, but shift variant.

Theorem 28. Let $f(\mathrm{X}) = \mathrm{diag}\left(h(\mathrm{A})\,\mathrm{X}\,\mathrm{X}^T\,h(\mathrm{A})^T\right) \in \mathbb{R}^N$, where $\mathrm{diag}(\cdot)$ extracts the diagonal elements. $f(\mathrm{X})$ is rotation invariant and shift invariant unless $h(\mathrm{A})\mathbf{1} = \mathbf{0} \in \mathbb{R}^N$.

See Appendix 14.13 for the proof of these results.

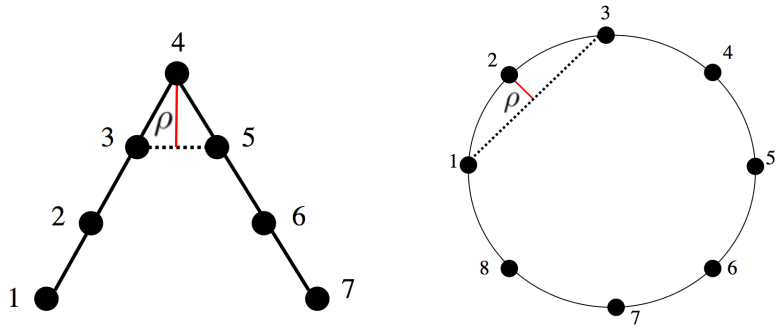
To guarantee local variation is naturally shift invariant without recentering a 3D point cloud, we simply use a transition matrix as a graph shift operator; that is, $\mathrm{A} = \mathrm{D}^{-1}\,\mathrm{W}$, where D is the diagonal degree matrix. The reason is that $\mathbf{1} \in \mathbb{R}^N$ is the eigenvector of a transition matrix, $\mathrm{A}\,\mathbf{1} = \mathrm{D}^{-1}\,\mathrm{W}\,\mathbf{1} = \mathbf{1}$. Thus,

$$h(\mathrm{A})\mathbf{1} = \sum_{\ell=0}^{N-1} h_\ell\,\mathrm{A}^\ell\,\mathbf{1} = \sum_{\ell=0}^{N-1} h_\ell \mathbf{1} = \mathbf{0},$$

when $\sum_{\ell=0}^{N-1} h_\ell = 0$. A simple design is a Haar-like high-pass graph filter

$$\begin{aligned} h_{\mathrm{HH}}(\mathrm{A}) &= \mathbf{I} - \mathrm{A} \\ &= \mathrm{V} \begin{bmatrix} 1-\lambda_1 & 0 & \cdots & 0 \\ 0 & 1-\lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1-\lambda_N \end{bmatrix} \mathrm{V}^{-1}, \end{aligned} \tag{11.11}$$

Note that $\lambda_{\max} = \max_i |\lambda_i| = 1$, where λ_i are eigenvalues of A, because the graph shift operator is a transition matrix. In this case, $h_0 = 1, h_1 = -1$ and $h_i = 0$ for all $i > 1$, $\sum_{\ell=0}^{N-1} h_\ell = 0$. Thus, a Haar-like high-pass graph filter is both shift and rotation invariant. The graph frequency response of a Haar-like high-pass graph filter is $h_{\mathrm{HH}}(\lambda_i) = 1 - \lambda_i$. Since the eigenvalues are ordered descendingly, we have $1 - \lambda_i \leq 1 - \lambda_{i+1}$, meaning low frequency response relatively attenuates and high frequency response relatively amplifies.



(a) Lines. (b) Circle.

Figure 11.2: Red line shows the local variation.

In the graph vertex domain, the response of the ith point is

$$\left(h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X}\right)_i \;=\; \mathbf{x}_i - \sum_{j\in\mathcal{N}_i} \mathrm{A}_{i,j}\,\mathbf{x}_j.$$

Because A is a transition matrix, $\sum_{j\in\mathcal{N}_i} \mathrm{A}_{i,j} = 1$ and $h_{\mathrm{HH}}(\mathrm{A})$ compares the difference between a point and the convex combination of its neighbors. The geometry interpretation of the proposed local variation is the Euclidean distance between the original point and the convex combination of its neighbors, reflecting how much information we know about a point from its neighbors. When the local variation of a point is large, the Euclidean distance between this point and the convex combination of its neighbors is long and this point provides a large amount of innovation.

We can verify the proposed local variation on some simple examples.

Example 1. When a point cloud form a 3D line, two endpoints belong to the contour.

Example 2. When a point cloud form a 3D polygon/polyhedron, the vertices (corner points) and the edges (line segment connecting two adjacent vertices) belong to the contour.

Example 3. When a point cloud form a 3D circle/sphere, there is no contour.

When the points are uniformly spread along the defined shape, the proposed local variation (11.10) satisfies Examples 1, 2 and 3 from the geometric perspective. For example, in Figure 11.2 (a), Point 2 is the convex combination of Points 1 and 3, and the local variation of Point 2 is thus zero. However, Point 4 is not the convex combination of Points 3 and 5 and the length of the red line indicates the local variation of Point 4. It turns out that only Points 1, 4 and 7 have nonzero local variation, which is what we expect. In Figure 11.2 (b), all the nodes are evenly spread on a circle and have the same amount of innovation, which is represented as a red line. Similar arguments show that the proposed local variation (11.10) satisfies Examples 1, 2 and 3.

The feature-extraction operator $f(\mathrm{X}) = \|h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X}\|_F^2$ is shift and rotation-invariant. Based on Theorem 26, the optimal resampling distribution is

$$\pi_i^* \;\propto\; \left\| \left(h_{\mathrm{HH}}(\mathrm{A})\,\mathrm{X}\right)_i \right\|_2^2 \;=\; \left\| \mathbf{x}_i - \sum_{j\in\mathcal{N}_i} \mathrm{A}_{i,j}\,\mathbf{x}_j \right\|_2^2, \tag{11.12}$$

where $\mathrm{A} = \mathrm{D}^{-1}\,\mathrm{W}$ is a transition matrix.

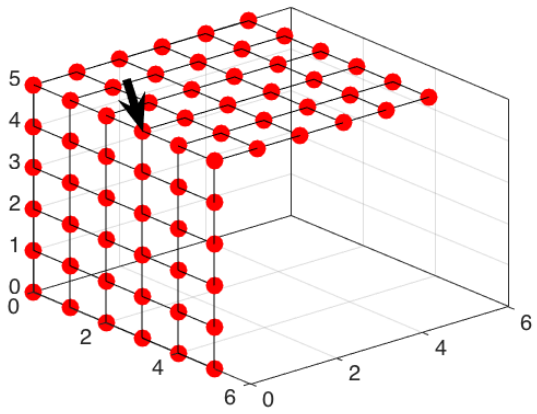


Figure 11.3: The pairwise difference based local variation cannot capture the contour points connecting two faces.

Note that the graph Laplacian matrix is commonly used to measure variations. Let $\mathrm{L} = \mathrm{D} - \mathrm{W} \in \mathbb{R}^{N\times N}$ be a graph Laplacian matrix. The graph Laplacian based total variation is

$$\mathrm{Tr}\left(\mathrm{X}^T\,\mathrm{L}\,\mathrm{X}\right) \;=\; \sum_i \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\,\|\mathbf{x}_i - \mathbf{x}_j\|_2^2\,. \tag{11.13}$$

where $\mathcal{N}_i$ is the neighbors of the ith node and the variation contributed by the ith point is

$$f_i(\mathrm{X}) \;=\; \sum_{j\in\mathcal{N}_i} \mathrm{W}_{i,j}\,\|\mathbf{x}_i - \mathbf{x}_j\|_2^2\,. \tag{11.14}$$

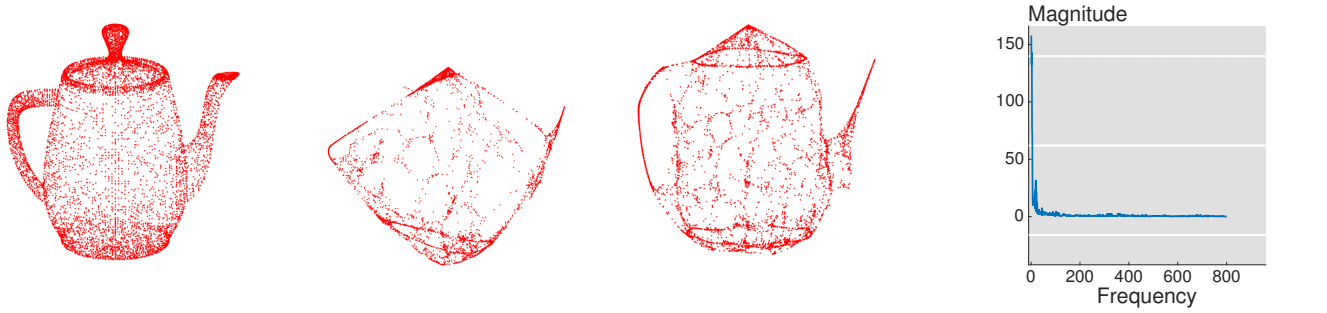
The variation here is defined based on the accumulation of pairwise differences. We call (11.14) pairwise difference based local variation.

The pairwise difference based local variation cannot capture geometry change and violates Example 2. We show a counter example in Figure 11.3. The points are uniformly spread along the faces of a cube and Figure 11.3 shows two faces. Each point connects to its adjacent four points with the same edge weight. The pairwise difference based local variations of all the points are the same, which means that there is no contour in this point cloud. However, the black arrow points to a point that should be a contour point.

Low-pass Graph Filtering

In classical signal processing, a low-pass filter is used to capture rough shape of a smooth signal and reduce noise. Similarly, we use a low-pass graph filter to capture rough shape of a point cloud and reduce sampling noise during resampling. Since we use the 3D coordinates of points to construct a graph, the 3D coordinates are naturally smooth on this graph, meaning that two adjacent points in the graph have similar coordinates in the 3D space. When a 3D point cloud

is corrupted by noises and outliers, a low-pass graph filter, as a denoising operator, uses local neighboring information to approximate a true position for each point. Since the output after low-pass graph filtering is a denoised version of the original point cloud, it is more appropriate to resample from denoised points than original points.



(a) Teapot. (b) Approximation with 10 graph frequencies. (c) Approximation with 100 graph frequencies. (d) Graph spectral distribution.

Figure 11.4: Low-pass approximation represents the main shape of the original point clouds. Plot (a) shows a point cloud with $8,000$ points representing a teapot. Plots (b) and (c) show the approximations with 10 and 100 graph frequencies. We see that the approximation with 10 graph frequencies shows a rough structure of a teapot and the approximation with 100 graph frequencies can be recognized as a teapot. Plot (d) shows the graph spectral distribution, which clearly shows that most energy is concentrated in the low-pass band.

Ideal low-pass graph filter. A straightforward choice is an ideal low-pass graph filter, which completely eliminates all graph frequencies above a given graph frequency while passing those below unchanged. An ideal low-pass graph filter with bandwidth b is

$$\begin{aligned} h_{\mathrm{IL}}(\mathrm{A}) &= \mathrm{V} \begin{bmatrix} \mathbf{I}_{b\times b} & \mathbf{0}_{b\times(N-b)} \\ \mathbf{0}_{(N-b)\times b} & \mathbf{0}_{(N-b)\times(N-b)} \end{bmatrix} \mathrm{V}^{-1} \\ &= \mathrm{V}_{(b)}\, \mathrm{V}_{(b)}^{T} \in \mathbb{R}^{N\times N}, \end{aligned}$$

where $\mathrm{V}_{(b)}$ is the first b columns of V, and the graph frequency response is

$$h_{\mathrm{IL}}(\lambda_i) = \begin{cases} 1, & i \le b; \\ 0, & \text{otherwise.} \end{cases} \tag{11.15}$$

The ideal low-pass graph filter h_{IL} projects an input graph signal into a bandlimited subspace [90] and $h_{\mathrm{IL}}(\mathrm{A})\mathbf{s}$ is a bandlimited approximation of the original graph signal $\mathbf{s}$. We show an example in Figure 11.4. Figures 11.4 (b) and (c) show that the bandlimited approximation of the 3D coordinates of a teapot gets better when the bandwidth b increases. We see that the bandwidth influence the shape of the teapot rapidly: with 10 graph frequencies, we only obtain a rough structure of the teapot. Figure 11.4 (d) shows that the main energy is concentrated in the low-pass graph frequency band.

The feature-extraction operator $f(\mathrm{X}) = \mathrm{V}_{(b)}\,\mathrm{V}_{(b)}^T\,\mathrm{X}$ is shift and rotation-varying. Based on Theorem 27, the corresponding optimal resampling strategy is

$$\begin{aligned}\pi_i^* &\propto \sqrt{c^2\left\|\left(\mathrm{V}_{(b)}\right)_i\right\|_2^2 + \left\|\left(\mathrm{V}_{(b)}\,\mathrm{V}_{(b)}^T\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2} \\ &= \sqrt{c^2\left\|\mathbf{v}_i\right\|_2^2 + \left\|\mathrm{X}_{\mathrm{o}}{}^T\,\mathrm{V}_{(b)}\,\mathbf{v}_i\right\|_2^2},\end{aligned} \tag{11.16}$$

where $\mathbf{v}_i \in \mathbb{R}^b$ is the ith row of $\mathrm{V}_{(b)}$.

A direct way to obtain $\|\mathbf{v}_i\|_2$ requires the truncated eigendecomposition (2.6), whose computational cost is $O(Nb^2)$, where b is the bandwidth. It is potentially possible to approximate the leverage scores through a fast algorithm [230, 231], where we use randomized techniques to avoid the eigendecomposition and the computational cost is $O(Nb\log(N))$. Another way to leverage computation is to partition a graph into several subgraphs and obtain leverage scores in each subgraph.

Haar-like low-pass graph filter. Another simple choice is Haar-like low-pass graph filter; that is,

$$\begin{aligned}h_{\mathrm{HL}}(\mathrm{A}) &= \mathbf{I} + \frac{1}{|\lambda_{\max}|}\,\mathrm{A} \\ &= \mathrm{V}\begin{bmatrix} 1+\frac{\lambda_1}{|\lambda_{\max}|} & 0 & \cdots & 0 \\ 0 & 1+\frac{\lambda_2}{|\lambda_{\max}|} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1+\frac{\lambda_N}{|\lambda_{\max}|} \end{bmatrix}\mathrm{V}^{-1},\end{aligned} \tag{11.17}$$

where $\lambda_{\max} = \max_i |\lambda_i|$ with λ_i eigenvalues of A. The normalization factor $\lambda_{\max}$ is presented to avoid the amplification of the magnitude. We denote $\mathrm{A}_{\mathrm{norm}} = \mathrm{A}\,/|\lambda_{\max}|$ for simplicity. The graph frequency response is $h_{\mathrm{HL}}(\lambda_i) = 1 + \lambda_i/|\lambda_{\max}|$. Since the eigenvalues are ordered in a descending order, we have $1 + \lambda_i \geq 1 + \lambda_{i+1}$, meaning low frequency response amplifies and high frequency response attenuates.

In the graph vertex domain, the response of the ith point is $(h_{\mathrm{HL}}(\mathrm{A})\,\mathrm{X})_i = \mathbf{x}_i + \sum_{j\in\mathcal{N}_i}(\mathrm{A}_{\mathrm{norm}})_{i,j}\mathbf{x}_j$, where $\mathcal{N}_i$ is the neighbors of the ith point. We see that $h_{\mathrm{HL}}(\mathrm{A})$ averages the attributes of each point and its neighbors to provide a smooth output.

The feature-extraction operator $f(\mathrm{X}) = h_{\mathrm{HL}}(\mathrm{A})\,\mathrm{X}$ is shift and rotation-variant. Based on Theorem 27, the corresponding optimal resampling strategy is

$$\pi_i^* \propto \sqrt{c^2\left\|\left(\mathbf{I}+\mathrm{A}_{\mathrm{norm}}\right)_i\right\|_2^2 + \left\|\left(\left(\mathbf{I}+\mathrm{A}_{\mathrm{norm}}\right)\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2}, \tag{11.18}$$

To obtain this optimal resampling distribution, we need to compute the largest magnitude eigenvalue $\lambda_{\max}$, which takes $O(N)$, and compute $\|(\mathbf{I}+\mathrm{A}_{\mathrm{norm}})_i\|_2^2$ and $\|((\mathbf{I}+\mathrm{A}_{\mathrm{norm}})\,\mathrm{X}_{\mathrm{o}})_i\|_2^2$ for each row, which takes $O(\|\mathrm{vec}(\mathrm{A})\|_0)$ with $\|\mathrm{vec}(\mathrm{A})\|_0$ the nonzero elements in the graph shift operator. We can avoid computing the largest magnitude by using a normalized adjacency matrix or a transition matrix as a graph shift operator. A normalized adjacency matrix is $\mathrm{D}^{-\frac{1}{2}}\,\mathrm{W}\,\mathrm{D}^{-\frac{1}{2}}$,

where D is the diagonal degree matrix, and a transition matrix is obtained by normalizing the sum of each row of an adjacency matrix to be one; that is $\mathrm{D}^{-1}\,\mathrm{W}$. In both cases, the largest eigenvalue of a transition matrix is one, we thus have $\mathrm{A} = \mathrm{A}_{\mathrm{norm}}$.

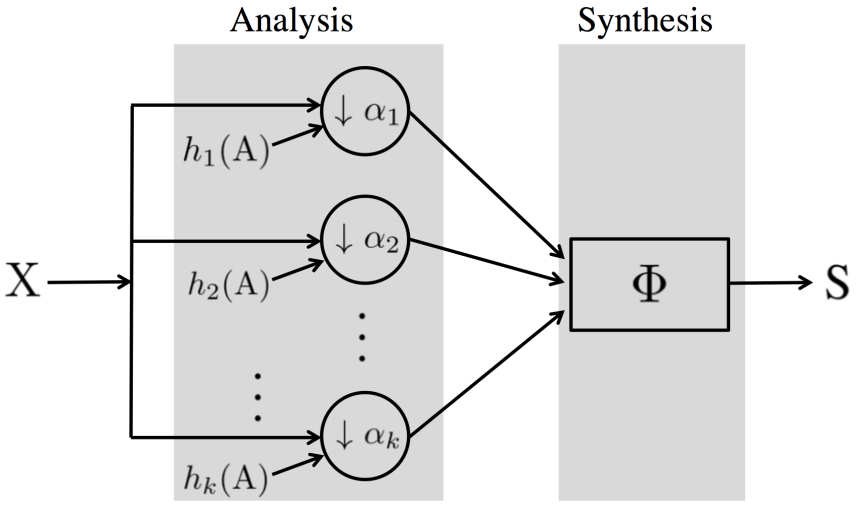


Figure 11.5: Graph filter bank analysis for 3D point clouds. In the analysis part, we separate a 3D point cloud into multiple subbands. In each subband, we resample a subset of 3D points based on a specific graph filter $h(\mathrm{A})$. The number of samples in each subband is determined by a downsampling ratio α. In the synthesis part, we use all the resampled points to reconstruct a surface via a reconstruction operator Φ.

11.3.3 Graph Filter Banks

In classical signal processing, a filter bank is an array of band-pass filters that analyze an input signal in multiple subbands and synthesize the original signal from all the subbands [84, 232]. We use a similar idea to analyze a 3D point cloud: separate an input 3D point cloud into multiple components via different resampling operators, allowing us to enhance different components of a 3D point cloud. For example, we resample both contour points and noncontour points to reconstruct the original surfaces, but we need more contour points to emphasize contours.

Figure 11.5 shows a surface reconstruction system for a 3D point cloud based on graph filter banks. In the analysis part, we separate a 3D point cloud X into k subbands. In each subband, the information preserved is determined by a specific graph filter and we resample a subset of 3D points according to (11.7) and (11.8). The number of samples in each subband is determined by a sampling ratio α. We have flexibility to use either the original 3D points or the 3D points after graph filtering. In the synthesis part, we use the resampled points to reconstruct the surface. A literature review on surface reconstruction algorithms is shown in [233]. Since each surface reconstruction algorithm has its own specific set of assumptions, different surface reconstruction algorithms perform differently on the same set of 3D points.

We measure the overall performance of a surface reconstruction system by reconstruction error, which is the difference between the surface reconstructed from resampled points and the

original surface. This leads to a rate-distortion like tradeoff: when we resample more points, we encode more bits and the reconstruction error is smaller; when we resample fewer points, we encode less bits and the reconstruction error is larger. The overall goal is: given an arbitrary tolerance of reconstruction error, we use as few samples as possible to reconstruct a surface by carefully choosing a graph filter and sampling ratio in each subband. Such a surface reconstruction system will benefit a 3D point cloud storage and compression because we only need to store a few resampled points.

11.4 Experimental Results

In this section, we apply the proposed resampling strategies to three applications: large-scale visualization, accurate registration and robust shape modeling.

11.4.1 Large-scale Visualization

Here we use the proposed resampling strategy (11.12) to efficiently visualize large-scale urban scenes. Since our visual system is sensitive to the contours of buildings and streets in a urban scene, instead of showing an entire point cloud, we only show a selected subset of points, which is useful for large-scale visualization and efficient data summarization. We consider a large-scale dataset [2], which involves several natural scenes with over 3 billion points in total and covers a range of diverse urban scenes: churches, streets, railroad tracks, squares, villages, soccer fields, castles.

Figure 11.6 (a) shows a point cloud of 'domfountain3', which contains $15,105,667$ points (we ignore the points on ground). We compare the resampled point clouds based on two resampling strategies: uniform resampling and high-pass graph filtering based resampling. Figures 11.6 (b), (d) and (f) show the resampled point clouds based on uniform resampling with $151,057$, $15,105$ and $1,510$ (1%, 0.1% and 0.01%) points, respectively. Figures 11.6 (c), (e) and (g) show the resampled point clouds based on high-pass graph filtering based resampling with $151,057$, $15,105$ and $1,510$ (1%, 0.1% and 0.01%) points, respectively. We see that Figures 11.6 (c), (e) and (g) show much clearer contours than Figures 11.6 (b), (d) and (f). This validates that the high-pass graph filtering based resampling strategy provides visual-friendly results for large-scale urban scenes. The entire computation process, including graph construction, local variation computation and resampling, was running on Matlab in a desktop and took less than 200 seconds.

To look into some details, we show two examples in Figure 11.7, including a building and a church, which contain $381,903$ and $1,622,239$ points, respectively. The four columns in Figure 11.7 show the original point cloud, the points that have top 1% largest local variations (11.10) (large local-variation points are shown in red, other points are shown in black), the resampled points based on uniform resampling and the resampled points based on high-pass graph filtering (11.12). In the second column, we successfully detect the outlines of both the building and the church as the contour. Some details, such the gates and windows of the building and the clock

[2] `http://semantic3d.net/view_dbase.php?chl=1`

	RMSE	$\text{Error}_{\text{shift}}$	$\text{Error}_{\text{rotation}}$
All points	4.22	8.76	2.30×10^{-3}
Uniform resampling	4.27	9.38	3.76×10^{-3}
High-pass graph filtering based resampling (11.12)	**1.49**	**0.01**	$\mathbf{4.29 \times 10^{-5}}$

Table 11.1: Proposed high-pass graph filtering based resampling strategy provides an accurate registration for a sofa. Best results are marked in bold. The first column shows RMSE; the second column shows the shift error; The third column shows the rotation error. High-pass graph filtering based resampling chooses key points and provides the best registration performance.

and the roof of the church, are also highlighted. This validates the effectiveness of the proposed local variation. Comparing the fourth column to the third column, the resampled points based on high-pass graph filtering preserves contours. We still recognize the contour of gates and windows of the building from the resampled points.

Note that the current results are compared visually, which may be biased. The difficulties of showing a quantitative result come from the irregular distribution of 3D points and the lack of proper models of visual perception. For example, human eyes may be more sensitive to contours in a 3D point cloud, but a reliable perception model for 3D point clouds is unknown. We leave the quality evaluation metric for 3D point clouds as future work.

11.4.2 Contour-based Registration

In this task, we use the proposed resampling strategy (11.12) to make two point clouds registered efficiently and accurately.

Figure 11.8 (a) shows a point cloud of a sofa, which contains $1,204,055$ points collected from a Kinect based SLAM system [229]. As shown in Figure 11.8 (a), we split the original point cloud into two overlapping point clouds marked in red and blue, respectively. We intentionally shift and rotate the red part. The task is to invert the process and retrieve shift and rotation. We use the iterative closest point (ICP) algorithm to register two point clouds, which is a standard algorithm to rotate and shift different scans into a consistent coordinate frame [234]. The ICP algorithm iteratively revises the rigid body transformation (combination of shift and rotation) needed to minimize the distance from the source to the reference point cloud. Figures 11.8 (b) and (c) show the registered sofa and the details of the overlapping part after registration, respectively. We see that the registration process recovers the overall structure of the original point cloud, but still leaves some mismatch in a detailed level.

Since it is inefficient to register two large-scale point clouds, we want to resample a subset of 3D points from each point cloud and implement registration. We will compare the registration performance between uniformly resampled point cloud and high-pass graph filtering based resampled point cloud. Note that high-pass graph filtering based resampling can enhance the contours and key points. Figures 11.8 (d) and (g) show the resampled point clouds based on uniform resampling and high-pass graph filtering based resampling, respectively. Two resampled versions have the same number of points, which is 5% of points in the original point cloud. We

see that Figures 11.8 (g) shows more contours than Figures 11.8 (d). Based on the uniformly resampled version Figures 11.8 (d), Figures 11.8 (e) and (f) show the registered sofa and the details of the overlapping part after registration, respectively. Based on the contour-enhanced resampled version Figures 11.8 (g), Figures 11.8 (h) and (i) show the registered sofa and the details of the overlapping part after registration, respectively. We see that the registration based on high-pass graph filtering based resampling precisely recovers the original point cloud, even in a detailed level. The intuition is that the high-pass graph filters enhance the contours, which make sharper match between the sources and targets, and thus the registration becomes easier.

The quantitative results are shown in Table 11.1, where the first column shows the root mean square error (RMSE); the second column shows the shift error; The third column shows the rotation error. Specially,

$$\begin{aligned} \mathrm{RMSE} &= \sqrt{\sum_{i=1}^{N} \min_{j=1,\ldots,N} \|\widehat{\mathbf{x}}_i - \mathbf{x}_j\|_2^2}, \\ \mathrm{Error}_{\mathrm{shift}} &= \|\widehat{\mathbf{a}} - \mathbf{a}\|_2 , \\ \mathrm{Error}_{\mathrm{rotation}} &= \left\|\widehat{\mathrm{R}} - \mathrm{R}\right\|_{\mathrm{Frobenius}} , \end{aligned}$$

where $\widehat{\mathbf{x}}_i$, $\widehat{\mathbf{a}}$ and $\widehat{\mathrm{R}}$ are the 3D coordinates of the ith point, recovered shift vector and recovered rotation matrix after registration, respectively; $\mathbf{x}_i$, $\mathbf{a}$ and R are the ground-truth 3D coordinates of the ith point, ground-truth shift vector and ground-truth recovered rotation matrix. We see that high-pass graph filtering based resampled point cloud uses 20-times less points and achieves even better results than using all the points. The shift and rotation errors of using high-pass graph filtering based resampling are significantly smaller than those of using all the points or using uniform resampling.

11.4.3 Robust Shape Modeling

In this task, we use the proposed resampling strategy (11.18) based on Haar-like low-pass graph filter to achieve robust shape modeling. In Section 11.3.2, we discussed the shape modeling for a fitness ball, which is a simple sphere. Here we consider a more sophisticated point cloud and use poisson reconstruction to retrieve the surface. The goal is to show that low-pass graph filtering makes the shape modeling more robust to noise.

As described earlier, Figure 11.9 (a) shows a point cloud of a sofa. Figure 11.9 (b) fits a surface to the point cloud based on Poisson surface reconstruction, a method to reconstruct object surfaces as meshes from point clouds [235].

To test the robustness to noise, we add the Gaussian noise with mean zero and variance 20 to each point. Figures 11.9 (c) and (d) show the noisy point cloud and its reconstructed surface, respectively. Figure 11.9 (e) shows the signed cloud-to-mesh distance from Figures 11.9 (d) to (b), which shows the influence of noise to surface reconstruction. The error of surface reconstruction is measured by a cloud-to-mesh (C2M) distance used in a public software CloudCompare. The C2M distance is defined as follows[3]: for each point in (d), search the nearest triangle in the

[3]More details see `http://www.cloudcompare.org/doc/wiki/index.php?title=`

	Mean distance	Std deviation
Noisy sofa	3.84	4.34
Uniform resampling	1.99	7.55
Low-pass graph filtering based resampling (11.18)	**0.34**	**3.89**

Table 11.2: Proposed low-pass graph filtering based resampling strategy provides a robust surface reconstruction for a sofa. Best results are marked in bold. Note that we resample the same number of 3D points from the original point cloud (noiseless) and the C2M distance is with mean 0.16 and standard deviation 1.65.

mesh (b), and then compute the signed distance from the point to that triangle using the triangle's normal.

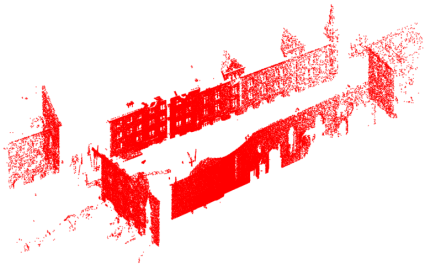
Figures 11.9 (f), (g) and (h) show the resampled version based on uniform resampling, its reconstructed surface and the C2M distance from (g) to (b), respectively. Figures 11.9 (i), (j) and (k) show the resampled version based on low-pass graph filtering based resampling, its reconstructed surface and the C2M distance from (j) to (b), respectively. The comparison of reconstruction errors are shown in Table 11.2. Uniform resampling resamples 3D points directly from the noisy point cloud, whose corresponding reconstructed surface is not robust: both the mean and the variance of the C2M distance are large. On the other hand, low-pass graph filtering based resampling denoises 3D points while resampling and produces a more robust surface. This validates that the proposed resampling strategy with low-pass graph filtering provides a robust shape modeling for noisy point clouds.

11.5 Conclusions

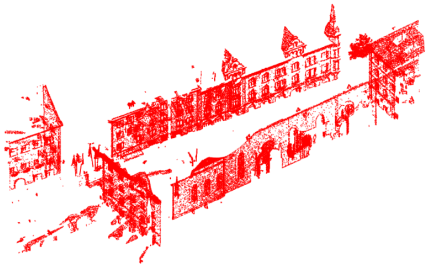
We proposed a resampling framework to select a subset of points to extract application-dependent features and reduce the subsequent computation in a large-scale point cloud. We formulated an optimization problem to obtain the optimal resampling distribution, which is also guaranteed to be shift/rotation/scale invariant. We then specified the feature extraction operator to be a graph filter and studied the resampling strategies based on all-pass, low-pass and high-pass graph filtering. A surface reconstruction system based on graph filter banks was introduced to compress 3D point clouds. Three applications, including large-scale visualization, accurate registration and robust shape modeling, were presented to validate the effectiveness and efficiency of the proposed resampling methods.

`Distances_Computation`.

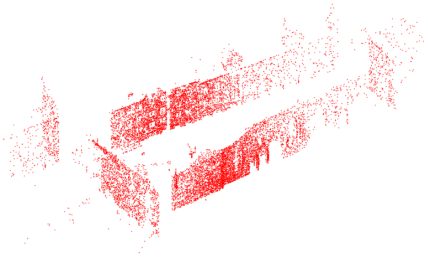
(a) Original.



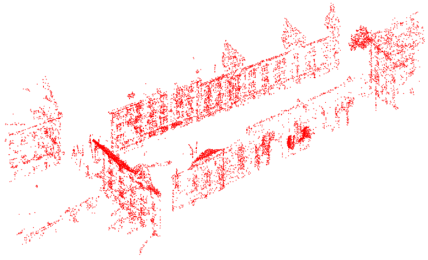
(b) Uniform resampling ($\downarrow 100$).



(c) High-pass graph filtering based resampling ($\downarrow 100$).

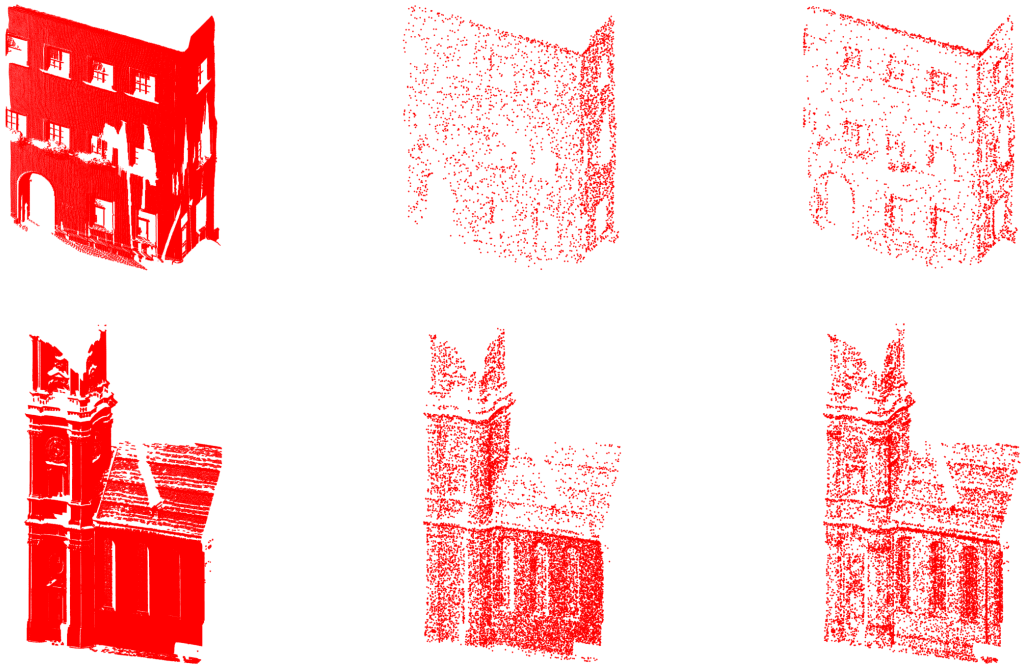


(d) Uniform resampling ($\downarrow 1,000$).



(e) High-pass graph filtering based resampling ($\downarrow 1,000$).

Figure 11.6: Proposed resampling strategy (11.12) helps efficiently visualize a large-scale urban scene. Plot (a) shows a point cloud of Station3 of domfountain. Plots (b), (d) and (f) show the resampled point clouds based on uniform resampling with $151,057$ and $15,105$ (1% and 0.1%) points. Plots (b), (d) and (f) show the resampled point clouds based on high-pass graph filtering based resampling with $151,057$ and $15,105$ (1% and 0.1%) points. High-pass graph filtering based resampling provides clear contours.

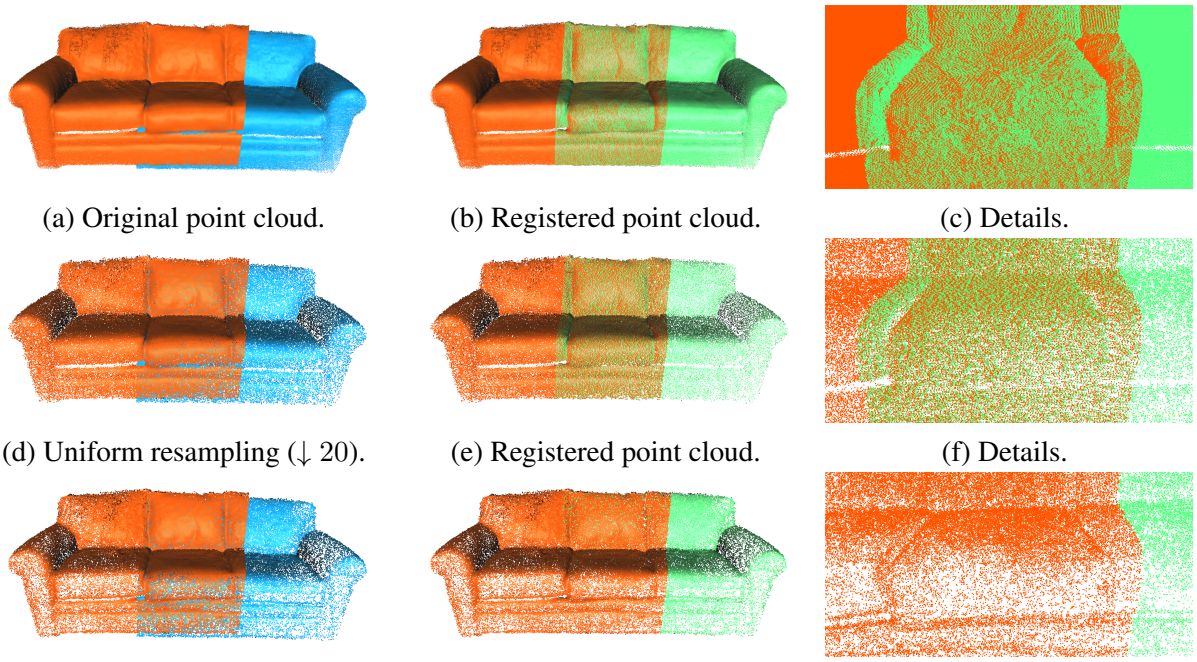


Original.

Uniform resampling after hard thresholding.

Resampling based on high-pass graph filtering.

Figure 11.7: High-pass graph filtering helps detect contours and preserve the contour information during resampling. The four columns show the original point cloud, the points that have top 1% largest local variations (11.10) (large local-variation points are shown in red, other points are shown in black), resampled points based on uniform resampling and resampled points based on the Haar-like high-pass graph filtering (11.12). Two resampled versions have the same number of points, which is 1% of points in the original point cloud.



(a) Original point cloud. (b) Registered point cloud. (c) Details.

(d) Uniform resampling (↓ 20). (e) Registered point cloud. (f) Details.

(g) High-pass graph filtering based graph filtering (↓ 20). (h) Registered point cloud. (i) Details.

Figure 11.8: Accurate registration for sofa. The first row shows the original point cloud; the second row shows the uniformly resampled point cloud; and the third row shows the high-pass graph filtering based resampled point cloud (11.12). The first column shows the point clouds before registration; the second column shows the point clouds after registration; and the second column shows the registration details. Comparing to the other two methods, high-pass graph filtering-based resampling provides more precise registration by using less points.

(a) Original point cloud. (b) Reconstruction.

(c) Noisy point cloud. (d) Reconstruction. (e) C2M distance.

(f) Uniform resampling (↓ 20). (g) Reconstruction. (h) C2M distance.

(i) Resampling based on low-pass graph filtering (↓ 20). (j) Reconstruction. (k) C2M distance.

Figure 11.9: Robust surface reconstruction for sofa. The first row shows the original point cloud; the second row shows the noisy point cloud; the third row shows the result of uniform resampling from the noisy point cloud; the fourth row shows the result of low-pass graph filtering-based resampling (11.18) from the noisy point cloud. Plots (e), (h) and (k) show the relative errors to the original surface, where the error bar is aligned and shown on the right corner. We see that low-pass graph filtering-based resampling provides the smallest error.

Chapter 12

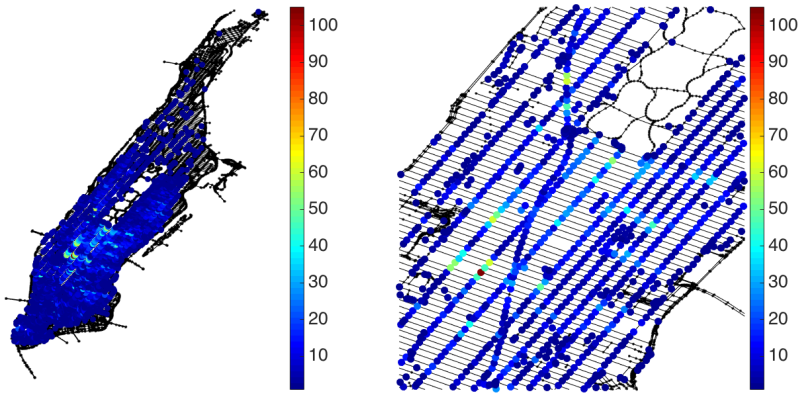
Urban Data Mining

12.1 Introduction

Urban data records the behavior of urban ecosystem and analyzing those urban data potentially leads to improvements of the urban lives [236, 237]. As one of the most critical components of urban data, traffic data is a key to understand the mobility pattern and make cities more efficient; however, traffic data is usually sparse because a few sensors are installed to cover a limited number of intersections [238]. In this paper, we aim to recover entire traffic data in the entire city from sensors installed at a few intersections. To verify the feasibility of this idea, we focus on Manhattan's taxi pickups during the years of 2014 and 2015 because taxis are valuable sensors of city life [239]. We model taxi-pick activities as graph signals supported on a city street network where signal coefficient at a node reflects the number of taxi pickups at the corresponding intersection. The recovery of taxi-pick activities is nothing but graph signal sampling and recovery [90, 98, 105]; however, previous works only consider sampling and recovery of smooth graph signals. Real taxi-pick activities may not be smooth on a city street network; see Figure 12.1. To handle this problem, we propose a series novel techniques based on graph signal processing [1, 2] to learn traffic patterns from historical taxi-pick activities and design targeted sampling and recovery strategies. We are able to approximately recover the taxi-pick activities in entire Manhattan by taking samples at only 5 selected intersections. Here we focus on taxi-pick activity, but the same techniques can be applied to recover many other types of traffic data.

12.2 Problem Formulation

We consider a city street network $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is the set of nodes (intersections), $\mathcal{E} = \{e_1, \ldots, e_M\}$ is the set of edges (streets). A *graph signal* models taxi pickups in a city that assigns the number of taxi pickups during a period of time $x_n \in \mathbb{R}$ to the node v_n; a vector form is $\mathbf{x} = \begin{bmatrix} x_1, x_2, \ldots, x_N \end{bmatrix}^T \in \mathbb{R}^N$. Let $C \subseteq \mathcal{V}$ be a set of nodes (an area in a city). We can represent this set by using an indicator function, $\mathbf{1}_C \in \mathbb{R}^N$, where $(\mathbf{1}_C)_i = 1$ when $v_i \in C$, and 0 otherwise. The signal coefficients are ones in the node set C and zeros in the complement node set $\bar{C} = \mathcal{V}/C$. When the node set C forms a connected subgraph, we call C a *piece* and $\mathbf{1}_C$ a *one-piece graph signal*. A piecewise-constant graph signal is a linear



(a) Global view. (b) Zoom-in plot.

Figure 12.1: Taxi-pickup distribution at 6 pm on January 1st, 2015. The data is not smooth on the Manhattan street network. We approximate this data by a piecewise-constant graph signal.

combination of several one-piece graph signals $\mathbf{x} = \sum_{i=1}^{K} \mu_i \mathbf{1}_{C_i}$, where each C_i is a piece, μ_i is a constant and K is the number of pieces.

Sampling & Recovery. We consider sampling the number of passing vehicles at several selected intersections and then recovering the number of passing vehicles at the rest intersections. Mathematically, we sample M coefficients at selected indices (intersections) in a graph signal $\mathbf{x} \in \mathbb{R}^N$ to produce a sampled signal $\mathbf{y} = \Psi\mathbf{x}$, where the sampling operator Ψ is a linear mapping from $\mathbb{R}^N$ to $\mathbb{R}^M$ with $\Psi_{i,j} = 1$ when we sample the jth node in the ith measurement, and 0, otherwise. Here we consider experimentally design sampling, which allows that sample indices are chosen beforehand. We then interpolate $\mathbf{y}$ to get a recovery $\widehat{\mathbf{x}} = \Phi\mathbf{y} \in \mathbb{R}^N$, where Φ is the interpolation operator designed based on Ψ.

12.3 Methodology

The proposed method involves two phases: learning phase and real-time processing phase. In the learning phase, we learn all the operators needed in the real-time processing phase from historical taxi-pickup activities. In the real-time processing phase, we sample the taxi-pickup activities at a few selected intersections and recover the rest by using the operators learned in the learning phase.

12.3.1 Learning Phase

The purpose of learning phase is to learn important patterns from historical traffic data and then decide which intersections we need to sample. A basic idea is to construct a graph that promotes smoothness for historical taxi-pickup activities and then use graph sampling techniques to design samples [70]. Does the original Manhattan street network promote smoothness for traffic data?

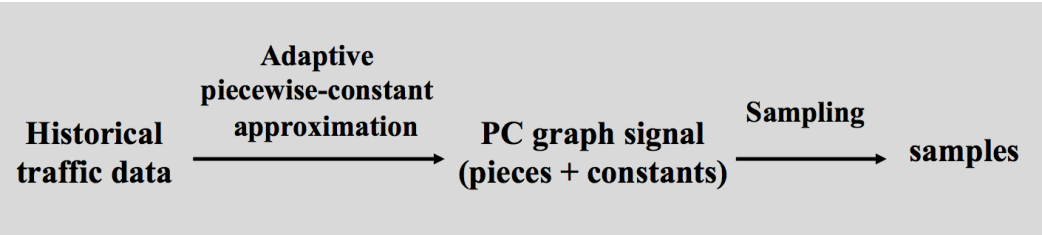
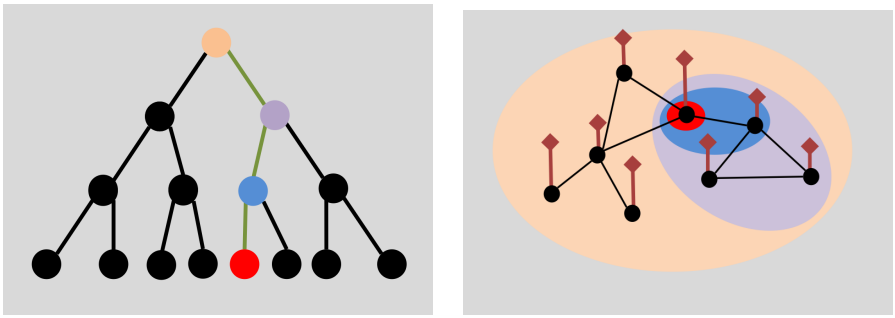


Figure 12.2: Learning phase includes two main blocks: adaptive piecewise-constant approximation implemented by adaptively pruning a decomposition tree and sampling implemented by sampling bandlimited graph signals. In the learning phase, we decide which node to sample.

Figure 12.1 shows the taxi-pickup distribution at 7 pm on January 1st, 2015. We see that many intersections have many more taxi pickups than their neighbors and the entire distribution is barely smooth. We thus need to learn a graph from traffic data. However, a city street network is usually huge and historical traffic data is limited. For example, Manhattan has $13,670$ intersections. It is thus inefficient and unrobust to construct a huge graph. To overcome this, we should reduce the size of graph. In real traffic data, we find that sometimes neighboring intersections have similar number of taxi pickups. We can significantly reduce the size of graph by exploring local information and grouping those neighboring intersections as one super-node. This is equivalent to approximate the original graph signal by using a piecewise-constant graph signal. Through approximation, the dimension reduces from the number of intersections to the number of pieces. We then construct a super-graph whose nodes are pieces and edges are the similarities between pieces. We then can use graph sampling to design samples. Figure 12.2 overviews the procedure of the learning phase. The two main modules are adaptive piecewise-constant approximation and graph sampling. We now elaborate these two blocks.

Adaptive piecewise-constant approximation. The goal is to adaptively find a piecewise-constant graph signal to approximate taxi-pickup activities. The key of approximation is to design a series of nonoverlapping pieces that captures the variation of an graph signal. There are usually two approaches to design such a series: predesigned approach and learning approach. In a predesigned approach, we design pieces before accessing any traffic data. We can simply use physical partitions, such as zipcodes and census blocks, but these partitions may not be flexible enough to capture complex variations in traffic data; on the other hand, in a learning approach, we learn a series of pieces to fit traffic data. However, there are multiple restrictions in the optimization: those pieces are connected, nonoverlapping and cover the vertex domain. It is thus inefficient and unrobust to solve a nonconvex optimization problem with multiple constraints and limited training data.

Here we consider combining the advantages of these two approaches. We first design a set of redundant pieces before having any data. Because of the redundancy, this set is able to capture various shapes and sizes. We then prune this set and selects the best series of nonoverlapping pieces according to historical taxi-pickup activities. This approach is both adaptive and efficient. The set of redundant pieces can be constructed beforehand and the bottleneck of the computational complexity is the pruning stage. By taking advantage of a tree-structure, the computational complexity is merely $O(N)$.



(a) Global view. (b) Zoom-in plot.

Figure 12.3: Grow a binary tree in (a) is equivalent to decompose a graph in (b). The green path in (a) is a decomposition in (b), where the same color indicates the one-to-one mapping from a node in the decomposition tree to a piece in a graph.

A set of redundant pieces can be constructed via a binary tree decomposition as shown in [93]. This set is provably useful to represent arbitrary piecewise-constant graph signals. The main idea is to recursively partition a piece into two disjoint pieces until that all the pieces are individual nodes. Figure 12.3 shows an example. A node in (a) represents a piece in (b) and an edge represents a kinship where a parent node partitions into two children nodes. The top node (in orange) represents the entire vertex domain and the bottom nodes represents all the individual nodes. The green path in (a) is a decomposition in (b), where the same color indicates the one-to-one mapping from a node in a decomposition tree to a piece in a graph. We use the 2-means clustering to implement graph partitioning [93]. For each piece, we select two nodes with longest geodesic distance as the community centers and assign all the other nodes to their nearest community center based on the geodesic distance. We then recompute the community center for each community by minimizing the summation of the geodesic distances to all the other nodes in the community and assign node to its nearest community center again. We keep doing this until the community centers converge after a few iterations. Please find more details in [93].

By using the binary tree decomposition, we obtain $(2N-1)$ pieces. which is redundant and captures various sizes and shapes of pieces. We then prune this set and selects the best series of nonoverlapping pieces according to historical traffic data. Let $\mathcal{C} = \{\mathbf{1}_{C_i}\}_{i=1}^{2N-1}$ be the set of constructed pieces. We aim to select a subset of pieces that minimizes the following optimization problem,

$$\begin{aligned} \widehat{\mathrm{D}}, \widehat{\mathrm{Z}} \;\; = \;\; & \arg\min_{\mathrm{D}_i \in \mathcal{C}, \mathrm{Z}} \|\mathrm{X} - \mathrm{D}\,\mathrm{Z}\|_F^2 + \lambda \dim(\mathrm{Z}), \\ & \text{subject to } \mathrm{D}\,\mathbf{1} \;=\; \mathbf{1}, \end{aligned} \tag{12.1}$$

where $\mathrm{X} \in \mathbb{R}^{N \times L}$ is a matrix representation of historical taxi-pick activities with L snapshots, $\mathrm{D} \in \mathbb{R}^{N \times K}$ is a matrix representation of constructed pieces with D_i being the ith column, λ is a tuning parameter and $\mathrm{Z} \in \mathbb{R}^{K \times L}$ stores the constants of all the pieces with $\dim(\mathrm{Z})$ the number of elements in Z. Note that K is variable during the optimization because we do not know how many pieces we need in advance.

The first term in the objective functions pushes the piecewise-constant approximation to fit

the given data. The second term punishes a large size of the constant matrix Z and avoids overfitting; that is, when λ is large, we tend to select fewer pieces from $\mathcal{C}$ to fit data and when λ is small, we tend to select all the pieces in $\mathcal{C}$ to fit data. The constraint requires that all the selected pieces are nonoverlapping and covers the entire vertex domain. Since each column in D is a one-piece graph signal, the optimization problem (12.1) finds the best piecewise-constant approximation for given traffic data. Since the constructed pieces in $\mathcal{C}$ have a nice tree structure, we can easily obtain the global optimum of (12.1) by pruning the tree, which follows the paradigm in [240, 241]. The main idea is to compare the representation based on a parent piece to the representation based on its two children pieces and see which representation minimizes the objective function (12.1). For example, C_1 is a parent piece and C_2, C_3 are its children pieces. Since the parent piece and two children pieces represent the same vertex domain ($C_1 = C_2 \cup C_3$), to satisfy the constraint, we either choose the parent piece or its two children pieces. The cost of using the parent piece is

$$\min_{\mathbf{z}} \left\| \mathrm{X} - \mathbf{1}_{C_1} \mathbf{z}^T \right\|_F^2 + \lambda L,$$

with optimum $\left\| \mathrm{X} - \mathbf{1}_{C_1} \mathbf{1}_{C_1}^T \mathrm{X} \right\|_F^2 + \lambda L$, and the cost of using the child pieces is

$$\min_{\mathrm{Z} \in \mathbb{R}^{2 \times L}} \left\| \mathrm{X} - \begin{bmatrix} \mathbf{1}_{C_2} & \mathbf{1}_{C_3} \end{bmatrix} \mathrm{Z} \right\|_F^2 + 2\lambda L,$$

with optimum $\left\| \mathrm{X} - \begin{bmatrix} \mathbf{1}_{C_2} & \mathbf{1}_{C_3} \end{bmatrix} \begin{bmatrix} \mathbf{1}_{C_2} & \mathbf{1}_{C_3} \end{bmatrix}^T \mathrm{X} \right\|_F^2 + 2\lambda L$ Each time, we compare their costs and choose the one with a smaller cost to update the representation at the parent piece. The pruning process starts from the bottom level of the decomposition tree and move to an upper level iteratively until we reach the top level. Through the pruning process, we obtain the global optimum of (12.1).

Sampling. We next model each selected piece after pruning as a super-node and construct a super-graph. Since the selected pieces already capture the local similarities, the connection among super-nodes are not relevant to the geodesic distance any more. We need to learn a super-graph to promote smoothness for historical taxi-pickup activities and then design which super-nodes to sample. In graph sampling, we usually model a smooth graph signal as a bandlimited graph signal [62, 90, 228] whose sampling strategy is designed based on the corresponding graph Fourier basis. Thus, instead of constructing a full super-graph, we directly construct a graph Fourier basis. Recall that the bandlimited assumption requires that most energy of a graph signal is concentrated in the low-pass band; that is, we need to find a graph Fourier basis that pushes the energy to the subspace spanned by its first few basis vectors. Thus, all we need is the first few columns in the graph Fourier basis, which can be simply obtained by principal component analysis. Principal component analysis uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables [117], which exactly fits our requirement. Mathematically, let the constant matrix $\widehat{\mathrm{Z}} \in \mathbb{R}^{K \times L}$ be a matrix of graph signals on the super-graph, we obtain the first M graph Fourier basis vectors (principal components) by solving the following optimization problem,

$$\widehat{\mathrm{V}} = \arg \min_{\mathrm{V} \in \mathbb{R}^{K \times M}} \left\| \widehat{\mathrm{Z}} - \mathrm{V}\, \mathrm{V}^T\, \widehat{\mathrm{Z}} \right\|_F^2, \text{ subject to } \mathrm{V}^T\, \mathrm{V} = \mathbf{I}\,.$$

It is true that we can obtain a truncated graph Fourier basis directly from X, which is equivalent to set λ be zero in (12.1); however, the computation is less efficient and the obtained principal components are learned from noisy and limited historical data and do not take advantage of the local grouping[1], which is explored by (12.1). Next, we design a sampling operator by using graph sampling techniques. For example, we solve $\widehat{\Psi} = \arg\max_{\Psi} \sigma_{\min}(\Psi\widehat{\mathrm{V}}) \in \mathbb{R}^{M\times K}$ by using a greedy method, which is shown in [90]. Note that we sample super-nodes, instead of individual intersections. To directly operate on individual intersections, the sampling operator is $\widehat{\Psi}\widehat{\mathrm{D}}^T \in \mathbb{R}^{M\times N}$, which means that $\widehat{\Psi}$ selects some pieces from $\widehat{\mathrm{D}}$ in (12.1). We then need to sample all the nodes in the selected pieces, or sample several nodes in selected pieces and estimate the average values. In the experiments, we find that the selected pieces happen to be single nodes; that is, we only need to sample one intersection for a piece.

12.3.2 Real-time Processing Phase

In the learning phase, we obtain three operators from historical taxi-pickup activities: selected pieces $\widehat{\mathrm{D}}$, truncated graph Fourier basis $\widehat{\mathrm{V}}$, sampling operator $\widehat{\Psi}$. Given a real-time traffic data $\mathbf{x} \in \mathbb{R}^N$, we first take samples at the selected intersections, $\mathbf{y} = \widehat{\Psi}\widehat{\mathrm{D}}^T\mathbf{x}$. Then we use the interpolation operator to recover all the constants, $\mathbf{z} = \widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\mathbf{y}$. Finally, we obtain a piecewise-constant approximation to the real taxi pickups by

$$\widehat{\mathbf{x}} = \widehat{\mathrm{D}}\mathbf{z} = \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\mathbf{y} = \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}\widehat{\Psi}\widehat{\mathrm{D}}^T\mathbf{x},$$

where the interpolation operator is $\Phi = \widehat{\mathrm{D}}\widehat{\mathrm{V}}(\widehat{\Psi}\widehat{\mathrm{V}})^{\dagger}$. Figure 12.4 illustrates the procedure in real-time processing.

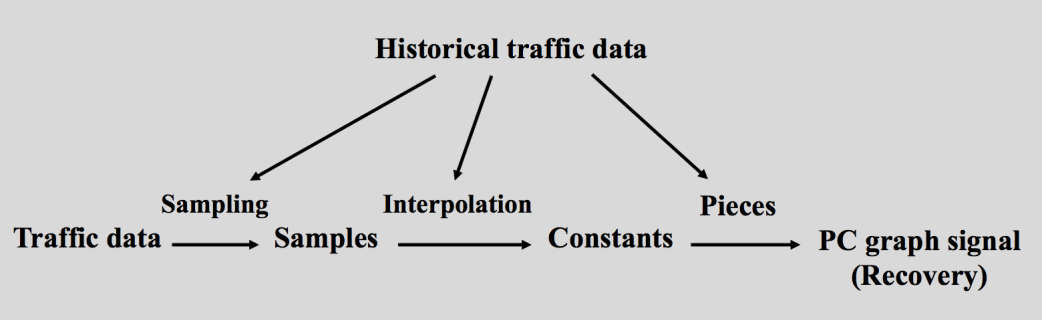


Figure 12.4: In real-time processing, we sample the selected nodes, recover all the constants, and finally obtain the piecewise-constant estimation to the real-time traffic data.

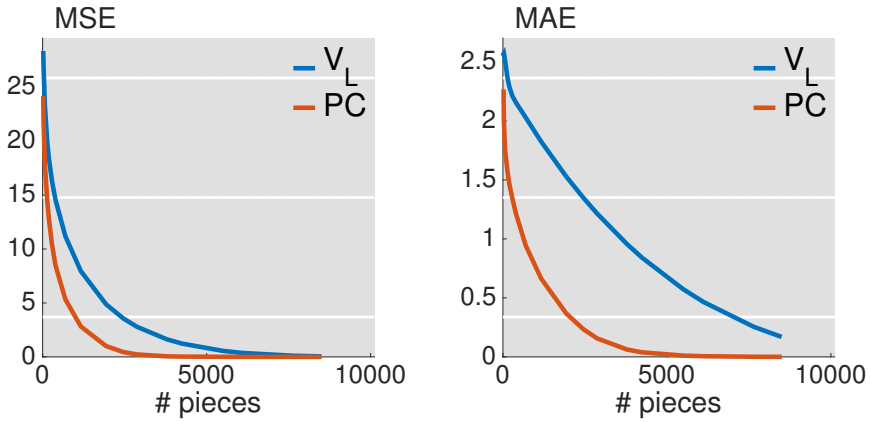
[1]Piecewise-constant approximation can be regarded as a denoising block. Many experiments indicate that reducing the dimension to $N/2$ provides the best recovery performance in the end, which is better and faster than directly working with X.

12.4 Experimental Results

We validate the proposed method on a dataset of Manhattan's taxi pickups. We sample the number of taxi pickups at several intersections and recover the taxi pickups at the rest intersections. This is similar to count the number of taxi pickups at several intersections and recover the number of taxi pickups at all the other intersections.

Dataset. We consider taxi pickups in Manhattan[2]. Here we use the dataset in the year of 2014 and 2015. We focus on rush hours during workdays (6 pm from Monday to Friday). We accumulate the taxi-pickup activities within a hour and project each taxi pickup to its closest intersection, obtaining 261 graph signals in the year of 2014 for learning and 261 graph signals in the year of 2015 for real-time processing.

Results. We first validate the proposed adaptive piecewise-constant approximation. We solve (12.1) based on 261 graph signals in 2014 by varying the regularization parameter λ. Two metrics are used to quantify the performance, including mean square error (MSE = $\frac{1}{261N}\sum_{i=1}^{261}\|\widehat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$) and mean absolute error (MAE = $\frac{1}{261N}\sum_{i=1}^{261}\|\widehat{\mathbf{x}}_i - \mathbf{x}_i\|_1$), where $\widehat{\mathbf{x}}_i$ is the recovered taxi pickups in the ith day and $\mathbf{x}_i$ is the real taxi pickups in the ith day. Figure 12.5 compares the approximation errors between the graph Fourier basis based on the Laplacian matrix (V_L, in blue) and adaptive piecewise-constant approximation (PC, in red). We see that PC significantly outperforms V_L in terms of both metrics. We then set $\lambda = 1$ (corresponds to 3788 pieces) and obtain 5 samples provided by the optimal sampling operator, as shown in Figure 12.6. As discussed before, these 5 pieces happen to be individual nodes. Two adjacency intersections around Penn Station are sampled, indicating that Penn Station is the weathercock of Manhattan's traffic.

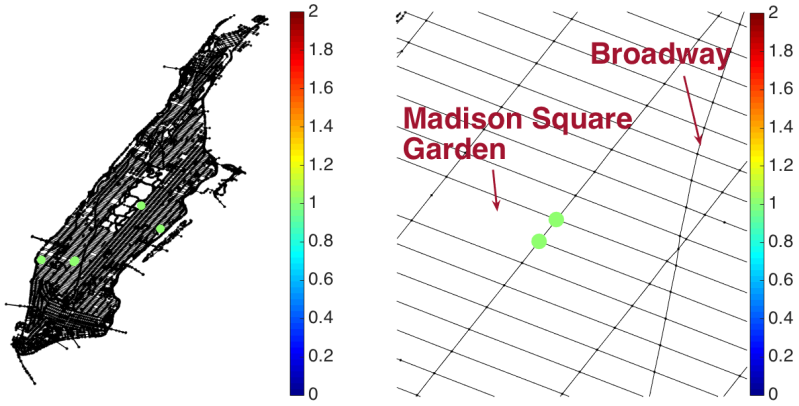


(a) MSE. (b) MAE.

Figure 12.5: Piecewise-constant approximation significantly outperforms smooth approximation.

We next validate those learned operators to the graph signals in 2015. Figure 12.7 shows the recovery of taxi-pick activity at 6 pm, Jan. 6th, 2015 by only using 5 samples. Even we just use 5 samples, the recovered taxi-pick distribution is very close to the real taxi-pick distribution.

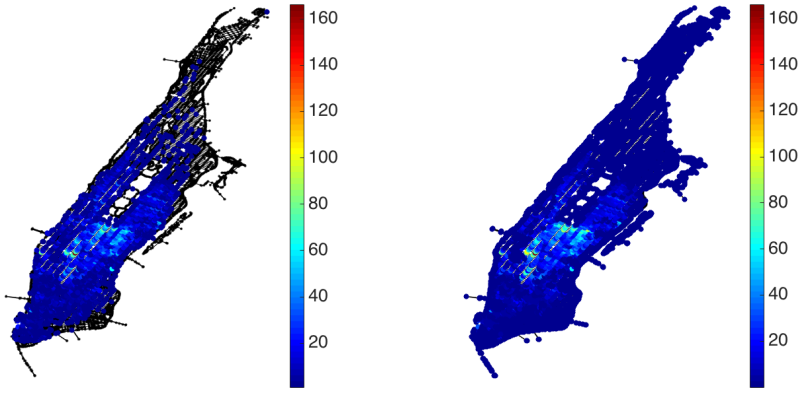
[2]Data is downloaded from `http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml`



(a) Global view. (b) Zoom-in plot.

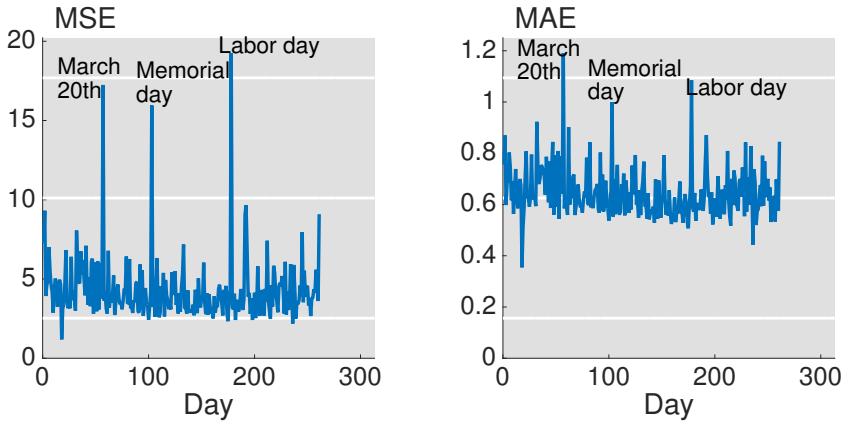
Figure 12.6: Selected 5 intersections. Two adjacency intersections around Penn Station are sampled.

Finally, we show the daily recovery errors in Figure 12.8. The recovery errors are particularly large at Memorial day and Labor day (not surprise), but in general, the recovery error is small. For example, Figure 12.8 (b) shows that the average error at each intersection is merely 0.6 taxi pickups during the rush hour every weekday.



(a) Real taxi-pick distribution. (b) Recovered taxi-pick distribution.

Figure 12.7: Recovered taxi picks at 6 pm, Jan. 6th, 2015.



(a) MSE. (b) MAE.

Figure 12.8: Daily recovery error in the year of 2015.

12.5 Conclusions

We set a goal to monitor Manhattan's traffic from a few intersections. Finally, we are able to obtain an approximate recovery of the taxi-pick activities by taking samples at only 5 selected intersections. The main techniques involves adaptive pieceiwise-constant approximation via decomposition tree pruning, super-graph Fourier basis construction via principal component analysis and sampling for bandlimited graph signals. The paper suggests that graph signal processing tools may aid in urban computing.

Part VI

Concluding Remarks and Further Work

Chapter 13

Concluding Remarks

13.1 Concluding remarks

The main goal of this thesis is to create a framework to analyze data supported on graphs from a signal processing perspective. To this end, we specifically consider five graph-related tasks, including representation, sampling, recovery, detection and localization. We now summarize the contributions.

Representations of Graph Signals. We explicitly define the graph signals of interest and propose a graph dictionary to represent such signals. The graph dictionary unifies the representations based on both the graph Fourier domain and the graph vertex domain and is powerful to make sense of real big data problems. We specifically study three typical classes of graph signals: smooth graph signals, piecewise-constant graph signals, and piecewise-smooth graph signals. To understand smooth graph signals, we introduce four smoothness criteria and propose four graph dictionaries to generate graph signals that satisfy these four smoothness criteria. We then study the graph Fourier basis and observe a localization phenomenon, which shows that the graph Fourier basis of many real networks capture both local and global behaviors on graphs. To represent piecewise-constant graph signals, we propose a local-set based piecewise-constant graph wavelet basis and graph dictionary. We show that the proposed graph dictionary can promote the sparsity for arbitrary piecewise-constant graph signals. We validate the proposed graph dictionary on both the Minnesota traffic network and the U. S sensor network. We then extend this result to piecewise-smooth graph signals and propose local-set based piecewise-smooth graph dictionary. We show that the proposed graph dictionary can promote the sparsity for a large set of piecewise-smooth graph signals and identify piecewise-smooth graphs signals from noises.

Sampling of Graph Signals. We consider sampling from two aspects: finite number of samples and infinite number of samples. For the first case, we propose a sampling theory for graph signals. The theory follows the same paradigm as classical sampling theory. We show that perfect recovery is possible for graph signals bandlimited under the graph Fourier transform. The sampled signal coefficients form a new graph signal, whose corresponding graph structure preserves the first-order difference of the original graph signal. For general graphs, an optimal sampling operator based on experimentally designed sampling is proposed to guarantee perfect

recovery and robustness to noise; for graphs whose graph Fourier transforms are frames with maximal robustness to erasures as well as for Erdős-Rényi graphs, random sampling leads to perfect recovery with high probability. We further establish the connection to the sampling theory of finite discrete-time signal processing and previous work on signal recovery on graphs. To handle full-band graph signals, we propose a graph filter bank based on sampling theory on graphs.

For the second case, we build a theoretical foundation to understand the fundamental limits of three sampling strategies: uniform sampling, experimentally designed sampling and active sampling. We derive the lower bounds on the maximum risk for the approximately bandlimited class under these three sampling strategies and show that active sampling cannot fundamentally outperform experimentally designed sampling. We propose a recovery strategy to compare uniform sampling with experimentally designed sampling. As the proposed recovery strategy lends itself well to statistical analysis, we derive the exact mean square error for each sampling strategy. To study convergence rates, we introduce two types of graphs and find that (1) the proposed recovery strategy achieves the optimal rates; and (2) the experimentally designed sampling fundamentally outperforms uniform sampling for Type-2 class of graphs. To validate our proposed recovery strategy, we test it on five specific graphs: a ring graph with k nearest neighbors, an Erdős-Rényi graph, a random geometric graph, a small-world graph and a power-law graph and find that experimental results match the proposed theory well. This work also presents a comprehensive explanation for when and why sampling for semi-supervised learning with graphs works.

Recovery of Graph Signals. To recover one or multiple smooth graph signals from noisy, corrupted, or incomplete measurements, we formulate graph signal recovery as an optimization problem, for which we provide a general solution through the alternating direction methods of multipliers. We show its relations to signal inpainting, matrix completion, robust principal component analysis and anomaly detection. We propose new algorithms and theoretical analysis for graph signal inpainting, graph signal matrix completion, and anomaly detection of graph signals, all applicable to semi-supervised classification, regression, and matrix completion. We validate the proposed methods on real-world recovery problems, including online blog classification, bridge condition identification, temperature estimation, recommender system for jokes, and expert opinion combination of online blog classification.

Detection of Graph Signals. We formulate a statistical hypothesis testing to decide whether the given attribute activates a community in a graph interfered by Bernoulli noise. We propose two statistics: graph wavelet statistic and graph scan statistic. Both are shown to be efficient and statistically effective to detect activations. The intuition behind the proposed statistics is that we study the interaction between graph structure and the given attribute, that is, we denoise the attribute based on the graph structure and localize the underlying community in the graph. We then test the proposed hypothesis tests on simulated data to validate the effectiveness and robustness of the proposed methods.

Localization of Graph Signals. We use piecewise-constant graph signals to model localized patterns, where each piece indicates a localized pattern that exhibits homogeneous internal behavior and the number of pieces indicates the number of localized patterns. To separate localized each localized pattern, we propose a specific graph signal model, an optimization problem

and a computationally efficient solver. The advantage of the proposed method is parameter-free, scalable and robust to noise. The proposed solver can directly find the the supports of arbitrary localized activated patterns without tuning any threshold, which is a notorious issue in many localization problems. We then conduct an extensive empirical study to validate the proposed methods on both simulated and real data based on the Manhattan street network. The results validate the effectiveness of our approach.

13.2 Future work

Filling in the blanks. Table 2.1 in Part II shows the work that has been explored in this thesis. There are still several holes to fill. For example, some previous works consider sampling discrete-time piecewise bandlimited signals [242, 243]. Is it possible to design a sampling strategy for piecewise-smooth graph signals? This will correspond to the the 2nd row, 3rd column in Table 2.1.

A fundamental understanding of the graph Fourier transform. As the bridge connecting the graph vertex and graph Fourier domains, the graph Fourier transform is the most important tool in graph signal processing. The graph spectrum provides an efficient tool to summarize the variations of a graph signal. Although spectral graph theory provides some information, a deep mathematical understanding of the graph Fourier transform is still needed. For example, what is the difference between the graph Fourier transforms based on graph adjacency matrix and graph Laplacian matrix? how does edge weight influence the graph Fourier transform? how does the local structure influence the graph Fourier transform? What is the mathematical reason of the localization phenomenon?

Another problematic issue is that different from the spectrum in classical signal processing, the graph spectrum does not have a concrete physical meaning. The graph Fourier transform is merely obtained from a mathematical generalization. Is there any other way to generalize the Fourier transform? If the answer is positive, which one we should adopt?

Signal representations based on network motifs. Due to irregular and complex connections, the representation in the graph vertex domain is usually much harder than the representation in the graph Fourier domain; however, the representation in the graph vertex domain is more straightforward and more meaningful, I believe. A subjective reason is that when dealing with graphs, we should understand the irregular connections in the graph vertex domain, instead of avoiding them through the graph Fourier transform. Those irregular connections make graphs unique and interesting. Another reason is that many useful local information may not be reflected in the graph Fourier transform. Network motifs are elementary subgraphs that repeat themselves in a complex network, which may reflect local, functional properties. Network motifs are important tools to analyze the functional abilities in biological networks, social networks, and many others. The motif of time-series is nothing but shift (delay), which is the most elementary building block in classical signal processing. A network-motif based representation may be more useful to understand data generated from biological and social networks than the graph Fourier transform.

Heterophily in graph signal processing. Most works in graph signal processing con-

sider smoothness, which reflects homophily. How about heterophily? As the mirror opposite of homophily, heterophily is defined as the degree to which pairs of individuals who interact are different in certain attributes. For example, a coach talks to players more than another coach. If homophily reflects small variations/low-frequency components, does heterophily high variations/high-frequency components?

Active sampling on graphs. In Chapter 7, we show that active sampling cannot fundamentally outperform experimentally designed sampling for approximately bandlimited graph signals. The next step is to show when active sampling can fundamentally outperform experimentally designed sampling.

General receipt of graph signal localization. The core idea behind sampling and recovery is that information we access is scarce and limited. In a big data era, it is usually easy to access a massive amount of data, but it is hard to efficiently discover patterns in a massive amount of data, which is precisely detection and localization. In data mining, people propose specific detection and localization methods for each pattern. Is it possible to propose a general solution from a signal processing perspective? Maybe a invariant graph translation operator will be useful in this task.

Signal processing on product graphs. The proposed methods are designed for a static graph. How about their counterpart on a time-evolving graph, a knowledge graph, or a multilayer graph? Those graphs can be modeled as product graphs, which can be stored as tensors. Is it possible to generalize the existing tools to product graphs?

Interdisciplinary thinking. Currently, a pattern of most graph signal processing papers is to generalize concepts or tools from classical signal processing to graph signal processing; however, some papers blindly pursue mathematical elegance and are not well motivated from the practical problems. When we look at a bigger picture, what we are really doing is data science with graphs, which overlaps with people from various fields, such machine learning, data mining, network science and mathematics. We should think interdisciplinarily. For example, [244] considers a generalization of convolutional neural networks from low-dimensional regular Euclidean domains, where images (2D), videos (3D) and audios (1D) are represented, to high-dimensional irregular domains such as social networks or biological networks represented by graphs via graph filtering. Over these years, people are always wondering what are the killer applications of graph signal processing. I personally do not believe there is a killer application specifically waiting for graph signal processing. As Prof. Moura mentioned, signal processing itself is really a stealth technology[1]. Some ideas or tools rooted in graph signal processing may be useful to build a module in a large data-analysis system. For example, graph filter bank and graph signal sampling could be modules in a 3D point cloud processing system; graph signal recovery and graph signal localization could be module for estimating an origin-destination matrix in a transportation management system.

[1] https://www.youtube.com/watch?v=0_wXsYKifBY

Chapter 14

Appendix

14.1 Proof of Theorem 14

We aim to construct a typical set of vectors in F, and use the Fano's method. Let $\mathbf{v}_k$ be the kth column of V, $\mathbf{v}^{(i)}$ be the ith row of V, $\mathcal{X}$ be a pruned hypercube and

$$\mathrm{F}' = \{\mathbf{x}^{(\mathbf{w})} = \mathrm{V}\,\widehat{\mathbf{x}} \odot \mathbf{w} = \sum_{k=\kappa_0}^{2\kappa_0-1} w_k \psi_k,\ \ \mathbf{w} \in \mathcal{X}\},$$

where κ_0 is no smaller than the bandwidth K,

$$\psi_k = \widehat{x}_k \mathbf{v}_k = (\pm)^k \sqrt{\frac{c\mu \left\|\mathbf{x}\right\|_2^2}{\kappa_0(1+k^{2\beta})}} \mathbf{v}_k,$$

and $0 < c < 1$. It is easy to check that $\mathrm{F}' \subseteq \mathrm{F}$. Let $d(\mathbf{x}, \mathbf{y}) = \left\|\mathbf{x} - \mathbf{y}\right\|_2$; we thus have

$$\begin{aligned}
d^2(\mathbf{x}^{(\mathbf{w})}, \mathbf{x}^{(\mathbf{u})}) &= \left\|\mathrm{V}\,\widehat{\mathbf{x}} \odot (\mathbf{w}-\mathbf{u})^2\right\|_2^2 \\
&= \sum_{k=\kappa_0}^{2\kappa_0-1} (w_k - u_k)^2 \widehat{x}_k^2 \\
&= \sum_{k=\kappa_0}^{2\kappa_0-1} (w_k - u_k)^2 \frac{c\mu \left\|\mathbf{x}\right\|_2^2}{\kappa_0(1+k^{2\beta})} \\
&\overset{(a)}{\geq} \sum_{k=\kappa_0}^{2\kappa_0-1} (w_k - u_k)^2 \cdot \frac{c\mu \left\|\mathbf{x}\right\|_2^2}{\kappa_0(1+(2\kappa_0)^{2\beta})} \\
&\overset{(b)}{\geq} \frac{\kappa_0}{8} \cdot \frac{c\mu \left\|\mathbf{x}\right\|_2^2}{\kappa_0(1+(2\kappa_0)^{2\beta})} \ \geq\ c_1 \mu \kappa_0^{-2\beta} \left\|\mathbf{x}\right\|_2^2,
\end{aligned}$$

where (a) follows from $k \leq 2K$, and (b) from the Varshamov-Gilbert lemma. To use the Fanno's method, we need to bound the Kullback-Leibler divergence,

$$
\begin{aligned}
& KL(p_{\mathbf{w}}, p_{\mathbf{w}_0}|\mathcal{M}) \\
= & \sum_{i\in\mathcal{M}} \mathbb{E}_{\mathbf{w}} \left[\log \frac{p(y_i - x_i^{(\mathbf{w})})}{p(y_i - x_i^{(\mathbf{w}_0)})} \right] \\
\leq & \sum_{i\in\mathcal{M}} \left[\frac{1}{2\sigma^2} (x_i^{(\mathbf{w})} - x_i^{(\mathbf{w}_0)})^2 \right] \\
= & \sum_{i\in\mathcal{M}} \left[\frac{1}{2\sigma^2} \left(\mathbf{v}^{(i)^*} (\widehat{\mathbf{x}} \odot \mathbf{w}) \right)^2 \right] \\
= & \frac{1}{2\sigma^2} \sum_{i\in\mathcal{M}} \left(\sum_{k=\kappa_0}^{2\kappa_0-1} \widehat{x}_k w_k \, \mathrm{V}_{ik} \right)^2 \\
= & \frac{1}{2\sigma^2} \sum_{i\in\mathcal{M}} \Bigg(\sum_{k=\kappa_0}^{2\kappa_0-1} (\widehat{x}_k w_k \, \mathrm{V}_{ik})^2 \\
& + \sum_{k,k'=\kappa_0, k\neq k'}^{2\kappa_0-1} \widehat{x}_k \widehat{x}_{k'} w_k w_{k'} \, \mathrm{V}_{ik} \, \mathrm{V}_{ik'} \Bigg) \\
\leq & \frac{1}{2\sigma^2} \sum_{i\in\mathcal{M}} \Bigg(\sum_{k=\kappa_0}^{2\kappa_0-1} \frac{c\mu \, \|\mathbf{x}\|_2^2 \, w_k^2 \, \mathrm{V}_{ik}^2}{2\kappa_0(1+k^{2\beta})} \\
& + \sum_{k,k'=\kappa_0, k\neq k'}^{2\kappa_0-1} (-1)^{k+k'} \frac{c\mu \, \|\mathbf{x}\|_2^2 \, w_k w_{k'} \, \mathrm{V}_{ik} \, \mathrm{V}_{ik'}}{\kappa_0 \sqrt{1+k^{2\beta}} \sqrt{1+k'^{2\beta}}} \Bigg) \\
\leq & \frac{c'\mu \, \|\mathbf{x}\|_2^2}{\sigma^2} \sum_{i\in\mathcal{M}} \sum_{k=\kappa_0}^{2\kappa_0-1} \frac{\mathrm{V}_{ik}^2}{\kappa_0(1+\kappa_0^{2\beta})} + \delta \\
\asymp & \frac{c'\mu \, \|\mathbf{x}\|_2^2}{\sigma^2} \kappa_0^{-(2\beta+1)} \sum_{i\in\mathcal{M}} \sum_{k=\kappa_0}^{2\kappa_0-1} \mathrm{V}_{ik}^2,
\end{aligned}
$$

where $\delta = \sum_{k,k'=\kappa_0,k\neq k'}^{2\kappa_0-1}(-1)^{k+k'}\frac{c\mu\|\mathbf{x}\|_2^2 w_k w_{k'}\,\mathrm{V}_{ik}\,\mathrm{V}_{ik'}}{\kappa_0\sqrt{1+k^{2\beta}}\sqrt{1+k'^{2\beta}}}$ is small because of the cross signs. For uniform sampling, the sampling set $\mathcal{M}$ is chosen randomly, thus, we have

$$\begin{aligned}
& KL(p_{\mathbf{w}}, p_{\mathbf{w}_0}) \\
= \;& \mathbb{E}_{\mathcal{M}}\left[KL(p_{\mathbf{w}}, p_{\mathbf{w}_0}|\mathcal{M})\right] \\
\leq \;& \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}\kappa_0^{-(2\beta+1)}\mathbb{E}_{\mathcal{M}}\left(\sum_{i\in\mathcal{M}}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{ik}^2\right) \\
\overset{(a)}{=} \;& \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}K^{-(2\beta+1)}\mathbb{E}_i\left(m\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{ik}^2\right) \\
= \;& \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}K^{-(2\beta+1)}m\sum_{j=1}^{N}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{jk}^2\,\mathbb{P}(j=i) \\
= \;& \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}\kappa_0^{-(2\beta+1)}\frac{m}{N}\sum_{j=1}^{N}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{jk}^2 \\
\leq \;& \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+1}N}\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2 m,
\end{aligned}$$

where (a) follows from the independence of each sample, $\mathbb{P}(j=i)$ denotes the probability to sample the ith node that equals j, and $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2 = \sum_{j=1}^{N}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{jk}^2$.

For experimentally designed sampling, we can choose the sampling set $\mathcal{M}$ to maximize $\sum_{i\in\mathcal{M}}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{ik}^2$; we thus have

$$\begin{aligned}
KL(p_{\mathbf{w}}, p_{\mathbf{w}_0}) \;&\leq\; \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}\kappa_0^{-(2\beta+1)}\max_{\mathcal{M}}\sum_{i\in\mathcal{M}}\sum_{k=\kappa_0}^{2\kappa_0-1}\mathrm{V}_{ik}^2 \\
&\leq\; \frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2}\kappa_0^{-(2\beta+1)}\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{2,\infty}^2 m.
\end{aligned}$$

For active sampling, we cannot get more benefit from signal coefficients, so the KL divergence is the same of the experimentally designed sampling.r By Fanno's lemma, we finally have the lower bounds for three sampling strategies. ■

14.2 Proof of Theorem 15

We aim to bound the MSE by splitting to a bias term and a variance term.

$$\begin{aligned}
& \mathbb{E}\left\|\mathbf{x}^* - \mathbf{x}\right\|_2^2 \\
= \;& \mathbb{E}\left\|\mathbf{x}^* - \mathbb{E}\mathbf{x}^* + \mathbb{E}\mathbf{x}^* - \mathbf{x}\right\|_2^2 \\
= \;& \mathbb{E}\left(\left\|\mathbf{x}^* - \mathbb{E}\mathbf{x}^*\right\|_2^2 + \left\|\mathbb{E}\mathbf{x}^* - \mathbf{x}\right\|_2^2 + 2(\mathbf{x}^* - \mathbb{E}\mathbf{x}^*)^T(\mathbb{E}\mathbf{x}^* - \mathbf{x})\right) \\
= \;& \left\|\mathbb{E}\mathbf{x}^* - \mathbf{x}\right\|_2^2 + \mathbb{E}\left\|\mathbf{x}^* - \mathbb{E}\mathbf{x}^*\right\|_2^2,
\end{aligned}$$

where the first term is bias and the second term is variance. For each element in the bias term, we have

$$\begin{aligned}
\mathbb{E}x_i^* &= \sum_{k<\kappa} \mathrm{V}_{ik}\, \mathbb{E}_{\mathcal{M},\epsilon} \left(\sum_{\mathcal{M}_j \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\, \mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j} (x_{\mathcal{M}_j} + \epsilon_{\mathcal{M}_j}) \right) \\
&\overset{(a)}{=} \sum_{k<\kappa} \mathrm{V}_{ik}\, m \mathbb{E}_\ell \left(\mathrm{U}_{k\ell}\, \frac{1}{m\pi_l} x_\ell \right) \\
&= \sum_{k<\kappa} \mathrm{V}_{ik}\, m \sum_{\ell=1}^{N} \left(\mathrm{U}_{k\ell}\, \frac{1}{m\pi_\ell} x_\ell \right) \pi_\ell \\
&= \sum_{k<\kappa} \mathrm{V}_{ik} \sum_{\ell=1}^{N} \mathrm{U}_{k\ell}\, x_\ell \\
&= \sum_{k<\kappa} \mathrm{V}_{ik}\, \widehat{x}_k,
\end{aligned}$$

where (a) follows from the independence of each sample. For all the elements, we have

$$\mathbb{E}\mathbf{x}^* = \mathrm{V}_{(\kappa)}\, \widehat{\mathbf{x}}_{(\kappa)},$$

which leads to Lemma 1.

We next bound the variance term by splitting into two parts, with and without noise. For each element in the variance term, we have

$$\begin{aligned}
& x_i^* - \mathbb{E}x_i^* \\
= & \sum_{k<\kappa} \mathrm{V}_{ik} \left(\sum_{\mathcal{M}_j \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\, \mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j}\, y_{\mathcal{M}_j} \right) - \sum_{k<\kappa} \mathrm{V}_{ik}\, \widehat{x}_k \\
= & \sum_{k<\kappa} \mathrm{V}_{ik} \sum_{\mathcal{M}_j \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\, \mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j}\, \epsilon_{\mathcal{M}_j} \\
& + \sum_{k<\kappa} \mathrm{V}_{ik} \left(\sum_{\mathcal{M}_j \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\, \mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j}\, x_{\mathcal{M}_j} - \widehat{x}_k \right) \\
= & \Delta_i^{(1)} + \Delta_i^{(2)},
\end{aligned}$$

where $\Delta^{(1)}$ is the variance from noise and $\Delta^{(2)}$ is the variance from sampling. To bound $\Delta_i^{(1)}$,

we have

$$
\begin{aligned}
& \mathbb{E}||\Delta_i^{(1)}||^2 \\
= \; & \mathbb{E}\Bigg[\Bigg(\sum_{k<\kappa} \mathrm{V}_{ik} \sum_{\mathcal{M}_j \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\, \mathrm{W}_{\mathcal{M}_j,\mathcal{M}_j}\, \epsilon_{\mathcal{M}_j}\Bigg) \\
& \Bigg(\sum_{k'<\kappa} \mathrm{V}_{ik'} \sum_{\mathcal{M}_{j'} \in \mathcal{M}} \mathrm{U}_{k'\mathcal{M}_{j'}}\, \mathrm{D}^2_{\mathcal{M}_{j'},\mathcal{M}_{j'}}\, \epsilon_{\mathcal{M}_{j'}}\Bigg)\Bigg] \\
= \; & \mathbb{E}\Bigg(\sum_{k,k'<\kappa} \mathrm{V}_{ik}\, \mathrm{V}_{ik'} \sum_{\mathcal{M}_j,\mathcal{M}_{j'} \in \mathcal{M}} \mathrm{U}_{k\mathcal{M}_j} \\
& \mathrm{U}_{k'\mathcal{M}_{j'}}\, \mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j}\, \mathrm{D}^2_{\mathcal{M}_{j'},\mathcal{M}_{j'}}\, \epsilon_{\mathcal{M}_j} \epsilon_{\mathcal{M}_{j'}}\Bigg) \\
= \; & \sum_{k,k'<\kappa} \mathrm{V}_{ik}\, \mathrm{V}_{ik'}\, m \mathbb{E}_{\ell,\epsilon}\left(\mathrm{U}^2_{k\ell}\, \frac{1}{m^2\pi_\ell^2} \epsilon_\ell^2\right) \\
= \; & \sum_{k,k'<\kappa} \mathrm{V}_{ik}\, \mathrm{V}_{ik'}\, m \sum_{\ell=1}^{N} \mathrm{U}_{k\ell}\, \mathrm{U}_{k'\ell}\, \frac{1}{m^2\pi_\ell^2} \mathbb{E}\epsilon_\ell^2 \pi_\ell \\
= \; & \sigma^2 \sum_{k,k'<\kappa} \mathrm{V}_{ik}\, \mathrm{V}_{ik'} \sum_{\ell=1}^{N} \frac{1}{m\pi_\ell}\, \mathrm{U}_{k\ell}\, \mathrm{U}_{k'\ell}\,.
\end{aligned}
$$

To bound $\Delta_i^{(2)}$, we have

$$
\begin{aligned}
& \mathbb{E}\left\|\Delta_i^{(2)}\right\|_2^2 \\
= \; & \mathbb{E}\Bigg[\sum_{k<\kappa} \mathrm{V}_{ik}\left(\sum_{\mathcal{M}_j\in\mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\,\mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j}\,x_{\mathcal{M}_j}-\widehat{x}_k\right) \\
& \sum_{k'<\kappa} \mathrm{V}_{ik'}\left(\sum_{\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{U}_{k'\mathcal{M}_{j'}}\,\mathrm{D}^2_{\mathcal{M}_{j'},\mathcal{M}_{j'}}\,x_{\mathcal{M}_{j'}}-\widehat{x}_{k'}\right)\Bigg] \\
= \; & \mathbb{E}\Bigg[\sum_{k,k'<K} \mathrm{V}_{ik}\,\mathrm{V}_{ik'}\Bigg(\sum_{\mathcal{M}_j,\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\,\mathrm{U}_{k'\mathcal{M}_{j'}}\,\mathrm{D}^2_{\mathcal{M}_j,\mathcal{M}_j} \\
& \mathrm{D}^2_{\mathcal{M}_{j'},\mathcal{M}_{j'}}\,x_{\mathcal{M}_j}x_{\mathcal{M}_{j'}}-\widehat{x}_k\widehat{x}_{k'}\Bigg)\Bigg] \\
= \; & \sum_{k,k'<\kappa} \mathrm{V}_{ik}\,\mathrm{V}_{ik'}\Bigg[\mathbb{E}_{\mathcal{M}}\Bigg(\sum_{\mathcal{M}_j\neq\mathcal{M}_{j'},\mathcal{M}_j,\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\,\mathrm{U}_{k'\mathcal{M}_{j'}} \\
& x_{\mathcal{M}_j}x_{\mathcal{M}_{j'}}\Bigg)+\mathbb{E}_{\mathcal{M}}\Bigg(\sum_{\mathcal{M}_j=\mathcal{M}_{j'},\mathcal{M}_j,\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{U}_{k\mathcal{M}_j}\,\mathrm{U}_{k'\mathcal{M}_{j'}} \\
& x_{\mathcal{M}_{j'}}x_{\mathcal{M}_{j'}}\Bigg)-\widehat{x}_k\widehat{x}_{k'}\Bigg] \\
= \; & \sum_{k,k'<\kappa} \mathrm{V}_{ik}\,\mathrm{V}_{ik'}\Bigg[(m^2-m)\mathbb{E}_{\ell,\ell'}\left(\frac{\mathrm{U}_{k\ell}\,\mathrm{U}_{k'\ell'}\,x_\ell x_{\ell'}}{m^2\pi_\ell\pi_{\ell'}}\right) \\
& +m\mathbb{E}_{\ell}\left(\frac{\mathrm{U}_{k\ell}\,\mathrm{U}_{k'\ell}\,x_\ell^2}{m^2\pi_\ell^2}\right)-\widehat{x}_k\widehat{x}_{k'}\Bigg] \\
= \; & \sum_{k,k'<\kappa} \mathrm{V}_{ik}\,\mathrm{V}_{ik'}\Bigg[(m^2-m)\sum_{\ell,\ell'=1}^{N}\frac{\mathrm{U}_{k\ell}\,\mathrm{U}_{k\ell'}\,x_\ell x_{\ell'}}{m^2\pi_\ell\pi_{\ell'}}\pi_\ell\pi_{\ell'} \\
& +m\sum_{\ell=1}^{N}\frac{\mathrm{U}_{k\ell}\,\mathrm{U}_{k'\ell}\,x_\ell^2}{m^2\pi_\ell^2}\pi_\ell-\widehat{x}_k\widehat{x}_{k'}\Bigg] \\
= \; & \sum_{k,k'<\kappa}\left(\mathrm{V}_{ik}\,\mathrm{V}_{ik'}\sum_{\ell=1}^{N}\mathrm{U}_{k\ell}\,\mathrm{U}_{k'\ell}\,\frac{x_\ell^2}{m\pi_\ell}-\frac{1}{m}\widehat{x}_k\widehat{x}_{k'}\right).
\end{aligned}
$$

We combine the bounds for both $\Delta_i^{(1)}$ and $\Delta_i^{(2)}$, and obtain the bounds for the variance term,

$$
\begin{aligned}
& \mathbb{E}\left\|\mathbf{x}^* - \mathbb{E}\mathbf{x}^*\right\|_2^2 \\
= & \sum_{i=1}^{N} \mathbb{E}\left\|x_i^* - \mathbb{E}x_i^*\right\| \\
= & \sum_{i=1}^{N} \left(\mathbb{E}\left\|\Delta_i^{(1)}\right\|_2^2 + \mathbb{E}\left\|\Delta_i^{(2)}\right\|_2^2 \right) \\
= & \sum_{i=1}^{N} \sum_{k,k'<\kappa} \mathrm{V}_{ik}\,\mathrm{V}_{ik'} \sum_{\ell=1}^{N} \mathrm{U}_{k\ell}\,\mathrm{U}_{k\ell'}\,\frac{\sigma^2 + x_\ell^2}{m\pi_\ell} - \frac{1}{m} \sum_{k,k'<\kappa} \widehat{x}_k \widehat{x}_{k'} \\
= & \mathrm{Tr}\left(\mathrm{V}_{(\kappa)}\,\mathrm{U}_{(\kappa)}\,\mathrm{W}_C\,\mathrm{U}_{(\kappa)}^T\,\mathrm{V}_{(\kappa)}^T\right) - \frac{1}{m}\left\|\widehat{\mathbf{x}}_{(\kappa)}\right\|_2^2 \\
= & \mathrm{Tr}\left(\mathrm{U}_{(\kappa)}\,\mathrm{W}_C\,\mathrm{V}_{(\kappa)}\right) - \frac{1}{m}\left\|\widehat{\mathbf{x}}_{(\kappa)}\right\|_2^2,
\end{aligned}
$$

which leads to Lemma 2. Finally, we obtain the MSE by combine the bias term and the variance term,

$$
\begin{aligned}
& \mathbb{E}\left\|\mathbf{x}^* - \mathbf{x}\right\|_2^2 \\
= & \left\|\mathbb{E}\mathbf{x}^* - \mathbf{x}\right\|_2^2 + \mathbb{E}\left\|\mathbf{x}^* - \mathbb{E}\mathbf{x}^*\right\|_2^2 \\
= & \left\|\mathrm{V}_{(-\kappa)}\,\widehat{x}_{(-\kappa)}\right\|_2^2 + \mathrm{Tr}\left(\mathrm{U}_{(\kappa)}\,\mathrm{W}_C\,\mathrm{V}_{(\kappa)}\right) - \frac{1}{m}\left\|\widehat{\mathbf{x}}_{(\kappa)}\right\|_2^2 \\
\leq & \frac{1}{1+\kappa^{2\beta}} \sum_{k\geq\kappa} \widehat{x}_k^2 (1+k^{2\beta}) + \mathrm{Tr}\left(\mathrm{U}_{(\kappa)}\,\mathrm{W}_C\,\mathrm{V}_{(\kappa)}\right) \\
\leq & \frac{\mu}{1+\kappa^{2\beta}}\left\|\mathbf{x}\right\|_2^2 + \mathrm{Tr}\left(\mathrm{U}_{(\kappa)}\,\mathrm{W}_C\,\mathrm{V}_{(\kappa)}\right).
\end{aligned}
$$

■

14.3 Proof of Corollaries 5 and 6

For a Type-1 graph, we assume that each element in an approximately bandlimited signal has a similar magnitude. Since all the elements in V have the same magnitude, each element of a graph signal, $x_i = \mathbf{v}_i^T \widehat{\mathbf{x}}$, should have a similar magnitude. In other words, $N \max_i x_i^2$ and $\|\mathbf{x}\|_2^2$ are of the same order.

For uniform sampling, based on Corollary 1, we have

$$\begin{aligned}
& \frac{\mu}{1+\kappa^{2\beta}}\left\|\mathbf{x}\right\|_2^2+\frac{N}{m}\sum_{k=1}^{\kappa}\sum_{i=1}^{N}\mathrm{U}_{k,i}^2\left(x_i^2+\sigma^2\right)\\
\leq\ & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{\kappa^{2\beta}}+\frac{N\left(\max_i x_i^2+\sigma^2\right)}{m\left\|\mathbf{x}\right\|_2^2}\sum_{k=1}^{\kappa}\sum_{i=1}^{N}\mathrm{U}_{k,i}^2\right)\\
\overset{(a)}{=}\ & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{\kappa^{2\beta}}+\frac{N(\max_i x_i^2+\sigma^2)}{\left\|\mathbf{x}\right\|_2^2}\frac{\kappa}{m}\right)\\
\leq\ & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{\kappa^{2\beta}}+\frac{c\kappa}{m}\right)\\
\asymp\ & C\left\|\mathbf{x}\right\|_2^2 m^{-\frac{2\beta}{2\beta+1}},
\end{aligned}$$

where (a) follows from U being orthornormal, κ being of the order of $m^{\frac{1}{2\beta+1}}$, and $C>0$ some constant. Since at least the sampled projection estimator satisfies this rate of convergence, we have

$$\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{u}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\frac{\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2}{\left\|\mathbf{x}\right\|_2^2}\right)\leq Cm^{-\frac{2\beta}{2\beta+1}}.$$

We next show the lower bound. Based on Theorem 3, we have

$$\begin{aligned}
& \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{u}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2\right)\\
\geq\ & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+2}}\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2 m\right),\\
=\ & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+1}N}m\right)\\
\geq\ & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\max_i x_i^2}{\sigma^2\kappa_0^{2\beta+1}}m\right)\\
\asymp\ & c\left\|\mathbf{x}\right\|_2^2 m^{-\frac{2\beta}{2\beta+1}},
\end{aligned}$$

where κ_0 is of the order of $m^{\frac{1}{2\beta+1}}$.

For optimal sampling scores based sampling, based on Corollary 2, we have

$$\begin{aligned}
& \frac{\mu}{1+\kappa^{2\beta}}\left\|\mathbf{x}\right\|_2^2+\frac{1}{m}\left(\sum_{i=1}^{N}\sqrt{\sum_{k=1}^{\kappa}\mathrm{U}_{k,i}^2\left(x_i^2+\sigma^2\right)}\right)^2 \\
\leq & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{1+\kappa^{2\beta}}+\frac{1}{m}\frac{\max_i x_i^2+\sigma^2}{\left\|\mathbf{x}\right\|_2^2}\left\|\mathrm{U}_{(\kappa)}\right\|_{2,1}\right) \\
\overset{(a)}{\leq} & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{1+\kappa^{2\beta}}+\frac{\max_i x_i^2+\sigma^2}{\left\|\mathbf{x}\right\|_2^2}\frac{N\kappa}{m}\right) \\
\asymp & C\left\|\mathbf{x}\right\|_2^2 m^{-\frac{2\beta}{2\beta+1}},
\end{aligned}$$

where (a) follows from that, based on Definition 5, $\left\|\mathrm{U}_{(\kappa)}\right\|_{2,1}^2=\left(N\sqrt{\kappa(\frac{c}{\sqrt{N}})^2}\right)^2=O(N\kappa)$, which is of the same order of $N\left\|\mathrm{U}_{(\kappa)}\right\|_F^2$. Since at least the sampled projection estimator satisfies this rate of convergence, we have

$$\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\frac{\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2}{\left\|\mathbf{x}\right\|_2^2}\right)\leq Cm^{-\frac{2\beta}{2\beta+1}}.$$

Based on Definition 5, $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2=\kappa_0(\frac{c}{\sqrt{N}})^2=c^2\kappa_0/N$, which is of the same order of $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_F^2/N$, we have

$$\begin{aligned}
& \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2\right) \\
\geq & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+2}}\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2 m\right), \\
= & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+1}N}m\right) \\
\geq & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\max_i x_i^2}{\sigma^2\kappa_0^{2\beta+1}}m\right) \\
\asymp & c\left\|\mathbf{x}\right\|_2^2 m^{-\frac{2\beta}{2\beta+1}},
\end{aligned}$$

where κ_0 is of the order of $m^{\frac{1}{2\beta+1}}$. ■

14.4 Proof of Corollaries 7 and 8

For a Type-2 graph, we assume that a few elements in an approximately bandlimited signal have much higher magnitudes than the others. The intuition is that, since V is sparse, the energy concentrates in $O(\kappa)$ rows of $\mathrm{V}_{(\kappa)}$, thus, $O(\kappa)$ components of an approximately bandlimited

signal, $x_i \approx \mathbf{v}_{i,(\kappa)}^T \widehat{\mathbf{x}}_{(\kappa)}$, have much higher magnitudes than the others. In other words, $\kappa \max_i x_i^2$ and $\|\mathbf{x}\|_2^2$ are of the same order.

For uniform sampling, based on Corollary 1, we have

$$\begin{aligned}
& \frac{\mu}{1+\kappa^{2\beta}} \|\mathbf{x}\|_2^2 + \frac{N}{m} \sum_{k=1}^{\kappa} \sum_{i=1}^{N} \mathrm{U}_{k,i}^2 \left(x_i^2 + \sigma^2\right) \\
\leq & \|\mathbf{x}\|_2^2 \left(\frac{\mu}{\kappa^{2\beta}} + \frac{N\left(\max_i x_i^2 + \sigma^2\right)}{m \|\mathbf{x}\|_2^2} \sum_{k=1}^{\kappa} \sum_{i=1}^{N} \mathrm{U}_{k,i}^2 \right) \\
= & \|\mathbf{x}\|_2^2 \left(\frac{\mu}{\kappa^{2\beta}} + \frac{N(\max_i x_i^2 + \sigma^2)}{\kappa \max_i x_i^2} \frac{\kappa}{m} \right) \\
\leq & \|\mathbf{x}\|_2^2 \left(\frac{\mu}{\kappa^{2\beta}} + \frac{cN}{m} \right) \\
\asymp & C \|\mathbf{x}\|_2^2 m^{-\frac{2\beta}{2\beta+\gamma}},
\end{aligned}$$

where γ varies with κ to satisfy $\kappa^\gamma \leq N$ ($\gamma > 1$) and κ is of the order of $m^{\frac{1}{2\beta+\gamma}}$, and $C > 0$ is some constant. Since at least Algorithm 1 satisfies this rate of convergence, we thus have

$$\inf_{(\mathbf{x}^*,\mathcal{M}) \in \Theta_{\mathrm{u}}} \sup_{\mathbf{x} \in \mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\|\mathbf{x}^* - \mathbf{x}\|_2^2 \right) \leq C m^{-\frac{2\beta}{2\beta+\gamma}}.$$

We next show the lower bound. Based on Theorem 3, we have

$$\begin{aligned}
& \inf_{(\mathbf{x}^*,\mathcal{M}) \in \Theta_{\mathrm{u}}} \sup_{\mathbf{x} \in \mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)} \mathbb{E}_{\mathbf{x},\mathcal{M}} \left(\|\mathbf{x}^* - \mathbf{x}\|_2^2 \right) \\
\geq & \frac{c_1 \mu \|\mathbf{x}\|_2^2}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu \|\mathbf{x}\|_2^2}{\sigma^2 \kappa_0^{2\beta+2} N} \left\| \mathrm{V}_{(2,\kappa_0)} \right\|_F^2 m \right), \\
= & \frac{c_1 \mu \|\mathbf{x}\|_2^2}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu\kappa_0 \max_i x_i^2}{\sigma^2 \kappa_0^{2\beta+1} N} m \right) \\
= & \frac{c_1 \mu \|\mathbf{x}\|_2^2}{\kappa_0^{2\beta}} \left(1 - \frac{c\mu \max_i x_i^2}{\sigma^2 \kappa_0^{2\beta+\gamma}} m \right) \\
\asymp & c m^{-\frac{2\beta}{2\beta+\gamma}},
\end{aligned}$$

where κ_0 is of the order of $m^{\frac{1}{2\beta+\gamma}}$.

For optimal sampling scores based sampling, based on Corollary 2, we have

$$\begin{aligned}
& \frac{\mu}{1+\kappa^{2\beta}}\left\|\mathbf{x}\right\|_2^2+\frac{1}{m}\left(\sum_{i=1}^{N}\sqrt{\sum_{k=1}^{\kappa}\mathrm{U}_{k,i}^2\left(x_i^2+\sigma^2\right)}\right)^2 \\
\leq & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{1+\kappa^{2\beta}}+\frac{1}{m}\frac{\max_i x_i^2+\sigma^2}{\left\|\mathbf{x}\right\|_2^2}\left\|\mathrm{U}_{(\kappa)}\right\|_{2,1}^2\right) \\
\overset{(a)}{\leq} & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{1+\kappa^{2\beta}}+\frac{\max_i x_i^2+\sigma^2}{\left\|\mathbf{x}\right\|_2^2}\frac{\kappa^2}{m}\right) \\
= & \left\|\mathbf{x}\right\|_2^2\left(\frac{\mu}{1+\kappa^{2\beta}}+\frac{\max_i x_i^2+\sigma^2}{\kappa\max_i x_i^2}\frac{\kappa^2}{m}\right) \\
\asymp & C\left\|\mathbf{x}\right\|_2^2 m^{-\frac{2\beta}{2\beta+1}},
\end{aligned}$$

where (a) follows from the energy concentrating in $O(\kappa)$ columns of $\mathrm{U}_{(\kappa)}$ as shown in Definition 6 and the upper bound reaching the minimum when κ is of the order of $m^{\frac{1}{2\beta+1}}$. Since at least Algorithm 1 satisfies this rate of convergence, we have

$$\inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2\right)\leq Cm^{-\frac{2\beta}{2\beta+1}}.$$

Based on Definition 6, $\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2=c$, we have

$$\begin{aligned}
& \inf_{(\mathbf{x}^*,\mathcal{M})\in\Theta_{\mathrm{e}}}\sup_{\mathbf{x}\in\mathrm{ABL}_{\mathrm{A}}(K,\beta,\mu)}\mathbb{E}_{\mathbf{x},\mathcal{M}}\left(\left\|\mathbf{x}^*-\mathbf{x}\right\|_2^2\right) \\
\geq & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\left\|\mathbf{x}\right\|_2^2}{\sigma^2\kappa_0^{2\beta+2}}\left\|\mathrm{V}_{(2,\kappa_0)}\right\|_{\infty,2}^2 m\right), \\
= & \frac{c_1\mu\left\|\mathbf{x}\right\|_2^2}{\kappa_0^{2\beta}}\left(1-\frac{c\mu\kappa_0\max_i x_i^2}{\sigma^2\kappa_0^{2\beta+2}}m\right) \\
\asymp & cm^{-\frac{2\beta}{2\beta+1}}.
\end{aligned}$$

■

14.5 Proof of Theorem 21

Under H_0, the observation is $\mathbf{y}=\mathrm{Bernoulli}(\epsilon\mathbf{1}_{\mathcal{V}})$. Let the ith wavelet basis vector be

$$\begin{aligned}
\mathbf{w}_i &= \sqrt{\frac{|S_1^{(i)}||S_2^{(i)}|}{|S_1^{(i)}|+|S_2^{(i)}|}}\left(\frac{\mathbf{1}_{S_1^{(i)}}}{|S_1^{(i)}|}-\frac{\mathbf{1}_{S_2^{(i)}}}{|S_2^{(i)}|}\right) \\
&= \frac{\sqrt{|S^{(i)}|}}{2}\left(\frac{\mathbf{1}_{S_1^{(i)}}}{|S_1^{(i)}|}-\frac{\mathbf{1}_{S_2^{(i)}}}{|S_2^{(i)}|}\right)
\end{aligned}$$

where $S_1^{(i)}, S_2^{(i)}$ are two local child sets that form $\mathbf{w}_i$ and $S^{(i)}$ is the parent local set. The second equality follows from the even partition. Thus,

$$\mathbf{w}_i^T \mathbf{y} \;=\; \frac{\sqrt{|S^{(i)}|}}{2} \left(\frac{\mathbf{1}_{S_1^{(i)}}^T \mathbf{y}}{|S_1^{(i)}|} - \frac{\mathbf{1}_{S_2^{(i)}}^T \mathbf{y}}{|S_2^{(i)}|} \right).$$

Since each element in $\mathbf{y}$ is a Bernoulli random variable that takes value one with probability ϵ, the first term in the parentheses is the mean of $|S_1^{(i)}|$ Bernoulli random variables with success probability of ϵ. We then can show that

$$\mathbb{E}\left(\frac{\mathbf{1}_{S_1^{(i)}}^T \mathbf{y}}{|S_1^{(i)}|} \right) = \frac{|S_1^{(i)}|\epsilon}{|S_1^{(i)}|} = \epsilon.$$

Following from the Hoeffding inequality [184], for any η,

$$\mathbb{P}\left(\left| \frac{\mathbf{1}_{S_1^{(i)}}^T \mathbf{y}}{|S_1^{(i)}|} - \epsilon \right| \geq \eta \right) \leq 2e^{-2|S_1^{(i)}|\eta^2}.$$

Thus, $(\mathbf{1}_{S_1^{(i)}}^T \mathbf{y}/|S_1^{(i)}| - \epsilon)$ is $1/(2\sqrt{|S_1^{(i)}|})$-sub-Gaussian random variable. Combing two terms, we obtain that $\mathbf{w}_i^T \mathbf{y}$ is $\sqrt{2}/2$-sub-Gaussian random variable. Then,

$$\mathbb{E}\left(\left\| \mathrm{W}_{(-1)}^T \mathbf{y} \right\|_\infty \right) = \mathbb{E}\left(\max_{2 \leq i \leq N} \mathbf{w}_i^T \mathbf{y} \right) \leq \sqrt{\log N}.$$

Finally, by the Cirelson's theorem [184], with probability at least $1 - \delta$,

$$\left\| \mathrm{W}_{(-1)}^T \mathbf{y} \right\|_\infty \leq \mathbb{E}\left(\left\| \mathrm{W}_{(-1)}^T \mathbf{y} \right\|_\infty \right) + \sqrt{2\log(1/\delta)}.$$

Under H_1, the observation is $\mathbf{y} = \mathrm{Bernoulli}(\mu \mathbf{1}_C + \epsilon \mathbf{1}_{\mathcal{V}})$. We need to show there exists at least one wavelet basis vector capturing C well. We start with the following lemmas.

Lemma 8.

$$\left\| \mathrm{W}_{(-1)}^T \mathbf{1}_C \right\|_\infty \;\geq\; \sqrt{\frac{|C|(1 - |C|/N)}{1 + \rho \log N}}.$$

Proof.

$$\begin{aligned}
\left\| \mathrm{W}_{(-1)}^T \mathbf{1}_C \right\|_\infty^2 &\geq \frac{\left\| \mathrm{W}_{(-1)}^T \mathbf{1}_C \right\|_2^2}{\left\| \mathrm{W}_{(-1)}^T \mathbf{1}_C \right\|_0} \\
&\geq \frac{\left\| \mathrm{W}^T \mathbf{1}_C \right\|_2^2 - \left(\frac{1}{\sqrt{N}} \mathbf{1}_{\mathcal{V}}^T \mathbf{1}_C \right)^2}{1 + \rho \log N} \\
&\geq \frac{\|\mathbf{1}_C\|_2^2 - |C|^2/N}{1 + \rho \log N} \;\geq\; \frac{|C|(1 - |C|/N)}{1 + \rho \log N}.
\end{aligned}$$

The second inequality follows from Theorem 8 with $\|\Delta \mathbf{1}_C\|_0 \leq \rho$ and the even partition.

□

Let

$$\begin{aligned} i &= \arg\max_{2\le j\le N} \mathbf{w}_j^T \mathbf{1}_C \\ &= \arg\max_{2\le j\le N} \frac{\sqrt{S^{(j)}}}{2}\left(\frac{|S_1^{(j)}\cap C|}{|S_1^{(j)}|} - \frac{|S_2^{(j)}\cap C|}{|S_2^{(j)}|}\right), \end{aligned}$$

where $S_1^{(j)}, S_2^{(j)}$ are the children local sets of $S^{(j)}$. Following from Lemma 8,

$$\frac{\sqrt{S^{(i)}}}{2}\left(\frac{|S_1^{(i)}\cap C|}{|S_1^{(i)}|} - \frac{|S_2^{(i)}\cap C|}{|S_2^{(i)}|}\right) \ge \sqrt{\frac{|C|(1-|C|/N)}{1+\rho\log N}}.$$

Thus, the assumption can be reformulated as

$$\mu - \epsilon \ge \frac{(\sqrt{\log N} + 2\sqrt{2\log(\frac{2}{\delta})})}{\frac{\sqrt{S^{(i)}}}{2}\left(\frac{|S_1^{(i)}\cap C|}{|S_1^{(i)}|} - \frac{|S_2^{(i)}\cap C|}{|S_2^{(i)}|}\right)}, \tag{14.1}$$

Lemma 9. When $a \sim \text{Binomial}\,(n_1, p_1)\,, b \sim \text{Binomial}\,(n_2, p_1)$, with probability $(1-\delta)^2$,

$$|a+b-(n_1p_1+n_2p_2)| \;\le\; \sqrt{\frac{n_1}{2}\log(\frac{2}{\delta})} + \sqrt{\frac{n_2}{2}\log(\frac{2}{\delta})}.$$

Proof. Following from the Hoeffding inequality, with probability $1-\delta$, $|a-n_1p_1| \;\le\; \sqrt{n_1\log(2/\delta)/2}$. We bound both a and b, and rearrange the terms to obtain Lemma 9. □

We aim to show that $|\mathbf{w}_i^T\mathbf{y}|$ is sufficiently large. Here, the term $\mathbf{1}_{S_1^{(i)}}^T\mathbf{y}$ counts how many ones appear inside the local set $S_1^{(i)}$ and is a random variable under the distribution of $\text{Binomial}\left(|S_1^{(i)}\cap C|, \mu\right) + \text{Binomial}\left(|S_1^{(i)}\cap \overline{C}|, \epsilon\right)$. Following Lemma 9, with probability $(1-\delta)^2$,

$$\begin{aligned} &\left|\mathbf{1}_{S_1^{(i)}}^T\mathbf{y} - \left(|S_1^{(i)}\cap C|\mu + |S_1^{(i)}\cap \overline{C}|\epsilon|\right)\right| \\ \le\; &\sqrt{\frac{|S_1^{(i)}\cap C|}{2}\log(\frac{2}{\delta})} + \sqrt{\frac{|S_1^{(i)}\cap \overline{C}|}{2}\log(\frac{2}{\delta})}. \end{aligned}$$

Similarly, we have with probability $(1-\delta)^2$,

$$\begin{aligned} &\left|\mathbf{1}_{S_2^{(i)}}^T\mathbf{y} - \left(|S_2^{(i)}\cap C|\mu + |S_2^{(i)}\cap \overline{C}|\epsilon|\right)\right| \\ \le\; &\sqrt{\frac{|S_2^{(i)}\cap C|}{2}\log(\frac{2}{\delta})} + \sqrt{\frac{|S_2^{(i)}\cap \overline{C}|}{2}\log(\frac{2}{\delta})}. \end{aligned}$$

Without losing generality, we assume that $|S_1^{(i)} \cap C| \geq |S_2^{(i)} \cap C|$. With probability $(1-\delta)^4$,

$$\begin{aligned}\mathbf{1}_{S_1^{(i)}}^T \mathbf{y} &\geq |S_1^{(i)} \cap C|\mu + |S_1^{(i)} \cap \overline{C}|\epsilon \\ &\quad - \left(\sqrt{|S_1^{(i)} \cap C|} + \sqrt{|S_1^{(i)} \cap \overline{C}|}\right)\sqrt{\frac{1}{2}\log(\frac{2}{\delta})},\end{aligned}$$

and

$$\begin{aligned}\mathbf{1}_{S_2^{(i)}}^T \mathbf{y} &\leq |S_2^{(i)} \cap C|\mu + |S_2^{(i)} \cap \overline{C}|\epsilon \\ &\quad + \left(\sqrt{|S_2^{(i)} \cap C|} + \sqrt{|S_2^{(i)} \cap \overline{C}|}\right)\sqrt{\frac{1}{2}\log(\frac{2}{\delta})}.\end{aligned}$$

Denote $\alpha_1^{(i)} = |S_1^{(i)} \cap C|/|S_1^{(i)}|, \alpha_2^{(i)} = |S_2^{(i)} \cap C|/|S_2^{(i)}|$, where $\mu_1^{(i)} = \alpha_1^{(i)}\mu + (1-\alpha_1^{(i)})\epsilon$, $\mu_2^{(i)} = \alpha_2^{(i)}\mu + (1-\alpha_2^{(i)})\epsilon$. We have

$$\begin{aligned}|\mathbf{w}_i^T \mathbf{y}| &= |\frac{1}{\sqrt{|S^{(i)}|}}\left(\mathbf{1}_{S_1^{(i)}}^T \mathbf{y} - \mathbf{1}_{S_2^{(i)}}^T \mathbf{y}\right)| \\ &\geq \frac{\sqrt{|S^{(i)}|}}{2}|\mu_1^{(i)} - \mu_2^{(i)}| - \sqrt{2\log(\frac{2}{\delta})} \\ &\geq \sqrt{\log N} + \sqrt{2\log(\frac{2}{\delta})}.\end{aligned}$$

The last inequality follows from (14.1). Finally, with probability at least $(1-\delta)^4$,

$$\left\|\mathrm{W}_{(-1)}^T \mathbf{y}\right\|_\infty \geq |\mathbf{w}_i^T \mathbf{y}| \geq \sqrt{\log N} + \sqrt{2\log(\frac{2}{\delta})}.$$

■

14.6 Proof of Theorem 22

Under H_0, the observation is $\mathbf{y} = \text{Bernoulli}(\epsilon \mathbf{1}_\mathcal{V})$. Let $\mathcal{C} = \{C : \|\Delta \mathbf{1}_C\|_1 \leq \rho\}$.

$$\begin{aligned}\widehat{g} &= \max_{C\in\mathcal{C}} |C|\,\mathrm{KL}\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}||\epsilon\right) \leq 8\max_{C\in\mathcal{C}}|C|\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|} - \epsilon\right)^2 \\ &\approx 8\left(\max_{C\in\mathcal{C}}\frac{\mathbf{1}_C^T(\mathbf{y}-\epsilon)}{\sqrt{|C|}}\right)^2.\end{aligned}$$

In the last step, we only consider one side of the distribution. We can evaluate both sides and take the maximum, but this is nearly the same. By using the Hoeffding inequality, we can show

that $\mathbf{1}_C^T(\mathbf{y}-\epsilon)/\sqrt{|C|}$ is a $1/2$-sub-Gaussian random variable. Similarly to Theorem 5 in [178], we can show that with probability $1-\delta$,

$$\begin{aligned}\max_{C\in\mathcal{C}}\frac{\mathbf{1}_C^T(\mathbf{y}-\epsilon)}{\sqrt{|C|}} &\leq \left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\\ &\quad+\sqrt{2\log 2}+\sqrt{2\log\frac{1}{\delta}}.\end{aligned}$$

Theorem 5 [178] is concerned with a Gaussian variable and here we are concerned with a sub-Gaussian variable. Thus, under the null hypothesis H_0, with probability $1-\delta$,

$$\begin{aligned}\widehat{g} &\leq 8\Bigg(\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\\ &\quad+\sqrt{2\log 2}+\sqrt{2\log(1/\delta)}\Bigg)^2.\end{aligned}$$

Under H_1, the observation is $\mathbf{y}=\text{Bernoulli}(\mu\mathbf{1}_C+\epsilon\mathbf{1}_{\overline{C}})$. Following from the Hoeffding inequality, with probability $1-\delta$,

$$|\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}-\mu|\leq\sqrt{\frac{1}{2|C|}\log(2/\delta)}.$$

Then,

$$\begin{aligned}\widehat{g} &= \max_{C\in\mathcal{C}}|C|\,\text{KL}\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}||\epsilon\right)\geq\max_{C\in\mathcal{C}}|C|\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}-\epsilon\right)^2\\ &\geq |C|\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}-\epsilon\right)^2\\ &\geq |C|\left(\mu-\epsilon-\sqrt{\frac{\log(2/\delta)}{2|C|}}\right)^2.\end{aligned}$$

When

$$\begin{aligned}\mu-\epsilon &\geq \frac{2\sqrt{2}}{\sqrt{|C|}}\Bigg(\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)\sqrt{2\log(N-1)}\\ &\quad+\sqrt{2\log 2}+\sqrt{2\log(1/\delta)}\Bigg)+\sqrt{\frac{\log(2/\delta)}{2|C|}},\end{aligned}$$

we can distinguish H_1 from H_0. ■

14.7 Proof of Theorem 23

Under H_0, the observation is $\mathbf{y} = \text{Bernoulli}(\epsilon \mathbf{1}_{\mathcal{V}})$. Let $\mathcal{X}_t = \{\mathbf{x} : \mathbf{x} \in [0,1]^N, \|\Delta \mathbf{1}_C\|_1 \leq \rho, \mathbf{1}^T\mathbf{x} \leq t.\}$.

$$\begin{aligned}
\widehat{r} &= \max_{t,\mathbf{x}\in\mathcal{X}_t} t\,\mathrm{KL}\left(\frac{\mathbf{x}^T\mathbf{y}}{t}||\epsilon\right) \\
&\leq 8 \max_{t,\mathbf{x}\in\mathcal{X}_t} t\left(\frac{\mathbf{x}^T\mathbf{y}}{t} - \epsilon\right)^2 \\
&\leq 8 \max_{t,\mathbf{x}\in\mathcal{X}_t} t\left(\frac{\mathbf{x}^T\mathbf{y}}{t} - \frac{\mathbf{x}^T\mathbf{1}}{t}\epsilon\right)^2 \\
&\approx 8\left(\max_{t,\mathbf{x}\in\mathcal{X}_t} \frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{\sqrt{t}}\right)^2,
\end{aligned}$$

where the second inequality follows from that $\mathbf{1}^T\mathbf{x} \leq t$ and in the the last step, we only consider one side of the distribution. We can evaluate both sides and take the maximum, but this is nearly the same.

Lemma 10.

$$\mathbb{E}_{\mathbf{y}} \max_{t,\mathbf{x}\in[0,1]^N,\mathbf{1}^T\mathbf{x}\leq t} \frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{\sqrt{t}} \leq \sqrt{\frac{N\log 2}{2}}.$$

Proof.

$$\begin{aligned}
&\mathbb{E}_{\mathbf{y}} \max_{t,\mathbf{x}\in[0,1]^N,\mathbf{1}^T\mathbf{x}\leq t} \frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{\sqrt{t}} \\
=\ &\mathbb{E}_{\mathbf{y}} \max_t \frac{1}{\sqrt{t}} \max_{\mathbf{x}\in[0,1]^N,\mathbf{1}^T\mathbf{x}\leq t} \mathbf{x}^T(\mathbf{y}-\epsilon) \\
=\ &\mathbb{E}_{\mathbf{y}} \max_t \frac{1}{\sqrt{t}} \max_{\mathbf{x}\in\{0,1\}^N,\mathbf{1}^T\mathbf{x}\leq t} \mathbf{x}^T(\mathbf{y}-\epsilon) \\
=\ &\mathbb{E}_{\mathbf{y}} \max_{\mathbf{x}\in\{0,1\}^N} \frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{\sqrt{\mathbf{1}^T\mathbf{x}}}.
\end{aligned}$$

When $\mathbf{x}$ is a binary variable, $\mathbf{x}^T(\mathbf{y}-\epsilon)/\sqrt{\mathbf{1}^T\mathbf{x}}$ is a 1/2-sub-Gaussian random variable as shown previously. The cardinality of the set $\{0,1\}^N$ is 2^N. Thus,

$$\mathbb{E}_{\mathbf{y}} \max_{t,\mathbf{x}\in[0,1]^N,\mathbf{1}^T\mathbf{x}\leq t} \frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{\sqrt{t}} \leq \sqrt{\frac{1}{2}\log 2^N}.$$

□

Combining Lemma 10 and to Theorem 5 in [178], we can show that with probability $1-\delta$,

$$\max_{t,\mathbf{x}\in\mathcal{X}_t}\frac{\mathbf{x}^T(\mathbf{y}-\epsilon)}{t} \;\leq\; \frac{\log 2N+1}{\sqrt{\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)^2\log N}} +2\sqrt{\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)^2\log N}+\sqrt{2\log 2}+\sqrt{2\log\frac{1}{\delta}}.$$

Thus, under the null hypothesis H_0, with probability $1-\delta$,

$$\widehat{r} \;\leq\; 8\left(\frac{\log 2N+1}{\sqrt{\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)^2\log N}}+\sqrt{2\log 2} +2\sqrt{\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)^2\log N}+\sqrt{2\log(1/\delta)}\right)^2.$$

Under H_1, the observation is $\mathbf{y}=\text{Bernoulli}(\mu\mathbf{1}_C+\epsilon\mathbf{1}_{\overline{C}})$. Let $t^*=|C|,\mathbf{x}^*=\mathbf{1}_C$. Similarly to the proof in Theorem 22,

$$\begin{aligned}\widehat{r} \;&\geq\; t^*\,\text{KL}\left(\frac{\mathbf{y}^T\mathbf{x}^*}{t^*}||\epsilon\right) \;=\; |C|\,\text{KL}\left(\frac{\mathbf{1}_C^T\mathbf{y}}{|C|}||\epsilon\right)\\ &\geq\; |C|\left(\mu-\epsilon-\sqrt{\frac{\log(2/\delta)}{2|C|}}\right)^2.\end{aligned}$$

When

$$\mu-\epsilon \;\geq\; \frac{2\sqrt{2}}{\sqrt{|C|}}\left(\frac{\log 2N+1}{\sqrt{\left(\rho+\sqrt{\frac{1}{2}\log N}\right)^2\log N}}+\sqrt{2\log 2} +2\sqrt{\left(\sqrt{\rho}+\sqrt{\frac{1}{2}\log N}\right)^2\log N}+\sqrt{2\log(1/\delta)}\right) +\sqrt{\frac{\log(2/\delta)}{2|C|}}$$

we can distinguish H_1 from H_0. ■

14.8 Proof of Lemma ??

For the nonweighted version, we have

$$\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi}\left(\Psi^T\Psi f(\mathrm{X})\right)_i \;=\; \mathbb{E}_{\mathcal{M}}\left(\sum_{\mathcal{M}_j\in\mathcal{M}} f_{\mathcal{M}_j}(\mathrm{X})\delta_{\mathcal{M}_j=i}\right) \\
\overset{(a)}{=}\; & M\mathbb{E}_\ell\left(f_\ell(\mathrm{X})\delta_{\ell=i}\right) \;=\; M\sum_{\ell=1}^{N} f_\ell(\mathrm{X})\pi_\ell\delta_{\ell=i} \\
=\; & M\pi_i f_i(\mathrm{X}).
\end{aligned}$$

where $\left(\Psi^T\Psi f(\mathrm{X})\right)_i \in \mathbb{R}^K$ is the ith row of $\Psi^T\Psi f(\mathrm{X})$, $f_i(\mathrm{X}) \in \mathbb{R}^K$ is the ith row of $f(\mathrm{X})$, δ_i denotes an indicator function and equality (a) follows from the independent and identically distributed random sampling.

For the reweighted version, we have

$$\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)_i \\
=\; & \mathbb{E}_{\mathcal{M}}\left(\sum_{\mathcal{M}_j\in\mathcal{M}} \mathrm{S}_{\mathcal{M}_j,\mathcal{M}_j}\, f_{\mathcal{M}_j}(\mathrm{X})\delta_{\mathcal{M}_j=i}\right) \\
=\; & M\mathbb{E}_\ell\left(\frac{1}{M\pi_l} f_\ell(\mathrm{X})\delta_{\ell=i}\right) \;=\; M\sum_{\ell=1}^{N}\frac{1}{M\pi_\ell} f_\ell(\mathrm{X})\pi_\ell\delta_{\ell=i} \\
=\; & f_i(\mathrm{X}).
\end{aligned}$$

■

14.9 Proof of Theorem 24

We first split the error into the bias term and the variance term,

$$\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi}\left\|\mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - f(\mathrm{X})\right\|_2^2 \\
=\; & \left\|\mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right) - f(\mathrm{X})\right\|_2^2 \\
& +\mathbb{E}_{\Psi\sim\pi}\left\|\mathrm{S}\,\Psi^T\Psi f(\mathrm{X}) - \mathbb{E}_{\Psi\sim\pi}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)\right\|_2^2,
\end{aligned}$$

where the first term is bias and the second term is variance. Lemma (7) shows that the bias term is zero. So, we only need to bound the variance term.

For each element in the variance term, we have

$$
\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi} \left\| \left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)_i - \mathbb{E}\left(\mathrm{S}\,\Psi^T\Psi f(\mathrm{X})\right)_i \right\|^2 \\
= \; & \mathbb{E}_{\mathcal{M}}\Bigg[\left(\sum_{\mathcal{M}_j\in\mathcal{M}} \mathrm{S}_{\mathcal{M}_j,\mathcal{M}_j} f_{\mathcal{M}_j}(\mathrm{X})\delta_{\mathcal{M}_j=i} - f_i(\mathrm{X}) \right)^T \\
& \left(\sum_{\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{S}_{\mathcal{M}_{j'},\mathcal{M}_{j'}} f_{\mathcal{M}_{j'}}(\mathrm{X})\delta_{\mathcal{M}_{j'}=i} - f_i(\mathrm{X}) \right) \Bigg] \\
= \; & \mathbb{E}_{\mathcal{M}}\Bigg(\sum_{\mathcal{M}_j,\mathcal{M}_{j'}\in\mathcal{M}} \mathrm{S}_{\mathcal{M}_j,\mathcal{M}_j}\,\mathrm{S}_{\mathcal{M}_{j'},\mathcal{M}_{j'}} f_{\mathcal{M}_j}(\mathrm{X})^T f_{\mathcal{M}_{j'}}(\mathrm{X}) \\
& \delta_{\mathcal{M}_j=i}\delta_{\mathcal{M}_{j'}=i} \Bigg) - f_i(\mathrm{X})^T f_i(\mathrm{X}) \\
= \; & M^2\mathbb{E}_\ell\,\mathrm{S}^2_{\ell,\ell}\, f_\ell(\mathrm{X})^T f_\ell(\mathrm{X})\delta_{\ell=i} - f_i(\mathrm{X})^T f_i(\mathrm{X}) \\
= \; & M^2\sum_{\ell=1}^{N} \frac{f_\ell(\mathrm{X})^T f_\ell(\mathrm{X})}{M^2\pi_\ell^2}\pi_\ell\delta_{\ell=i} - f_i(\mathrm{X})^T f_i(\mathrm{X}) \\
= \; & \left(\frac{1}{\pi_i} - 1\right) f_i(\mathrm{X})^T f_i(\mathrm{X}).
\end{aligned}
$$

We finally combine all the elements and obtain (11.4). ■

14.10 Proof of Theorem 25

Proof. Based on Theorem 24, we have

$$
\begin{aligned}
& \mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right) \\
= \; & \mathbb{E}_{\Psi\sim\pi} \max_{\mathrm{X}'_c:\|\mathrm{X}'_c\|_2=c} \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right) f\left(\begin{bmatrix}\mathrm{X}'_c & \mathrm{X}_\mathrm{o}\end{bmatrix}\right)\right\|_F^2 \\
= \; & \mathbb{E}_{\Psi\sim\pi} \max_{\mathrm{X}'_c:\|\mathrm{X}'_c\|_2=c} \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right)\mathrm{F}\begin{bmatrix}\mathrm{X}'_c & \mathrm{X}_\mathrm{o}\end{bmatrix}\right\|_F^2 \\
= \; & \mathbb{E}_{\Psi\sim\pi}\Bigg(\max_{\mathrm{X}'_c:\|\mathrm{X}'_c\|_2=c} \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right)\mathrm{F}\,\mathrm{X}'_c\right\|_F^2 \\
& + \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right)\mathrm{F}\,\mathrm{X}_\mathrm{o}\right\|_F^2 \Bigg) \\
= \; & \mathbb{E}_{\Psi\sim\pi}\Bigg(c^2 \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right)\mathrm{F}\right\|_F^2 \\
& + \left\| \left(\mathrm{S}\,\Psi^T\Psi - \mathbf{I}\right)\mathrm{F}\,\mathrm{X}_\mathrm{o}\right\|_F^2 \Bigg) \\
= \; & c^2\mathrm{Tr}\left(\mathrm{F}\,\mathrm{Q}\,\mathrm{F}^T\right) + \mathrm{Tr}\left(\mathrm{F}\,\mathrm{X}_\mathrm{o}\,\mathrm{Q}(\mathrm{F}\,\mathrm{X}_\mathrm{o})^T\right).
\end{aligned}
$$

□

14.11 Proof of Theorem 26

The optimal resampling strategy is the solution of the following optimization problem,

$$\begin{aligned} &\min_{\pi} \ \mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right) \\ &\text{subject to } \sum_{i=1}^{N}\pi_i = 1, \quad \pi_i \geq 0. \end{aligned} \tag{14.2}$$

The corresponding Lagrange function is

$$\begin{aligned} & L(\pi_i, \lambda, \mu) \\ = \ & \mathbb{E}_{\Psi\sim\pi}\left(D_{f(\mathrm{X})}(\Psi)\right) + \lambda\left(\sum_{i=1}^{N}\pi_i - 1\right) + \sum_{i=1}^{N}\mu_i\pi_i \\ = \ & \sum_{i=1}^{N}\left(\frac{1}{\pi_i} - 1\right)\left\|f_i(\mathrm{X})\right\|_2^2 + \lambda\left(\sum_{i=1}^{N}\pi_i - 1\right) + \sum_{i=1}^{N}\mu_i\pi_i, \end{aligned}$$

where the equality follows from Theorem 24. The derivative to π_i is

$$\frac{\partial L}{\partial \pi_i} \ = \ -\frac{1}{\pi_i^2}\left\|f_i(\mathrm{X})\right\|_2^2 + \lambda + \mu_i. \tag{14.3}$$

By setting its derivative to zero, we have

$$\pi_i = \frac{\left\|f_i(\mathrm{X})\right\|_2}{\sqrt{\lambda + \mu_i}}.$$

Due to the complementary slackness, we have

$$\mu_i\pi_i \ = \ \frac{\mu_i\left\|f_i(\mathrm{X})\right\|_2}{\sqrt{\lambda + \mu_i}} \ = \ 0.$$

Thus, either μ_i or $\left\|f_i(\mathrm{X})\right\|_2$ is zero. In both cases, $\pi_i \propto \left\|f_i(\mathrm{X})\right\|_2$. ■

14.12 Proof of Theorem 27

The optimal resampling strategy is the solution of the following optimization problem,

$$\begin{aligned} &\min_{\pi} \ \mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right) \\ &\text{subject to } \sum_{i=1}^{N}\pi_i = 1, \quad \pi_i \geq 0. \end{aligned} \tag{14.4}$$

The corresponding Lagrange function is

$$\begin{aligned} & L(\pi_i, \lambda, \mu) \\ = \; & \mathbb{E}_{\Psi\sim\pi}\left(D_f(\Psi)\right) + \mu\left(\sum_{i=1}^{N}\pi_i - 1\right) + \sum_{i=1}^{N}\mu_i\pi_i \\ = \; & c^2\sum_{i=1}^{N}\left(\frac{1}{\pi_i}-1\right)\left\|\mathrm{F}_i\right\|_2^2 + \sum_{i=1}^{N}\left(\frac{1}{\pi_i}-1\right)\left\|\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2 \\ & +\mu\left(\sum_{i=1}^{N}\pi_i - 1\right) + \sum_{i=1}^{N}\mu_i\pi_i, \end{aligned}$$

where F_i is the ith row of F and $\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i$ is the ith row of $\mathrm{F}\,\mathrm{X}_{\mathrm{o}}$. The derivative to π_i is

$$\frac{\partial L}{\partial \pi_i} \;=\; -\frac{1}{\pi_i^2}\left(c^2\left\|\mathrm{F}_i\right\|_2^2 + \left\|\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2\right) + \mu + \mu_i.$$

By setting its derivative to zero, we have

$$\pi_i = \frac{\sqrt{c^2\left\|\mathrm{F}_i\right\|_2^2 + \left\|\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2}}{\sqrt{\mu+\mu_i}}.$$

Due to the complementary slackness, we have

$$\mu_i\pi_i \;=\; \frac{\mu_i\sqrt{c^2\left\|\mathrm{F}_i\right\|_2^2 + \left\|\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2}}{\sqrt{\mu+\mu_i}} \;=\; 0.$$

Thus, either μ_i or $\left\|f_i(\mathrm{X})\right\|_2$ is zero. In both cases, $\pi_i \propto \sqrt{c^2\left\|\mathrm{F}_i\right\|_2^2 + \left\|\left(\mathrm{F}\,\mathrm{X}_{\mathrm{o}}\right)_i\right\|_2^2}$. ■

14.13 Proof of Theorem 28

We first show the rotational invariance. Let X be the 3D coordinates of an original point cloud and $\mathrm{R} \in \mathbb{R}^{3\times 3}$ be a rotation matrix. The point cloud after rotating is $\mathrm{X}\,\mathrm{R}$. The local variation of $\mathrm{X}\,\mathrm{R}$ is

$$\begin{aligned} f_i(\mathrm{X}\,\mathrm{R}) &= \left\|\left(h(\mathrm{A})\,\mathrm{X}\,\mathrm{R}\right)_i\right\|_2^2 \\ &= \left\|\left(h(\mathrm{A})\right)_i\mathrm{X}\,\mathrm{R}\right\|_2^2 \\ &= \left(h(\mathrm{A})\right)_i\mathrm{X}\,\mathrm{R}\,\mathrm{R}^T\,\mathrm{X}^T\left(h(\mathrm{A})\right)_i^T \\ &\overset{(a)}{=} \left(h(\mathrm{A})\right)_i\mathrm{X}\,\mathrm{X}^T\left(h(\mathrm{A})\right)_i^T \\ &= \left\|\left(h(\mathrm{A})\,\mathrm{X}\right)_i\right\|_2^2 \;=\; f_i(\mathrm{X}), \end{aligned}$$

where $\left(h(\mathrm{A})\right)_i$ is the ith row of $h(\mathrm{A})$ and (a) follows from any rotation matrix R is orthonormal.

We next show the shift variance. Let $\mathbf{a} \in \mathbb{R}^3$ be the shift and the point cloud after shifting is $\mathrm{X} + \mathbf{1}\mathbf{a}^T$. The local variation of $\mathrm{X} + \mathbf{1}\mathbf{a}^T$ is

$$\begin{aligned} f_i(\mathrm{X} + \mathbf{1}\mathbf{a}^T) &= \left\| \left(h(\mathrm{A}) \left(\mathrm{X} + \mathbf{1}\mathbf{a}^T \right) \right)_i \right\|_2^2 \\ &= \left\| \left(h(\mathrm{A})\, \mathrm{X} \right)_i + \left(h(\mathrm{A}) \mathbf{1}\mathbf{a}^T \right)_i \right\|_2^2 \end{aligned}$$

Thus, $f_i(\mathrm{X} + \mathbf{1}\mathbf{a}^T) = f_i(\mathrm{X})$ only when $h(\mathrm{A})\mathbf{1} = \mathbf{0} \in \mathbb{R}^N$. ■

Bibliography

[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and 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.*, vol. 30, pp. 83–98, May 2013.

[2] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[3] S. Zhang and M. A. Karim, "A new impulse detector for switching median filters," *IEEE Signal Process. Lett.*, vol. 9, no. 12, pp. 360–363, Nov. 2002.

[4] Michèle and Igor V. Nikiforov, "Detection of abrupt changes: Theory and application," *Automatica*, vol. 32, no. 8, pp. 1235–1236, 1996.

[5] J. D. Haupt, R. M. Castro, and R. D. Nowak, "Distilled sensing: selective sampling for sparse signal recovery," in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS*, Clearwater Beach, Florida, April 2009, pp. 216–223.

[6] V. Cevher, P. Indyk, C. Hegde, and R. G. Baraniuk, "Recovery of clustered sparse signals from compressive measurements," in *Proc. Sampling Theory Appl.*, Marseille, May 2009.

[7] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, June 2005.

[8] F. Buggenthin, C. Marr, M. Schwarzfischer, P. S. Hoppe, O. Hilsenbeck, T. Schroeder, and F. J. Theis, "An automatic method for robust and fast cell detection in bright field images from high-throughput microscopy," *BMC Bioinformatics*, vol. 14, pp. 297, 2013.

[9] G. L. Turin, "An introduction to matched filters," *IRE Trans. Information Theory*, vol. 6, no. 3, pp. 311–329, 1960.

[10] A. Mahalanobis, B. V. K. V. Kumar, and D. Casasent, "Minimum average correlation energy filters," *Appl. Opt.*, vol. 26, pp. 3633–3640, 1987.

[11] G. Birkhoff and S. MacLane, *Introduction to Graph Theory*, Longman, 1953.

[12] F. Harary, *Graph theory*, Addison-Wesley, Reading, MA, 1969.

[13] F. Chung and L. Lu, *Complex Graphs and Networks (Cbms Regional Conference Series in Mathematics)*, American Mathematical Society, Boston, MA, USA, 2006.

[14] D. Eppstein, "Subgraph isomorphism in planar graphs and related problems," *J. Graph*

Algorithms Appl., vol. 3, no. 3, 1999.

[15] D. Brélaz, "New methods to color the vertices of a graph," *Commun. ACM*, vol. 22, no. 4, pp. 251–256, Apr. 1979.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press and McGrawHill, London, UK, 2 edition, 2001.

[17] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[18] A. Schrijver, "On the history of the transportation and maximum flow problems," *Math. Program.*, vol. 91, no. 3, pp. 437–445, 2002.

[19] I. Herman, G. Melancon, and M. S. Marshall, "Graph visualization and navigation in information visualization: A survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 6, no. 1, pp. 24–43, 2000.

[20] F. R. K. Chung, *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*, Am. Math. Soc., 1996.

[21] D. A. Spielman, "Spectral graph theory and its applications," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS, 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, 2007, pp. 29–38.

[22] U. V. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[23] Y. Koren, "Drawing graphs by eigenvectors: Theory and practice," *Comput. Math. Appl.*, vol. 49, no. 11-12, pp. 1867–1888, June 2005.

[24] D. A. Spielman and N. Srivastava, "Graph sparsification by effective resistances," *SIAM J. Comput.*, vol. 40, no. 6, pp. 1913–1926, 2011.

[25] D. A. Spielman, "Algorithms, graph theory, and the solution of laplacian linear equations," in *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, 2012, pp. 24–26.

[26] M. Newman, *Networks: An Introduction*, Oxford University Press, 2010.

[27] R. Pastor-Satorras, C. Castellano, P. V. Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *Rev. Mod. Phys.*, vol. 87, Aug. 2015.

[28] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci.*, vol. 99, pp. 7821–7826, 2002.

[29] H. Schmidt, G. Petkov, M. P. Richardson, and J. R. Terry, "Dynamics on networks: The role of local dynamics and global networks on the emergence of hypersynchronous neural activity," *PLoS Computational Biology*, vol. 10, no. 11, 2014.

[30] D. Chakrabarti and C. Faloutsos, *Graph Mining: Laws, Tools, and Case Studies*, Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers, 2012.

[31] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Min. Knowl. Discov.*, vol. 29, no. 3, pp. 626–688, May 2015.

[32] Y. Tian, R. A. Hankins, and J. M. Patel, "Efficient aggregation for graph summarization," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Vancouver, Canada, June 2008, pp. 567–580.

[33] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "FRAUDAR: bounding graph fraud in the face of camouflage," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 895–904.

[34] J. J. McAuley and J. Leskovec, "Learning to discover social circles in ego networks," in *Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, Dec. 2012, pp. 548–556.

[35] J. Leskovec and R. Sosic, "SNAP: A general-purpose network analysis and graph-mining library," *ACM TIST*, vol. 8, no. 1, pp. 1, 2016.

[36] A. Sandryhaila and J. M. F. Moura, "Big data processing with signal processing on graphs," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, 2014.

[37] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, pp. 129–150, Mar. 2011.

[38] S.l K. Narang, G. Shen, and A. Ortega, "Unidirectional graph-based wavelet transforms for efficient data gathering in sensor networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Dallas, TX, Mar. 2010, pp. 2902–2905.

[39] E. Pavez, H. E. Egilmez, Y. Wang, and A. Ortega, "GTT: graph template transforms with applications to image coding," in *2015 Picture Coding Symposium, PCS 2015, Cairns, Australia, May 31 - June 3, 2015*, 2015, pp. 199–203.

[40] D. I. Shuman, M. J. Faraji, , and P. Vandergheynst, "A multiscale pyramid transform for graph signals," *IEEE Trans. Signal Process.*, vol. 64, pp. 2119–2134, Apr. 2016.

[41] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Kyoto, Japan, Mar. 2012, pp. 3921 – 3924.

[42] D. Thanou, D. I. Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Trans. Signal Process.*, vol. 62, pp. 3849–3862, June 2014.

[43] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," *IEEE Trans. Signal Processing*, vol. 63, no. 16, pp. 4223–4235, 2015.

[44] A. Agaskar and Y. M. Lu, "A spectral graph uncertainty principle," *IEEE Trans. Inf. Theory*, vol. 59, no. 7, pp. 4338–4356, July 2013.

[45] M. Tsitsvero, S. Barbarossa, and P. D. Lorenzo, "Signals on graphs: Uncertainty principle and sampling," *IEEE Trans. Signal Process.*, vol. 64, pp. 4845 – 4860, 2015.

[46] N. Perraudin, B. Ricaud, D. Shuman, and P. Vandergheynst, "Global and local uncertainty principles for signals on graphs," *CoRR*, vol. arXiv:1603.03030, 2016.

[47] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, pp. 2786–2799, June 2012.

[48] S. K. Narang and Antonio Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4673–4685, Oct. 2013.

[49] V. N. Ekambaram, G. C. Fanti, B. Ayazifar, and K. Ramchandran, "Spline-like wavelet filterbanks for multiresolution analysis of graph-structured data," *IEEE Trans. Signal and Information Processing over Networks*, vol. 1, no. 4, pp. 268–278, 2015.

[50] N. Tremblay and P. Borgnat, "Subgraph-based filterbanks for graph signals," *IEEE Trans. Signal Processing*, vol. 64, no. 15, pp. 3827–3840, 2016.

[51] J. Zeng, G. Cheung, and A. Ortega, "Bipartite subgraph decomposition for critically sampled wavelet filterbanks on arbitrary graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, Mar. 2016, pp. 6210–6214.

[52] N. Tremblay and P. Borgnat, "Graph wavelets for multiscale community mining," *IEEE Trans. Signal Process.*, vol. 62, pp. 5227–5239, Oct. 2014.

[53] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds," *IEEE Trans. Signal Process.*, vol. 62, no. 4, pp. 905–918, Feb. 2014.

[54] P.-Y. Chen and A.O. Hero, "Local Fiedler vector centrality for detection of deep and overlapping communities in networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, 2014, pp. 1120–1124.

[55] B. Girault, "Stationary graph signals using an isometric graph translation," in *23rd European Signal Processing Conference, EUSIPCO 2015*, Nice, France, Aug. 2015, pp. 1516–1520.

[56] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *arXiv preprint arXiv:1603.04667*, Jan. 2016.

[57] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *arXiv preprint arXiv:1603.04667*, Mar. 2016.

[58] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal denoising on graphs via graph filtering," in *Proc. IEEE Glob. Conf. Signal Information Process.*, Atlanta, GA, Dec. 2014, pp. 872–876.

[59] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Trans. Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 137–148, 2016.

[60] A. Anis, A. Gadde, and A. Ortega, "Towards a sampling theorem for signals on arbitrary graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, May 2014, pp. 3864–3868.

[61] A. Gadde, A. Anis, and A. Ortega, "Active semi-supervised learning using sampling theory for graph signals," in *KDD*, New York, New York, USA, 2014, pp. 492–501.

[62] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3775–3789, July 2016.

[63] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Dec 2015.

[64] X. Wang, J. Chen, and Y. Gu, "Local measurement and reconstruction for noisy graph signals," *Signal Processing*, vol. 129, pp. 119–129, 2016.

[65] S. K. Narang, Akshay Gadde, and Antonio Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Vancouver, May 2013, pp. 5445–5449.

[66] X. Wang, P. Liu, and Y. Gu, "Local-set-based graph signal reconstruction," *IEEE Trans. Signal Process.*, vol. 63, no. 9, May 2015.

[67] X. Wang, M. Wang, and Y. Gu, "A distributed tracking algorithm for reconstruction of graph signals," *IEEE Journal of Selected Topics on Signal Processing*, vol. 9, no. 4, June 2015.

[68] S. Segarra, A. G. Marques, G. Leus, and A. Ribeiro, "Reconstruction of graph signals through percolation from seeding nodes," *IEEE Trans. Signal Process.*, vol. 64, pp. 4363–4378, Aug. 2016.

[69] M S. Kotzagiannidis and P. L. Dragotti, "Sampling and reconstruction of sparse signals on circulant graphs-an introduction to graph-fri," *arXiv preprint arXiv:1606.08085*, 2016.

[70] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, pp. 6160–6173, Dec. 2016.

[71] J. Mei and J. M. F. Moura, "Signal processing on graphs: Estimating the structure of a graph," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Brisbane, Apr. 2015, pp. 5495–5499.

[72] J. Mei and J. M. F. Moura, "Signal processing on graphs: Modeling (causal) relations in big data," *CoRR*, vol. abs/1503.00173, 2015.

[73] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *CoRR*, vol. abs/1608.03008, 2016.

[74] E. Pavez and A. Ortega, "Generalized laplacian precision matrix estimation for graph signal processing," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, Shanghai, China, Mar. 2016, pp. 6350–6354.

[75] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[76] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, June 2014.

[77] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Appl. Comput. Harmon. Anal.*, pp. 53–94, July 2006.

[78] A. Krishnamurthy J. Sharpnack and A. Singh, "Detecting activations over graphs using spanning tree wavelet bases," in *AISTATS*, Scottsdale, AZ, Apr. 2013.

[79] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Appl. Comput. Harmon. Anal.*, February 2015, To appear.

[80] M. Gavish, B. Nadler, and R. R. Coifman, "Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning," in *Proc. Int. Conf. Mach. Learn.*, Haifa, Israel, June 2010, pp. 367–374.

[81] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *Proc. IEEE INFOCOM*, Mar. 2003, vol. 3, pp. 1848–1857.

[82] M. Belkin, P. Niyogi, and P. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples.," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, 2006.

[83] M. Püschel and J. M. F. Moura, "Algebraic signal processing theory: Foundation and 1-D time," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3572–3585, Aug. 2008.

[84] M. Vetterli, J. Kovačević, and V. K. Goyal, *Foundations of Signal Processing*, Cambridge University Press, Cambridge, 2014, http://www.fourierandwavelets.org/.

[85] H. Triebel, *Theory of function spaces. II*, Monographs in mathematics. Birkhuser Verlag, Basel, Boston, Berlin, 1992.

[86] D. L. Donoho, "Wedgelets: Nearly minimax estimation of edges," *Ann. Stat.*, vol. 27, no. 3, pp. 859–897, June 1999.

[87] R. R. Coifman and S. Lafon, "Diffusion maps," *Appl. Comput. Harmon. Anal.*, vol. 21, pp. 5–30, July 2006.

[88] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, New York, NY, third edition, 2009.

[89] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neur. Comput.*, vol. 13, pp. 1373–1396, 2003.

[90] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Trans. Signal Process.*, vol. 63, pp. 6510 – 6523, Aug. 2015.

[91] D. L. Donoho and P. B. Stark, "Uncertainty principles and signal recovery," *SIAM J. Appl. Math*, vol. 49, no. 3, pp. 906–931, June 1989.

[92] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Random versus experimentally designed sampling," in *Proc. Sampling Theory Appl.*, Washington, DC, May 2015, pp. 337–341.

[93] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal representations on graphs: Tools and applications," *http://arXiv:1512.05406*, 2015.

[94] M. Cucuringu and M. W. Mahoney, "Localization on low-order eigenvectors of data matrices," *Technical Report, Preprint: arXiv:1109.1355*, 2011.

[95] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Trans. Knowledge Discovery from Data*, vol. 1, no. 2, pp. 4339–4347, Mar. 2007.

[96] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. Asilomar Conf. Signal, Syst. Comput.*, Pacific Grove, CA, Nov. 1993, vol. 1, pp. 40–44.

[97] D. Gleich, ," The MatlabBGL Matlab library, http://www.cs.purdue.edu/homes/dgleich/packages/matlab bgl/index.html.

[98] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal recovery on graphs: Variation minimization," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4609–4624, Sept. 2015.

[99] T. Ji, S. Chen, R. Varma, and J. Kovačević, "Energy-efficient route planning for autonomous aerial vehicles based on graph signal recovery," in *Proc. Allerton Conf. on Commun., Control and Comput.*, Allerton, IL, 2015.

[100] M. Vetterli and T. Kalker, "Matching pursuit for compression and application to motion compensated video coding," in *Proc. IEEE Int. Conf. Image Process.*, Austin, TX, Nov. 1994, vol. 1, pp. 725–729.

[101] E. J. Candès and D. L. Donoho, "New tight frames of curvelets and optimal representation of objects with piecewise C2 singularities," *Commun. Pure Appl. Math.*, vol. 57, pp. 219–266, 2004.

[102] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, "Rate-distortion optimized tree-structured compression algorithms for piecewise polynomial images," *IEEE Trans. Image Processing*, vol. 14, no. 3, pp. 343–359, 2005.

[103] J-L Starck, M. Elad, and D. L. Donoho, "Image decomposition via the combination of sparse representations and a variational approach," *IEEE Trans. Image Processing*, vol. 14, no. 10, pp. 1570–1582, 2005.

[104] J. Kovačević and M. Püschel, "Algebraic signal processing theory: Sampling for infinite and finite 1-D space," *IEEE Trans. Signal Process.*, vol. 58, no. 1, pp. 242–257, Jan. 2010.

[105] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Signal recovery on graphs: Fundamental limits of sampling strategies," *IEEE Trans. Signal and Inf. Process. over Networks sp. iss. Inference and Learning over Networks*, 2016, To appear.

[106] X. Zhu, "Semi-supervised learning literature survey," Tech. Rep. 1530, University Wisconsin-Madison, 2005.

[107] M. Unser, "Sampling – 50 years after Shannon," *Proc. IEEE*, vol. 88, no. 4, pp. 569–587, Apr. 2000.

[108] S. Chen, A. Sandryhaila, and J. Kovačević, "Sampling theory for graph signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Brisbane, Apr. 2015, pp. 3392 – 3396.

[109] I. Z. Pesenson, "Sampling in Paley-Wiener spaces on combinatorial graphs," *Trans. Am. Math. Soc.*, vol. 360, no. 10, pp. 5603–5627, May 2008.

[110] H. Avron and C. Boutsidis, "Faster subset selection for matrices and applications," *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 4, pp. 1464–1499, 2013.

[111] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–

1306, Apr. 2006.

[112] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.

[113] E. J. Candès, Y. Eldar, D. Needell, and P. Randall, "Compressed sensing with coherent and redundant dictionaries," *Applied and Computational Harmonic Analysis*, vol. 31, pp. 59–73, 2010.

[114] H. Q. Nguyen and M. N. Do, "Downsampling of signals on graphs via maximum spanning trees," *IEEE Trans. Signal Process.*, vol. 63, no. 1, pp. 182–191, Jan. 2015.

[115] M. Vetterli, "A theory of multirate filter banks," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 35, no. 3, pp. 356–372, Mar. 1987.

[116] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. election: Divided they blog," in *Proc. LinkKDD*, 2005, pp. 36–43.

[117] C. M. Bishop, *Pattern Recognition and Machine Learning*, Information Science and Statistics. Springer, 2006.

[118] S. Chen, A. Sandryhaila, G. Lederman, Z. Wang, J. M. F. Moura, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević, "Signal inpainting on graphs via total variation minimization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, May 2014, pp. 8267–8271.

[119] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Intelligent Signal Processing*. 2001, pp. 306–351, IEEE Press.

[120] J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.

[121] R. Castro, R. Willett, and R. Nowak, "Faster rates in regression via active learning," in *Proc. Neural Information Process. Syst.*, Vancouver, Dec. 2005.

[122] R. Castro and R. Nowak, "Minimax bounds for active learning," *IEEE Trans. Inf. Theory*, vol. 54, no. 5, pp. 2339–2353, July 2008.

[123] A. P. Korostelev and A. B. Tsybakov, *Minimax theory of image reconstruction*, Lecture Notes on Statistics. Springer, 1993.

[124] I. M. Johnstone, *Minimax Bayes, Asymptotic Minimax and Sparse Wavelet Priors*, Statistical Decision Theory and Related Topics V. Springer, 1994.

[125] A. Sakiyama, Y. Tanaka, T. Tanaka, and A. Ortega, "Efficient sensor position selection using graph signal sampling theory," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, China, Mar. 2016, pp. 6225 – 6229.

[126] N. Tremblay, G. Puy, R. Gribonval, and P. Vandergheynst, "Compressive spectral clustering," *arXiv:1602.02018*, 2016.

[127] A. P. Korostelev and A. B. Tsybakov, *Minimax Theory of Image Reconstruction*, Lecture Notes on Statistics. Springer, 1993.

[128] P. Ma, M. W. Mahoney, and B. Yu, "A statistical perspective on algorithmic leveraging," *J. Mach. Learn. Res.*, vol. 16, pp. 861–911, 2015.

[129] A. Sandryhaila, J. Kovačević, and M. Püschel, "Algebraic signal processing theory: 1-D nearest-neighbor models," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2247–2259, May 2012.

[130] J. Dall and M. Christensen, "Random geometric graphs," *Phys. Rev. E*, vol. 66, July 2002.

[131] M. Jackson, *Social and Economic Networks*, Princeton University Press, 2008.

[132] K. I. Goh, B. Kahng, and D. Kim, "Spectra and eigenvectors of scale-free networks," *Phys Rev E*, vol. 64, no. 5, Nov. 2001.

[133] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. Int. Conf. Mach. Learn. Workshop on Continuum from Labeled to Unlabeled Data in Mach. Learn. Data Mining*, Washington, DC, 2003, pp. 58–65.

[134] Q. Gu and J. Han, "Towards active learning on graphs: An error bound minimization approach," in *Proc. IEEE Int. Conf. Data Mining*, Brussels, Dec. 2012, pp. 882 – 887.

[135] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, pp. 490–530, July 2005.

[136] L. I. Rudin, Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, no. 1–4, pp. 259–268, Nov. 1992.

[137] T. F. Chan, S. Osher, and J. Shen, "The digital TV filter and nonlinear denoising," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 231–241, Feb. 2001.

[138] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, no. 1–2, pp. 89–97, Jan. 2004.

[139] E. J. Candés, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Commun. Pure Appl. Math.*, vol. 59, pp. 1207–1223, Aug. 2006.

[140] A. Elmoataz, Olivier Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1047–1060, July 2008.

[141] T. F. Chan and J. Shen, "Variational image inpainting," *Commun. Pure Appl. Math.*, vol. 58, pp. 579–619, Feb. 2005.

[142] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. Image Process.*, vol. 17, pp. 53–69, Jan. 2008.

[143] E. J. Candés, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *J. ACM*, vol. 58, May 2011.

[144] J. Wright, A. Ganesh., S. Rao, Y. Peng, and Yi Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization," in *Proc. Neural Information Process. Syst.*, Dec. 2009, pp. 2080–2088.

[145] E. J. Candés and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 2, pp. 717–772, Dec. 2009.

[146] E. J. Candés and Y. Plan, "Matrix completion with noise," *Proc. IEEE*, vol. 98, pp.

925–936, June 2010.

[147] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from noisy entries," *jmach-learn*, vol. 11, pp. 2057–2078, July 2010.

[148] M. Mardani, G. Mateos, and G. B. Giannakis, "Decentralized sparsity-regularized rank minimization: Algorithms and applications," *IEEE Trans. Signal Process.*, vol. 61, pp. 5374 – 5388, Nov. 2013.

[149] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. ICML Workshop Stat. Rel. Learn.*, 2003, pp. 912–919.

[150] D. Zhou and B. Scholkopf, "A regularization framework for learning from graph data," in *Proc. ICML Workshop Stat. Rel. Learn.*, 2004, pp. 132–137.

[151] D. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," *Proc. Nat. Acad. Sci.*, vol. 100, no. 5, pp. 2197–2202, Mar. 2003.

[152] E. J. Candès and Y. Plan, "Near-ideal model selection by ℓ^1 minimization," *Ann. Stat.*, vol. 37, no. 5A, pp. 2145–2177, 2009.

[153] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[154] S. Boyd and L. Vandenberghe, *Convex Optimization*, vol. 17, Cambridge University Press, New York, NY, 2004.

[155] F. Cerda, J. H. Garrett, J. Bielak, P. Rizzo, J. A. Barrera, Z. Zhang, S. Chen, M. T. McCann, and J. Kovačević, "Indirect structural health monitoring in bridges: Scale experiments," in *Proc. Int. Conf. Bridge Maint., Safety Manag.*, Lago di Como, July 2012, pp. 346–353.

[156] F. Cerda, S. Chen, J. Bielak, J. H. Garrett, P. Rizzo, and J. Kovačević, "Indirect structural health monitoring of a simplified laboratory-scale bridge model," *Int. J. Smart Struct. Syst., sp. iss. Challenge on bridge health monitoring utilizing vehicle-induced vibrations*, vol. 13, no. 5, pp. 849–868, May 2014.

[157] G. Lederman, Z. Wang, J. Bielak, H. Noh, J. H. Garrett, S. Chen, J. Kovačević, F. Cerda, and P. Rizzo, "Damage quantification and localization algorithms for indirect SHM of bridges," in *Proc. Int. Conf. Bridge Maint., Safety Manag.*, Shanghai, July 2014, pp. 640–647.

[158] Jester jokes, ," http://eigentaste.berkeley.edu/user/index.php.

[159] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Adaptive graph filtering: Multiresolution classification on graphs," in *Proc. IEEE Glob. Conf. Signal Information Process.*, Austin, TX, Dec. 2013, pp. 427–430.

[160] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *J. Mach. Learn. Res.*, vol. 11, pp. 2287–2322, Aug. 2010.

[161] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. Opt.*, vol. 20, no. 4, pp. 1956–1982, Mar. 2010.

[162] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *SIAM Conf. Data Mining*, 2006, pp. 549–553.

[163] Q. Gu, J. Zhou, and C. Ding, "Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs," in *SIAM Conf. Data Mining*, 2010, pp. 199–210.

[164] R. H. Keshavan, A. Montanari, and S. Oh, "Low-rank matrix completion with noisy observations: A quantitative comparison," in *Proc. Allerton Conf. on Commun., Control and Comput.*, 2009, pp. 1216–1222.

[165] A. Kuruvilla, N. Shaikh, A. Hoberman, and J. Kovačević, "Automated diagnosis of otitis media: A vocabulary and grammar," *Int. J. Biomed. Imag., sp. iss. Computer Vis. Image Process. for Computer-Aided Diagnosis*, Aug. 2013.

[166] S. R. Cholleti, S. A. Goldman, A. Blum, David G. Politte, S Don, K Smith, and F Prior, "Veritas: Combining expert opinions without labeled data.," *International Journal on Artificial Intelligence Tools*, vol. 18, no. 5, pp. 633–651, 2009.

[167] Committee on the Analysis of Massive Data; Committee on Applied, Theoretical Statistics; Board on Mathematical Sciences, Their Applications; Division on Engineering, and Physical Sciences; National Research Council, *Frontiers in Massive Data Analysis*, National Academies Press, 2013.

[168] D. O. North, "An analysis of the factors which determine signal/noise discrimination in pulsed-carrier systems," *Proceedings of the IEEE*, vol. 51, pp. 10161027, July 1963.

[169] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, 2002.

[170] Y. Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 7307, pp. 7821–7826, Aug. 2010.

[171] E. Abbe, A. S. Bandeira, and G. Hall, "Exact recovery in the stochastic block model," *IEEE Trans. Information Theory*, vol. 62, no. 1, pp. 471–487, 2016.

[172] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, pp. 75–174, Feb. 2010.

[173] E. Arias-Castro, E. J. Candès, and A. Durand, "Detection of an anomalous cluster in a network," *The Annals of Statistics*, vol. 39, no. 1, pp. 278304, 2011.

[174] S. Zou, Y. Liang, and V. H. Poor, "Nonparametric detection of an anomalous disk over a two-dimensional lattice network," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Shanghai, 2016, pp. 2394–2398.

[175] C. Hu, L. Cheng, J. Sepulcre, G. E. Fakhri, Y. M. Lu, and Q. Li, "Matched signal detection on graphs: Theory and application to brain network classification," in *Proc. 23rd International Conference on Information Processing in Medical Imaging*, Asilomara, CA, 2013.

[176] A. Krishnamurthy, "Minimaxity in structured normal means inference," *http://arxiv.org/pdf/1506.07902*, 2015.

[177] J. Sharpnack, A. Rinaldo, and A. Singh, "Changepoint detection over graphs with the spectral scan statistic," in *Artifical Intelligence and Statistics (AISTATS)*, 2013.

[178] J. Sharpnack, A. Krishnamurthy, and A. Singh, "Near-optimal anomaly detection in graphs using lovasz extended scan statistic," in *Neural Information Processing Systems (NIPS)*, 2013.

[179] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1222–1239, Nov. 2001.

[180] J. Yang and J. Leskovec, "Overlapping community detection at scale: a nonnegative matrix factorization approach," in *WSDM*, Rome, Italy, Feb. 2013, pp. 587–596.

[181] J. Yang, J. J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *ICDM*, Dallas, TX, Dec. 2013, pp. 1151–1156.

[182] J. Zhu Y. Zhang, E. Levina, "Community detection in networks with node features," *http://arXiv:1509.01173*, 2015.

[183] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice Hall, Englewood Cliffs, NJ, 1995, http://waveletsandsubbandcoding.org/.

[184] M. Ledoux, *The concentration of measure phenomenon*, vol. 89, American Mathematical Soc., 2001.

[185] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[186] Y-L Zhang and F. Cao, "Fine particulate matter (pm2.5) in china at a city level," *Nature, Scientific Reports*, 2015.

[187] "Learn about air," http://www.epa.gov/learn-issues/learn-about-air.

[188] "IEEE Xplore digital library," http://ieeexplore.ieee.org/search/advsearch.jsp.

[189] J. L. Myers, A. D. Well, and R. F. Lorch Jr, *Research Design and Statistical Analysis (3nd ed.)*, Routledge, 2010.

[190] C. Faloutsos, "Large graph mining: patterns, cascades, fraud detection, and algorithms," in *Proceedings of the 23rd international conference on World wide web*. ACM, 2014, pp. 1–2.

[191] A. Beutel, L. Akoglu, and C. Faloutsos, "Fraud detection through graph-based user behavior modeling," in *Proc. ACM Conf. Comput. Commun. Security*, 2015, pp. 1696–1697.

[192] S. Chen, Y. Yang, S. Zong, A. Singh, and J. Kovačević, "Detecting structure-correlated attributes on graphs," *IEEE Trans. Signal Process.*, 2016, Submitted.

[193] C. Chevalier and I. Safro, "Comparison of coarsening schemes for multilevel graph partitioning," in *Learning and Intelligent Optimization, Third International Conference*, Trento, Italy, Jan. 2009, pp. 191–205.

[194] P. Liu, X. Wang, and Y. Gu, "Coarsening graph signal with spectral invariance," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Florence, May 2014, pp. 1075–1079.

[195] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, John Wiley & Sons, Englewood Cliffs, NJ, 2001.

[196] Y-X Wang, J. Sharpnack, A. Smola, and R. J. Tibshirani, "Trend filtering on graphs," in *AISTATS*, San Diego, CA, May 2015.

[197] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, CN, May 2011.

[198] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *ACM Trans. Graph. Proceedings of ACM SIGGRAPH*, vol. 26, pp. 71–78, Jul. 1992.

[199] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, St. Paul, MN, USA, May 2012, pp. 778–785.

[200] C. Zhang, D. Florencio, and C. T. Loop, "Point cloud attribute compression with graph transform," in *Proc. IEEE Int. Conf. Image Process.*, Paris, France, Oct. 2014, pp. 2066–2070.

[201] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based compression of dynamic 3d point cloud sequences," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1765–1778, Feb. 2016.

[202] A. Anis, P. A. Chou, and A. Ortega, "Compression of dynamic 3d point clouds using subdivisional meshes and graph wavelet transforms," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, 2016, pp. 6360–6364.

[203] P. Oesterling, C. Heine, H. Jänicke, G. Scheuermann, and G. Heyer, "Visualization of high-dimensional point clouds using their density distribution's topology," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 11, pp. 1547–1559, 2011.

[204] S. Pecnik, D. Mongus, and B. Zalik, "Evaluation of optimized visualization of lidar point clouds, based on visual perception," in *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data (HCI-KDD)*, Maribor, Slovenia, Jul. 2013, pp. 366–385.

[205] B. F. Gregorski, B. Hamann, and K. I. Joy, "Reconstruction of b-spline surfaces from scattered data points," in *Computer Graphics International*, Geneva, Switzerland, Jun. 2000, pp. 163–170.

[206] A. Golovinskiy, V. G. Kim, and T. A. Funkhouser, "Shape-based recognition of 3d point clouds in urban environments," in *Proc. IEEE Int. Conf. Comput. Vis.*, Kyoto, Japan, Sep. 2009, pp. 2154–2161.

[207] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Trans. on Visualization and Computer Graphics*, vol. 9, pp. 3–15, Jan. 2003.

[208] F. Ryden, S. N. Kosari, and H. J. Chizeck, "Proxy method for fast haptic rendering from time varying point clouds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, San Francisco, CA, Sep. 2011, pp. 2614–2619.

[209] I. Guskov, W. Sweldens, and P. Schröder, "Multiresolution signal processing for meshes," *Comput. Graphics Proc.*, pp. 325–334, 1999.

[210] M. Wand, A. Berner, M. Bokeloh, A. Fleck, M. Hoffmann, P. Jenke, B. Maier, D. Staneker, and A. Schilling, "Interactive editing of large point clouds," in *Symposium on Point Based Graphics*, Prague, Czech Republic, 2007, pp. 37–45.

[211] K. Demarsin, D. Vanderstraeten, T. Volodine, and D. Roose, "Detection of closed sharp feature lines in point clouds for reverse engineering applications," in *Geometric Modeling and Processing*, Pittsburgh, PA, 2006, pp. 571–577.

[212] C. Weber, S. Hahmann, and H. Hagen, "Sharp feature detection in point clouds," in *Shape Modeling International Conference*, Aix en Provence, France, Jun. 2010, pp. 175–186.

[213] J. Daniels, L. K. Ha, T. Ochotta, and C. T. Silva, "Robust smooth feature extraction from point clouds," in *Shape Modeling and Applications, 2007. SMI'07. IEEE International Conference on*, Lyon, France, June 2007, IEEE, pp. 123–136.

[214] C. Choi, A. J. Trevor, and H.I. Christensen, "Rgb-d edge detection and edge-based registration," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, Nov. 2013, IEEE, pp. 1568–1575.

[215] C. Feng, Y. Taguchi, and V. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *Proceedings of IEEE International Conference on Robotics and Automation*, Hong Kong, May 2014, pp. 6218–6225.

[216] M. Shahzad and X. Zhu, "Robust reconstruction of building facades for large areas using spaceborne tomosar point clouds," *IEEE Trans. Geoscience and Remote Sensing*, vol. 53, no. 2, pp. 752–769, 2015.

[217] W. Luo and H. Zhang, "Visual analysis of large-scale lidar point clouds," in *IEEE International Conference on Big Data*, Santa Clara, CA, Nov. 2015, pp. 2487–2492.

[218] J. Ryde and H. Hu, "3D mapping with multi-resolution occupied voxel lists," *Autonomous Robots*, pp. 169–185, Apr. 2010.

[219] C. T. Loop, C. Zhang, and Z. Zhang, "Real-time high-resolution sparse voxelization with application to image-based modeling," in *Proceedings of the 5th High-Performance Graphics 2013 Conference HPG '13*, Anaheim, California, USA, July 2013, pp. 73–80.

[220] J. Peng and C.-C. Jay Kuo, "Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition," *ACM Trans. Graph. Proceedings of ACM SIGGRAPH*, vol. 24, no. 3, pp. 609–616, Jul. 2005.

[221] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, pp. 189–206, Apr. 2013.

[222] B. Eckart and A. Kelly, "Rem-seg: A robust em algorithm for parallel segmentation and registration of point clouds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, Jun. 2013, pp. 4355–4362.

[223] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, "Accelerated generative models for 3D point cloud data," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, Las Vegas, Nevada, Jun. 2016.

[224] J. Peng, C-S. Kim, and C.-C. Jay Kuo, "Technologies for 3D mesh compression: A

survey," *J. Vis. Comun. Image Represent.*, vol. 16, no. 6, pp. 688–733, Dec. 2005.

[225] P. Alliez and C. Gotsman, "Recent advances in compression of 3D meshes," in *In Advances in Multiresolution for Geometric Modelling*. 2003, pp. 3–26, Springer-Verlag.

[226] A. Maglo, G. Lavoué, F. Dupont, and C. Hudelot, "3D mesh compression: Survey, comparisons, and emerging trends," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 44:1–44:41, Feb. 2015.

[227] T. Hackel, J. D. Wegner, and K. Schindler, "Contour detection in unstructured 3d point clouds," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recogn.*, Las Vegas, Nevada, Jun. 2016.

[228] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, pp. 1832 – 1843, 2015.

[229] Y. Taguchi, Y-D Jian, S. Ramalingam, and C. Feng, "Point-plane slam for hand-held 3d sensors," in *Proceedings of IEEE International Conference on Robotics and Automation*, Karlsruhe, May 2013.

[230] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. Woodruff, "Fast approximation of matrix coherence and statistical leverage," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 3475–3506, 2012.

[231] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Foundations and Trends in Theoretical Computer Science*, vol. 10, pp. 1–157, Oct. 2014.

[232] J. Kovačević, V. K. Goyal, and M. Vetterli, *Fourier and Wavelet Signal Processing*, Cambridge University Press, Cambridge, 2015, http://www.fourierandwavelets.org/.

[233] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva, "A survey of surface reconstruction from point clouds," *Computer Graphics Forum*, 2016.

[234] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[235] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, 2006, vol. 7.

[236] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva, "Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips," *IEEE Trans. on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2149–2158, Dec. 2013.

[237] H. Doraiswamy, N. Ferreira, T. Damoulas, J. Freire, and C. T. Silva, "Using topological analysis to support event-guided exploration in urban data," *IEEE Trans. on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2634–2643, 2014.

[238] J. Poco, H. Doraiswamy, H. T. Vo, J. L. D. Comba, J. Freire, and C. T. Silva, "Exploring traffic dynaics in urban environments using vector-valued functions," *Computer Graphics Forum (in proceedings of EuroVis 2015)*, vol. 34, no. 3, pp. 161–170, 2015.

[239] J. A. Deri and J. M. F. Moura, "Taxi data in New York City: A network perspective,"

in *Proc. Asilomar Conf. Signal, Syst. Comput.*, Pacific Grove, CA, Nov. 2015, pp. 1829–1833.

[240] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inf. Theory, sp. iss. Wavelet Transforms Multiresolution Signal Anal.*, vol. 38, no. 2, pp. 713–718, Mar. 1992.

[241] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Process.*, vol. 2, no. 2, pp. 160–175, Apr. 1993.

[242] M. Vetterli, P. Marziliano, and T. Blu, "Sampling discrete-time piecewise bandlimited signals," in *Proc. Sampling Theory and Applications Workshop*, 2001, pp. 97–102.

[243] Martin M. Vetterli, P. Marziliano, and T. Blu, "A sampling theorem for periodic piecewise polynomial signals," in *IEEE Conference on Acoustics, Speech and Signal Processing*, 2001, vol. 6, pp. 3893–3896.

[244] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *arXiv:1606.09375*, 2016.