Graph-based structures in data science: fundamental limits and applications to machine learning

THÈSE N^{O} 7644 (2017)

PRÉSENTÉE LE 11 MAI 2017

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE TRAITEMENT DES SIGNAUX 2
PROGRAMME DOCTORAL EN GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Nathanaël PERRAUDIN

acceptée sur proposition du jury:

Prof. J.-Ph. Thiran, président du jury
Prof. P. Vandergheynst, directeur de thèse
Dr R. Gribonval, rapporteur
Prof. A. Ortega, rapporteur
Prof. P. Frossard, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE
Suisse
2017

To my godson Louis Coquoz.

Abbreviations

CDF	**C**umulative **D**ensity **F**unction
DCT	**D**iscrete **C**osine **T**ransform
DFT	**D**iscrete **F**ourier **T**ransform
EMD	**E**arth **M**oving **D**istance
GSP	**G**raph **S**ignal **P**rocessing
i.i.d.	**i**ndependent and **i**dentically **d**istributed
k**-NN**	k-**N**earest **N**eighbors
PDF	**P**robability **D**ensity **F**unction
PSD	**P**ower **S**pectral **D**ensity
RKHS	**R**eproducing **K**ernel **H**ilbert **S**pace
SNR	**S**ignal to **N**oise **R**atio
STFT	**S**hort **T**ime **F**ourier **T**ransform
TV	**T**otal **V**ariation

Symbols

$\mathbb{R}$	Space of real numbers
$\mathbb{C}$	Space of complex numbers
$\mathbb{Z}$	Space of integer numbers
$\mathcal{M}$	Manifold
$\mathcal{C}^1(\mathcal{M})$	Space of continuous derivable function on $\mathcal{M}$
$\operatorname{vol}\mathcal{M}$	Volume of the manifold
$\mathcal{V}$	Vertex set
$\mathcal{E}$	Edge set
$\mathcal{W}$	Weight function $\mathcal{V}\times\mathcal{V}\to\mathbb{R}$
N	Number of vertices (Node): $\lvert\mathcal{V}\rvert$
E	Number of edges: $\lvert\mathcal{E}\rvert$
O_c	Order of the Chebyshev Polynomial
v_i	Vertex indexed by i
i,n	Indexes for the Vertex set: $1,2,\dots N$
ℓ	Indexes for the frequency set: $0,1,\dots N-1$
$\boldsymbol{W}$	Weight matrix
$\boldsymbol{D},\boldsymbol{d},d$	Degree matrix, vector, function
$\boldsymbol{L}$	Laplacian matrix
$\boldsymbol{U}$	Graph Fourier matrix
$\boldsymbol{\Lambda}$	Diagonal matrix of Laplacian eigenvalues $\boldsymbol{\lambda}$
${\mathcal{T}_i}^G g$	Graph localization operator of the kernel g on the vertex v_i (Definition 15)
$\boldsymbol{x}$	Graph signal
$\boldsymbol{A}^*$	Hermitian of $\boldsymbol{A}$ (conjugate and transposed)
$H(\boldsymbol{x})$	Shannon entropy of $\boldsymbol{x}$ (See equation 4.3)

$s_p(\boldsymbol{x})$	p-concentration function of $\boldsymbol{x}$ (See equation 4.4)
$\mu_{\mathcal{G}}$	Coherence of the graph Fourier basis $\boldsymbol{U}$ (Definition 9)
$\boldsymbol{\delta}_i$	Kroneker delta localized on the node i
$\lfloor n \rfloor_N$	Maps an integer n to $1 \dots N$ using $\lfloor n \rfloor_N = n - N \text{ floor } \left(\frac{n-1}{N}\right)$
$\mathcal{S}_{\mathcal{G}}$	Graph spectrogram function (Definition 16)
$\boldsymbol{S}_{\mathcal{G}}$	Discretized graph spectrogram (Definition 17)
$\boldsymbol{A}_{\text{g}}$	Analysis operator of a filter bank (Definition 11)
$g(\boldsymbol{\lambda})$	Vector composed of the kernel evaluated on the eigenvalue set, i.e., $g(\boldsymbol{\lambda})[\ell] = g(\boldsymbol{\lambda}[\ell]) = g(\lambda_\ell)$

Notation conventions

In this thesis we use bold fonts for vectors ($\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\mu}$) and matrices ($\boldsymbol{A}, \boldsymbol{W}, \boldsymbol{L}$) and we reserve normal fonts for indexes (i, n, m, k) and functions (f, g, h). Also the localization operator returns a vector, it is written as $\mathcal{T}_i^G g$. Vector elements are indexed with brackets $\boldsymbol{x}[i]$ and functions with parentheses $f(t)$. The Hermitian (and the transposed) of a vector/matrix $\boldsymbol{x}$ is written $\boldsymbol{x}^*$. Finally, we use $\dot{x}$ as the estimator of x.

Acknowledgements

A Ph.D. is fantastic but laborious adventure, and it is impossible to carry out alone. I am grateful to the people that directly or indirectly, helped me during this journey.

First of all, I would like to thank my advisor Pierre Vandergheynst for accepting me in his research group. He trusted me and gave me the freedom to choose my own research topics. He was generous, supported my collaborations with other scientists and he also advised and guided me through the jungle of the academic world.

Then, I thank my great friend and office mate Vassilis Kalofolias for enduring my company and bringing a lot of joy both in my academic and personal life. I am also grateful to my awesome colleague Andreas Loukas who helped me in my research and my thesis. Likewise, I have a special thought for the amazing Nauman Shahid. He made my academic world a fun place.

Next, I thank Johny Johan and Johny Lionel for their direct inputs on my thesis. They belong with Johny Max, Johny Yann and Johny Fab to the sensational geekroom. I thank them all for sharing interesting, funny and unpredictable discussions. With them, I found a group of great persons with whom I felt very comfortable and accepted.

I also thank Kirell Benzi for the astonishing Shanghai experience and all our interesting talks. I thank Michaël Defferrard for the nice projects we supervised together and all the ELE 227 people that made the LTS2 lab-life memorable.

Then, I thank my great co-authors David Shuman, Benjamin Ricaud, Nicki Hollighaus and Peter Sondergaard for introducing me to research. I am grateful to Peter Balazs for inviting me in his group before I started my Ph.D.

Furthermore, I thank my mother, my father and my siblings. They have shaped my personality and were always present when needed. I would not be in this situation without them.

Finally, I thank my better half, Milena, who stood by me during my entire thesis and always supported me. Thanks to her, the four years of my Ph.D. were so far the best of my life.

Lausanne, February 2017 Nathanaël Perraudin

Abstract

State-of-the-art data analysis tools have to deal with high-dimensional data. Fortunately, the inherent dimensionality of data is often much smaller, as it has an internal structure limiting its degrees of freedom. In most cases, this structure can be approximated using a graph, i.e., a set of nodes connected by edges. Based on this idea, graphs have been largely used in semi-supervised and unsupervised learning. The canonical assumption when incorporating a graph prior is to assume that the signal is smooth with respect to the graph, i.e., that connected pieces of data have similar values. Nevertheless, this hypothesis is insufficient to characterize more complex relations between the data and its structure, the graph.

To this end, the field of Graph Signal Processing (GSP) extends the graph smoothness hypothesis to a more general assumption. Its key idea is to think of the signal as a sum of harmonic modes, which are obtained from the eigen-decomposition of the Laplacian, an operator representing the second order derivative. Thus, the essence of GSP resides in the Fourier transform defined by the projection on the Laplacian eigenvectors basis, which allows us to filter graph signals.

However, GSP suffers from a counter-intuitive irregularity. Contrarily to the classical Fourier basis, the energy of the graph Laplacian eigenvectors is not spreading uniformly on the vertex set and can be highly concentrated in a region of the graph. This non-uniformity results in a lack of *any intuitive translation operator* and consequently poses challenges to generalize some of the classical signal processing theory, e.g., stationarity, uncertainty principles, or sampling. In this thesis, we answer these difficulties by treating each node specially according to its surrounding structure. This is achieved using an operator called *localization* that shifts a kernel defined in the spectral domain around a specific node while adapting to the local graph structure.

In the first part of the thesis, we focus on harmonic analysis in GSP. We start by making use of the norm of localized kernels to capture the local spectral features of the graph vertices. To illustrate the relevance of this analysis we then introduce *structural clustering*, an algorithm that groups nodes according to the role they play in the graph. Then, we bound the vertex-frequency concentration of graph filter banks atoms. Because of the irregular structure of graphs, we construct a *local uncertainty principle* that is, again, driven by the norm of a

localized kernel.

In the second part of the thesis, we tackle GSP problems from a machine learning perspective. We use the localization operator to extend the notion of *stationarity* to graph signals. It consists of a *graph signal probabilistic framework* allowing us to optimally solve inverse problems. Finally, we show how the *graph total variation* can be used in semi-supervised learning and prove its consistency with respect to the underlying graph manifold.

Key words: Graph, graph signal processing, spectral graph theory, semi-supervised learning, stationarity, graph stationarity, graph probabilistic framework, graph uncertainty principle, local uncertainty principle, structural clustering, clustering, graph total variation, manifold regularization, supervised learning, graph learning, graph spectrogram, machine learning, data science, graph structure

Résumé

La haute dimensionnalité des données est un problème important en science des données car une solution globale demande une quantité exponentielle de calculs et d'échantillons par rapport à la dimension. Heureusement, les données possèdent aussi souvent une structure interne qui limite leurs degrés de liberté et restreint drastiquement l'espace des solutions possibles. Même si dans la plupart des cas, cette structure n'est pas directement accessible, elle peut être approximée un utilisant un graphe, c.-à-d. un ensemble de nœuds connectés par des arrêtes. En se fondant sur ce principe, les graphes ont été largement utilisés en apprentissage semi et non supervisé où il est supposé que le signal varie lentement sur le graphe, c.-à-d. que les nœuds connectés ont des valeurs similaires. Néanmoins, cette hypothèse est insuffisante pour caractériser des relations plus complexes entre les données et leur structure. Par exemple, lors d'un embouteillage, la vitesse des voitures oscille le long de la route et des véhicules relativement proches peuvent avoir des vitesses significativement différentes.

Le traitement du signal sur graphe est un domaine de la science des données qui généralise l'hypothèse traditionnelle de régularité à un concept plus général. Son idée principale est de voir le signal comme une somme de modes harmoniques obtenus par la décomposition spectrale du Lapacien. Cela permet de définir une transformée de Fourier et une notion de filtrage sur graphe.

Malheureusement, contrairement au cas standard, les vecteurs propres du graphe ne s'étendent pas uniformément sur les nœuds et peuvent être très concentrés sur une région du graphe. De cette différence résulte l'absence d'opérateur de translation et, en conséquence, des problèmes pour généraliser certains concepts classiques comme la stationnarité, les principes d'incertitude ou l'échantillonnage. Dans cette thèse, nous répondons à ces difficultés en acceptant que chaque nœud est unique et doit être traité selon la structure de son voisinage. Nous réalisons cet objectif en utilisant un operateur appelé localisation qui déplace un noyau vers un nœud spécifique tout en s'adaptant à la structure locale du graphe autour de celui-ci.

Dans la première partie de la thèse, nous nous concentrons sur l'analyse harmonique en traitement du signal sur graphe. Nous débutons en utilisant la norme des noyaux localisés pour capturer les caractéristiques locales des nœuds du graphe. Afin de montrer la pertinence de notre analyse, nous construisons un algorithme de clustering qui groupe les nœuds selon leur

rôle dans le graphe. Ensuite, nous proposons une borne à la concentration vertex-fréquence des atomes d'un banc de filtre sur graphe. En raison de la structure irrégulière du graphe, nous construisons un principe d'incertitude local, qui est à nouveau conditionné par la norme d'un noyau localisé.

Dans la deuxième partie de la thèse, nous prenons une perspective d'apprentissage automatique. Nous utilisons l'opérateur de localisation pour étendre la stationnarité aux signaux sur graphe. Le résultat est une théorie probabiliste qui permet de résoudre de façon optimale des problèmes inverses. Enfin, nous montrons comment la variation totale peut être utilisée en apprentissage semi-supervisé et nous démontrons sa convergence vis-à-vis de la variété sous-jacente au graphe.

Mots clefs: Graphe, traitement du signal sur graphe, théorie spectrale des graphes, apprentissage semi-supervisé, stationnarité, stationnarité sur graphe, framework probabiliste pour graphe, principe d'incertitude sur graphe, principe d'incertitude local, clustering structurel, clustering, variation totale sur graphe, régularisation sur variété, apprentissage supervisé, apprentissage sur graphe, spectrogramme de graphe, apprentissage automatique, science des données, structure des graphes

Contents

1 Introduction

The amount of captured data has grown drastically during the last decades bringing new possibilities but also challenges for data science. One of the most prevalent ones is that the dimensionality of the information, i.e., the number of degrees of freedom, has greatly increased as well. Unfortunately, both the number of samples and the computations required to model the entire space grow exponentially with the data dimensionality. Fortunately, in general, the data possesses a strong intrinsic structure, which prevents it to span the entire space. As a result, models do not need to be able to provide solutions for the whole space but only need to consider the structured subspace on which the data lives. For example, a classifier built from a digits dataset only needs to categorize deformed digits and not houses, cars, trees or any other images.

A convenient way to express this assumption mathematically is to consider that the data is sampled from a low-dimensional manifold embedded in a high-dimensional space. In practice, as we only observe samples, the manifold is unknown and can at best be estimated. Nevertheless, in many cases, we can utilize a graph (a set of nodes called vertices connected by edges) to approximate its structure and its operators. For example, as illustrated in Figure 1.1, following the graph edges allows us to estimate the manifold geodesic distances.

A graph is usually acquired in two different ways. First, it may be built using the data itself. In point clouds for instance, we typically connect the nearest samples. In this setting, it has been proven that the most important graph operators converge toward their manifold equivalent [9, 8]. Second, the graph may come naturally with the data. In traffic signals for example, cars are forced to follow roads and to circulate in specific directions. As a result, a graph can be defined directly from the road network. Another example is the activation patterns of brain signals that follow the cerebral neurons.

Graph structures have been exploited by the machine learning community in mainly three different directions. First, in unsupervised learning, where the objective is to automatically group/cluster similar samples, graphs can be used to approximate the manifold geodesic distance (see Figure 1.1) and allows us to partition the samples in an embedded space [131,

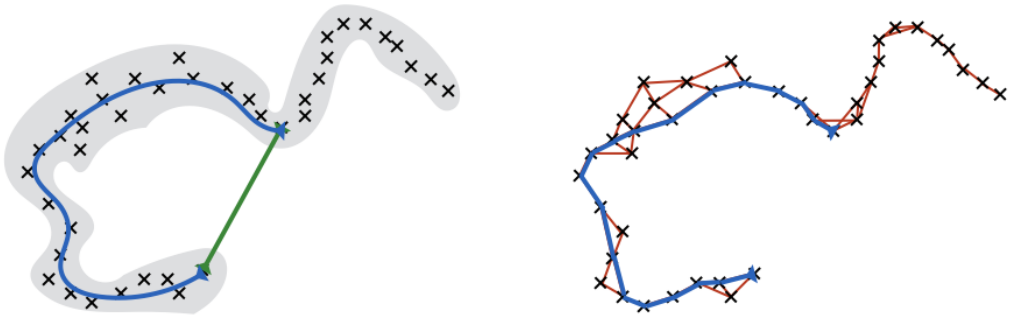


Figure 1.1 – On the left, we are given random samples from the manifold displayed in gray. We show the Euclidean distance in green and the geodesic distance in blue. On the left, we construct a graph by connecting a sample to its nearest neighbors. Following the graph edges, we can approximate the geodesic distance.

132]. Second, in semi-supervised learning, where one tries to infer missing information/labels given the available ones, the graph inserts the structure contained in the unlabeled samples and improves substantially the quality of the prediction [7, 122]. Finally, in a recommender system, graphs are used to transfer pieces of information between similar profiles or items [55].

This thesis argues that most of these algorithms are based on the same hypothesis: "similar samples are connected together and have analogous labels," i.e., that the label function fluctuates slowly along the graph edges. Problematically, this assumption may prove to be insufficient in many cases. Consider the example presented in Figure 1.2, where the goal is to predict the color of the center node. If we only look at the neighbors of the center node and assume that the signal changes slowly, the value assigned would be yellow. However, an overall observation of the signal's oscillating behavior naturally suggests that the hidden color is red. One promising way to overcome this issue is to use different hypotheses to regularize graph signals. For example, in a traffic jam, the car velocity has an oscillating behavior in function of the road position and additionally nearby cars can drive with significantly different speeds. If the road is assumed to be a one-dimensional manifold, the car speed can be modeled as a sum of oscillating functions in a well-specified frequency band.

Graph Signal Processing (GSP) is designed to handle and unify all these cases under the comprehensive hypothesis that *the data/signal depends on the graph structure.*

Similarly to classical signal processing where the signal is seen as a sum of frequency components, the fundamental idea of GSP is to view the signal as a sum of harmonic modes which are obtained from the eigen-decomposition of the graph Laplacian $\boldsymbol{L}$. This interpretation is justified by the following facts. First, the Laplacian corresponds to the second order derivative, which can classically be used to define Fourier modes. Second, we recover the traditional graph Fourier transform when a ring graph is used (see Section 2.2.1). And third, the Laplacian eigenvectors converge toward the Fourier mode of the underlying manifold [9]. In practice, if we denote the eigendecomposition of $\boldsymbol{L}$ as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^*$, the orthonormal matrix $\boldsymbol{U}$ is considered

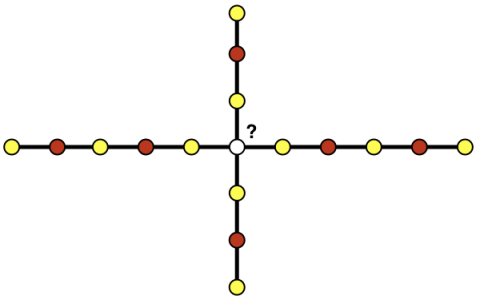


Figure 1.2 – Which color should the center node be labeled? The nearest neighbor approach leads to the color yellow. On the contrary, the spectral approach developed by graph signal processing leads to the color red.

as the graph Fourier basis and the diagonal matrix $\boldsymbol{\Lambda}$ as the graph squared frequencies (or pulsations) [114]. As a result, the graph Fourier transform reads $\hat{\boldsymbol{x}} = \boldsymbol{U}^* \boldsymbol{x}$.

This spectral approach plays a prominent role in GSP and generalizes previous machine learning techniques. For example, the classical way to employ a graph in a semi-supervised learning algorithm is to regularize the signal $\boldsymbol{x}$ using a Sobolev norm on the graph, i.e., the norm of the gradient on the graph. This quantity is expressed as the sum of the signal variations,

$$\|\nabla \boldsymbol{x}\|^2 = \sum_{i=1}^{N} \sum_{n=1}^{N} \boldsymbol{W}[i,n] \left(\boldsymbol{x}[n] - \boldsymbol{x}[i]\right)^2, \tag{1.1}$$

where $\boldsymbol{W}$ is the weight/connectivity matrix and ∇ the gradient on the graph [122]. In GSP, we usually observe the same quantity with a spectral point of view, i.e.,

$$\|\nabla \boldsymbol{x}\|^2 = \sum_{\ell} \lambda_\ell \hat{\boldsymbol{x}}^2[\ell], \tag{1.2}$$

where λ_ℓ is the eigenvalue corresponding to the spectral component $\hat{\boldsymbol{x}}[\ell]$. It can be seen that (1.2) penalizes the spectral component of the signal $\hat{\boldsymbol{x}}[\ell]$ linearly with respect to the graph eigenvalues λ_ℓ. While both quantities (1.1) and (1.2) are equivalent, (1.2) has an interesting signal processing interpretation: it represents a high pass filter. This suggests that, if the signal $\boldsymbol{x}$ belongs to a specific frequency band, we can change the frequency regularization weights to another band-cut function h [82, 138], i.e.,

$$\|h(\boldsymbol{L})\boldsymbol{x}\|^2 = \sum_{\ell} h^2(\lambda_\ell) \hat{\boldsymbol{x}}^2[\ell]. \tag{1.3}$$

For example, if we search for a low-frequency signal in Figure 1.2, i.e., using (1.1) or (1.2) as a penalization, we obtain a yellow label. On the contrary, if we search for a high-frequency signal using (1.3) with $h(\lambda) = \frac{1}{1+\lambda}$, we recover the red label.

The spectral approach adopted by GSP allows to generalize important tools of traditional signal processing. Filtering is obtained by weighting the signal in the Fourier domain [114] and transformations such as the wavelets transform can be constructed by aggregating different filters [52]. This being said, besides these similarities with traditional signal processing, GSP also has two counterintuitive discrepancies.

- First, as illustrated in Figure 1.3, the Laplacian eigenvectors do not spread uniformly over the graph nodes and some Fourier modes can be highly localized on a particular set of vertices. As a consequence, an operation targeting a filtering operation can affect a part of the graph vertices [89].

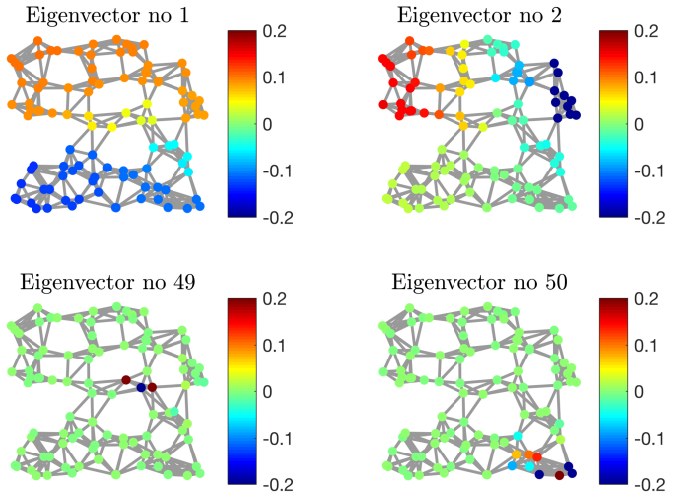


Figure 1.3 – Eigenvector localization. In a random geometric graph, the eigenvectors associated with the lowest eigenvalues span all vertices, while the ones associated with the highest eigenvalues are generally localized around a set of nodes (The graph contains $N = 100$ vertices and has eigenvalues $\lambda_1 = 0.0044$, $\lambda_2 = 0.0174$, $\lambda_{49} = 2.0709$, $\lambda_{50} = 2.0836$, $\lambda_{\max} = 4.7515$).

- Second, the eigenvalues are irregularly spaced. As such, compared to the classical case, there is in general no obvious relation with a period of oscillation of the corresponding Fourier mode. As a result, the same filter may have a completely different effect on two different graphs.

The challenges of GSP are consequences of these differences. From a computational point of view, the irregular structure of the graph Fourier basis hinders an efficient implementation of the Fourier transform. Essentially, an explicit computation requires the diagonalization of the Laplacian, which has a cubic complexity with respect to the number of vertices. From a theoretical point of view, there exists no translation operator for graphs. Hence, traditional signal processing operations and transforms are often complicated to generalize to the graph setting, e.g., stationarity, uncertainty principles and sampling theorems.

We posit that the key idea to overcome these issues lies in accepting that the translation

operator is not necessary on graphs. By nature, the local structure of the graph changes from node to node. As a result, we argue that each node should be treated differently according to its surrounding topology. This is exactly the motivation behind the *localization operator* defined as

$$\mathcal{T}_i^G g := \boldsymbol{U} g(\boldsymbol{\Lambda}) \boldsymbol{U}^* \boldsymbol{\delta}_i,$$

which adapts the kernel g, defined in the spectral domain, to the graph structure around the vertex v_i (see for example Figure 1.4). Localization can be viewed as a weaker version of translation, because the localized kernels can be interpreted as shifted versions of each other (see Chapter 3.1). However, contrarily to the traditional translation that preserves both the shape and the energy of the signal, the localization by adapting to the irregular graph structure does not. Nevertheless, in our perspective we consider this behavior as a way to deal with the irregular graph eigenvector spreading.

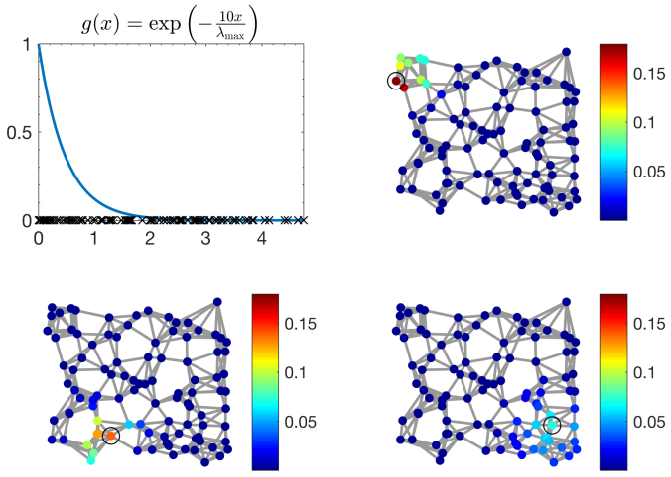


Figure 1.4 – The heat kernel localized in 3 different vertices. The localization operator adapts the kernel to the graph structure.

Within this context, the contributions of this thesis are oriented in two main directions. First, we study the foundations of graph signal processing and show how the localization operator can be leveraged to handle the irregular graph structure (Chapters 3, 4 and 5). Second, we provide new algorithms and consistency results for semi-supervised learning (Chapters 5 and 6).

1.1 Thesis outline and contributions

The first three chapters of the thesis focus on studying the foundations of GSP harmonic analysis. In Chapter 2, we introduce GSP with a special emphasis on how it generalizes the classical case. This approach provides a natural motivation for the harmonic decomposition

of the Laplacian. Moreover, we present two efficient methods to filter graph signals without diagonalizing the Laplacian. These techniques are essential for the scalability of the algorithms presented in this thesis. Each of the next three chapters leverages, with a different perspective, the localization operator to handle the irregular graph structure.

Chapter 3: After introducing the localization operator in detail and describing its main properties, we focus on the correlation between the norm of a localized kernel and the irregular spectral structure of a graph. It turns out that this norm is a convenient tool to access the locality of the graph Fourier eigenvectors. Based on this property we introduce *the graph spectrogram*, a transform assigning to each node a feature vector depending on its spectral content. This leads us to propose structural clustering, an algorithm that classifies the nodes according to their role in the graph, i.e., the local structure around them.

Chapter 4: We further investigate some of the fundamental limits of GSP filter banks. Due to the irregular eigenvector spreading, the atoms of a vertex frequency transform [117] are not translated versions of each other and can have very different shapes and concentrations. Again, the norm of the localization operator appears as the prominent factor of the concentration bound. As shown by Lieb [64], the classical uncertainty principles provides a global bound that applies to the signal, regardless of its time-frequency localization. On the contrary, when it comes to graph signals, the bound evolves depending on its vertex-frequency localization. Driven by this concept, we introduce the notion of local uncertainty principles. We conclude the chapter by demonstrating how our proposed local uncertainty measures can improve the random sampling of graph signals.

In the last two chapters, we develop GSP from a more machine learning point of view. Our analysis focuses on how graphs can be used in semi-supervised learning problems.

Chapter 5: We leverage the localization operator by generalizing the notion of Wide Sense Stationarity (WSS) to graph signals. Given a temporal signal, WSS is an assumption on the first two moments of a signal stating that both are invariant with respect to time, namely: 1) the signal expectation is constant over time and 2) the correlation between instants t_1 and t_2 depends only on a single function applied to the difference $t_2 - t_1$. Similarly, Graph Wide Sense Stationarity (GWSS) is an assumption on the first two moments of a graph signal, namely 1) its expectation is constant over the vertex set and 2) its correlation between the vertices v_i and v_n is obtained through the localization of a single kernel γ called Power Spectral Density (PSD), i.e., for a zero mean signal $\mathbb{E}[\boldsymbol{x}[i]\boldsymbol{x}[n]] = \mathcal{T}_i^G g[n]$. After describing the properties of stationarity, we introduce efficient tools for model estimation and process recovery of graph signals. In particular, we construct a scalable method that estimates the PSD from a graph signal, and we build a general estimator to solve inverse problems involving stationary signals.

Chapter 6: In a semi-supervised learning context, graphs are also a predilection tool to introduce unlabeled samples in a learning algorithm because they give access to the underlying data structure (the manifold). Typically, this is done by searching for a function in a Reproduc-

ing Kernel Hilbert Space (RKHS) that not only fits the labeled points, but that is also slowly varying on the graph [7, 9]. In Chapter 6, we generalize this concept to piecewise constant functions by leveraging the graph total variation, i.e., the ℓ_1-norm of the gradient. To justify our method, we prove that the total variation of the graph signal converges toward the total variation of the function on the underlying manifold when the number of samples tends toward infinity.

The thesis concludes in Chapter 7 where we propose future research directions and give some considerations on GSP.

Contributions by order of appearance

- We propose a new scalable graph filter algorithm using Lanczos method.
- We define a new transform for graphs assigning to each node a feature vector characterizing the local structure surrounding it.
- We construct a structural clustering algorithm that classifies graph vertices according to their role in the graph.
- We build several uncertainty principles bounding the vertex-frequency concentration of graph filter banks.
- We show that the vertex-frequency concentration of a graph filter bank atom depends on its vertex-frequency localization and hence propose a local uncertainty principle.
- We generalize the notions of stationarity, Power Spectral Density (PSD) and Wiener filters to graph signals.
- Given a graph signal, we propose and characterize a scalable estimator for the PSD.
- In the context of probabilistic graph signal processing, we present a novel regularization scheme that is proven to be optimal for graph stationary signals and provide a scalable implementation for standard inverse problems.
- We propose a new semi-supervised learning scheme for piece-wise constant functions using the graph total variation.
- We prove the convergence of the graph total variation toward its manifold equivalent.

Part I

Harmonic analysis in graph signal processing

2 A gentle introduction to graph signal processing

2.1 First steps into graph signal processing

Let us start with an introduction to Graph Signal Processing (GSP), where we give a formal definition to the different objects used in this thesis.

2.1.1 Graph nomenclature

A graph consists of two sets $\mathcal{V}, \mathcal{E}$ and optionally a weight function. $\mathcal{V}$ is the set of vertices that represent the nodes of the graph. $\mathcal{E}$ is the set of edges that connects two nodes if there is a particular relation between them. To obtain a finer graph structure, this relation can be quantified by a weight function $\mathcal{W} : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ that reflects to what extent two nodes are related to each other. A graph is therefore tuple $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{W}\}$. For simplicity, we will restrict ourselves to undirected, connected and weighed graphs. Nevertheless, some of the definitions and tools presented in this thesis have been extended to directed graphs [21, 22, 140] and hypergraphs [141]. The vector/matrix terminology happens to be very convenient for graphs. Let us index the nodes from $1, \dots, N = |\mathcal{V}|$ and construct the weight matrix $\boldsymbol{W}$ by setting $\boldsymbol{W}[i, n] = \mathcal{W}(v_i, v_n)$ as the weight associated to the edge connecting the node i and the node n. If no edge exists between i and n, the weight is set to 0.

The weight function $\mathcal{W}$ is in general custom-made for each application. In the case where the data is a collection of N features vector $\boldsymbol{c}_i$ in $\mathbb{R}^M$, there are three typical schemes to create a graph.

Graph 1 (Weighted point-cloud graph)**.** *Given a set of N points with coordinates $\{\boldsymbol{c}_i\}_{i=1,\cdots N}$ and a kernel $k : \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}_+$, a weighted point-cloud graph is a graph where each point $\boldsymbol{c}_i$ is associated to a vertex v_i and the weight function $\mathcal{W}$ satisfies $\mathcal{W}(v_i, v_n) = k(\boldsymbol{c}_i, \boldsymbol{c}_n)$. In general, the kernel k is a decreasing function of the distance between the points $\boldsymbol{c}_i$ and $\boldsymbol{c}_n$. The most typical case is the Gaussian kernel*

$$k(\boldsymbol{c}_i, \boldsymbol{c}_n) = e^{-\frac{\|\boldsymbol{c}_i - \boldsymbol{c}_n\|_2^2}{\sigma^2}}.$$

Graph 2 (k-nearest neighbors graph)**.** *Given a set of N points with coordinates $\{\boldsymbol{c}_i\}_{i=1,\cdots N}$, a k-Nearest Neighbors (k-NN) graph is a graph where each point $\boldsymbol{c}_i$ is associated to a vertex v_i and two nodes v_i and v_n are connected if $\boldsymbol{c}_i$ is in the k-nearest neighbors of $\boldsymbol{c}_n$ or if $\boldsymbol{c}_n$ is in the k-nearest neighbors of $\boldsymbol{c}_i$. The weight associated to each edge is* 1*. In the literature, typical values for k range from* 6 *to* 10*.*

Graph 3 (Weighted k-NN graph)**.** *A weighted k-NN graph is a k-NN graph with a non-unitary weight associated to the edges. In this thesis, we use in most of the case a Gaussian kernel*

$$k(\boldsymbol{c}_i, \boldsymbol{c}_n) = e^{-\frac{\|\boldsymbol{c}_i - \boldsymbol{c}_n\|_2^2}{\sigma^2}},$$

where σ is set to the average of the distances between k-NN points.

A particularity of GSP is that the graph is the domain and the signal (the information) resides on it.

Definition 1 (Graph signal)**.** *A signal is defined as a function $x : \mathcal{V} \to \mathbb{R}$ assigning one value to each vertex. For convenience, we consider a signal x as a vector $\boldsymbol{x}$ of size N with the n^{th} component representing the signal value at the n^{th} vertex, i.e., $x(v_n) = \boldsymbol{x}[n]$.*

While we work in $\mathbb{R}$, almost all results can be extended to $\mathbb{C}$. We now generalize straightforwardly some classical concepts.

Definition 2 (Graph scalar product)**.** *The scalar product of two graph signals x and y:*

$$\langle x, y \rangle_{\mathcal{V}} = \sum_{n=1}^{N} x(v_n) y^*(v_n) = \sum_{n=1}^{N} \boldsymbol{x}[n] \boldsymbol{y}^*[n] = \boldsymbol{y}^* \boldsymbol{x} = \langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{V}|}} \tag{2.1}$$

where $\boldsymbol{y}^*$ is the complex conjugate transposed of $\boldsymbol{y}$.

For convenience, we abusively say that $\boldsymbol{x}$ is a graph signal and use only this notation. We will additionally not use the subscript for the scalar product, but the reader should always keep in mind the equivalence presented above.

Definition 3 (Degree of a vertex)**.** *Given a node $v_i \in \mathcal{V}$, its degree $d(v_i) = \boldsymbol{d}[i]$ is defined as the sum of the weights of the connected edges, i.e., $\boldsymbol{d} = \boldsymbol{W}\boldsymbol{1}$ or*

$$d(v_i) = \sum_{n=1}^{N} \mathcal{W}(v_i, v_n).$$

The degree somewhat measures the connectivity of each node of the network. Nodes with a low degree are more isolated. Figure 2.1 shows the node for a sensor network graph, i.e., a random geometric graph.

Example 1 (Geometric sensor network). *Let us suppose that some sensors are distributed uniformly at random over a given area. Each of them records some data. We can model this network with a graph where the vertices represent the sensors and the edges represent physical proximity. In this case, the signal associated to each vertex is the value returned by the sensor. We use a special type of graph to model this process that we refer to as* sensor graph

Graph 4 (Sensor graph). *A sensor Graph is a random geometric sensor network generated as follows. First the coordinates of the N points are drawn uniformly from the plane unit square* $[0,1] \times [0,1]$. *Then the weighted k-NN scheme of Graph 3 is used to connect the vertices. An example is displayed in Figure 2.1.*

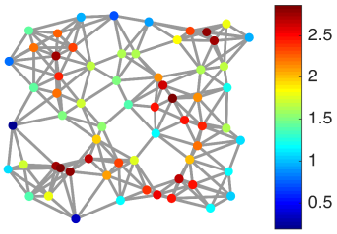


Figure 2.1 – Geometric sensor graph. The color on the node represents the value of the signal on the graph: here the degree.

On a graph, the notion of distance has to be redefined and several possibilities exist, i.e., the resistance distance [57], or the diffusion distance [61]. A very elementary one is the length of the shortest path that we refer as *hop distance*.

Definition 4 (Graph Path). *A* graph path *is a set of vertices* $(v_1, v_2, ..., v_p)$ *with the property that* $[v_i, v_{i+p}] \in \mathcal{E}$ *for* $1 \leq i \leq p-1$. *The length of a path is defined as the cardinality of the path set minus one.*

From the definition of graph path, we now derive a notion of connected graph.

Definition 5 (Connected vertices and connected graph). *We say that two vertices* v_i *and* v_j *are connected if there exists a path such that* $v_1 = v_i$ *and* $v_p = v_j$.
A graph is connected if every vertex is connected to every other vertex.

As an example, the graph presented in Figure 2.1 is connected.

Definition 6 (Hop distance). *The hop distance between vertices* v_i *and* v_n: $h_{\mathcal{G}}(v_i, v_n)$ *is defined as the length of the shortest path between them. If no path exists, then the hop distance is infinite.*

2.1.2 Gradient and Laplacian

In GSP, the most fundamental operator is the graph Laplacian. Let us start by presenting the discrete combinatorial Laplacian operator in the classical case. Let $\boldsymbol{x}$ be a discrete signal.

The gradient is usually defined as $\nabla_d \boldsymbol{x}[i] = \boldsymbol{x}[i+1] - \boldsymbol{x}[i]$, which is the finite difference of two consecutive values. In this case, the divergence operator (adjoint of the gradient) becomes $\mathrm{div}_d \boldsymbol{x}[i] = \boldsymbol{x}[i-1] - \boldsymbol{x}[i]$. Using theses definitions, the discrete Laplacian becomes

$$\begin{aligned}(\Delta_d \boldsymbol{x})[i] &= (\mathrm{div}_d \nabla_d \boldsymbol{x})[i] \\ &= \nabla_d \boldsymbol{x}[i-1] - \nabla_d \boldsymbol{x}[i] \\ &= 2\boldsymbol{x}[i] - \boldsymbol{x}[i-1] - \boldsymbol{x}[i+1] \end{aligned} \tag{2.2}$$

A similar procedure is used for the graph case. In fact, the construction of the Laplacian operator for graphs generalizes the classical discrete Laplacian. It acts as a second order derivative and is defined as:

$$\boldsymbol{L} := \mathrm{div}_{\mathcal{G}} \nabla_{\mathcal{G}}, \tag{2.3}$$

where $\mathrm{div}_{\mathcal{G}}$ and $\nabla_{\mathcal{G}}$ are the divergence and the gradient[1] defined on the graph. *We emphasize that the final formulation of the Laplacian is dependent of the definition of the gradient* $\nabla_{\mathcal{G}}$. We illustrate the process in this section. To get more details about those computations, we invite the reader to read [49, 35].

Gradient of a graph signal: For our derivations, we chose to define the combinatorial gradient. However other definitions are widely used such as the normalized gradient.

Definition 7 (Edge-derivative)**.** *For a signal* $x : \mathcal{V} \to \mathbb{R}$*, the* edge derivative *is an application* $\frac{\partial x}{\partial_e} : \mathcal{E} \to \mathbb{R}$ *defined as:*

$$\frac{\partial x}{\partial e_{i,n}} = \sqrt{\mathcal{W}(v_i, v_j)}\,(x(v_n) - x(v_i)) = \sqrt{\boldsymbol{W}[i,n]}\,(\boldsymbol{x}[n] - \boldsymbol{x}[i]) \tag{2.4}$$

The term $(\boldsymbol{x}[n] - \boldsymbol{x}[i])$ is the finite difference between the values at two adjacent vertices. The term $\sqrt{\boldsymbol{W}[i,n]}$ weights those finite differences. When two nodes are strongly connected (i.e., close), then the weight increases the derivative. On the contrary if the two nodes are weakly connected (i.e., far away), the weight decreases the derivative. The gradient of a graph signal $\boldsymbol{x}$ groups all edge derivatives into a linear operator $\nabla_G : \mathbb{R}^{|\mathcal{V}|} \to \mathbb{R}^{|\mathcal{E}|}$ defined as:

$$\boldsymbol{y} = \nabla_G \boldsymbol{x} = \left[\frac{\partial \boldsymbol{x}}{\partial e}\right]_{e \in \mathcal{E}}. \tag{2.5}$$

Divergence of an edge signal: The divergence of an edge signal is defined as the adjoint of the gradient operator on graphs. For a vertex signal $\boldsymbol{x}$ and an edge signal $\boldsymbol{y}$, it satisfies:

$$\langle \nabla_{\mathcal{G}} \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{E}|}} = \langle \boldsymbol{x}, \mathrm{div}_{\mathcal{G}} \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{V}|}}. \tag{2.6}$$

[1] When the graph is unweighted, the linear operator associated to the gradient is simply the incidence matrix.

Using this relationship, the divergence operator $\mathrm{div}_{\mathcal{G}}\boldsymbol{y}:\mathbb{R}^{|\mathcal{E}|}\to\mathbb{R}^{|\mathcal{V}|}$ is defined as the linear operator:

$$\left(\mathrm{div}_{\mathcal{G}}\boldsymbol{y}\right)[n]=\frac{1}{2}\sum_{i}\sqrt{\boldsymbol{W}[i,n]}\boldsymbol{y}[i,n]-\sqrt{\boldsymbol{W}[n,i]}\boldsymbol{y}[n,i] \tag{2.7}$$

The details of computations can be found in Appendix A.1.

From definition (2.3) and the definitions of the gradient and the divergence, we now derive the expression of the combinatorial Laplacian operator $\boldsymbol{L}:\mathbb{R}^{|\mathcal{V}|}\to\mathbb{R}^{|\mathcal{V}|}$ as

$$\boldsymbol{L}=\mathrm{div}_{\mathcal{G}}\nabla_{\mathcal{G}}=\boldsymbol{D}-\boldsymbol{W}, \tag{2.8}$$

where $\boldsymbol{D}$ is the diagonal degree matrix (with diagonal entries $\boldsymbol{D}[i,i]=\boldsymbol{d}[i]$). The derivation details are given in Appendix A.2. In this thesis we use the notation $\boldsymbol{L}$ for both the Laplacian operator and its associated matrix.

Example 2 (Laplacian of the "ring" graph)**.** *On the "ring" graph (see Figure 2.4 left and Graph 6), every node is connected to its left and right neighbors. The resulting weight matrix is*

$$\boldsymbol{W}[i,n]=\begin{cases}1 & \text{if } |i-n|=1\\ 1 & \text{if } i=1, \text{ and } n=N\\ 1 & \text{if } i=N, \text{ and } n=1\\ 0 & \text{otherwise.}\end{cases}$$

From the weight matrix the Laplacian is computed as

$$\boldsymbol{L}[i,n]=\begin{cases}-1 & \text{if } |i-n|=1\\ -1 & \text{if } i=1, \text{ and } n=N\\ -1 & \text{if } i=N, \text{ and } n=1\\ 2 & \text{if } i=n\\ 0 & \text{otherwise.}\end{cases}$$

This expression matches the classical discrete Laplacian of (2.2).

The "ring" graph is very important in graph signal processing as it makes the connection between traditional and graph signal processing. We detail this case in Section 2.2.1.

Other Laplacian definitions: Other Laplacian definitions are widely used in practice. The most common alternative is the normalized Laplacian:

$$\boldsymbol{L}'=\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{D}^{-\frac{1}{2}}=\boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{D}-\boldsymbol{W})\boldsymbol{D}^{-\frac{1}{2}}=\boldsymbol{I}-\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}}. \tag{2.9}$$

Using this definition, each edge is re-weighted such that the degree of each node is 1. In this case, the spectrum of the Laplacian is bounded by 2. The choice of the Laplacian highly depends on the application and, unfortunately, there is no rule to tell which one should be used in any given situation.

The results presented in this thesis are independent of the Laplacian definition and only require a SPSD operator. Furthermore, they can be used with directed graphs [140, 22, 21] and hypergraphs [141]. Independently of the chosen edge derivative definition (provided it is linear), the graph Laplacian will be, by construction, a symmetric positive semi-definite operator. We thus build a unified graph spectral theory that applies to all of these cases.

2.2 Spectral graph theory

The spectral theory of graphs is a generalization of the classical case. Let us start by reminding some of the classical properties of the continuous case. For L^2 continuous functions, the derivative in the time domain is equivalent to a multiplication in the Fourier domain, i.e.,

$$\widehat{\nabla f}(\omega) = j\omega \hat{f}(\omega), \tag{2.10}$$

where $\nabla f(x) = \frac{\partial f}{\partial x}(x)$. From this relation, we find a special expression for the second order derivative,[2] i.e., the Laplacian, as

$$\widehat{\nabla^* \nabla f}(\omega) = \widehat{\Delta_L f}(\omega) = \omega^2 \hat{f}(\omega), \tag{2.11}$$

where $\Delta_L = \nabla^* \nabla = -\frac{\partial^2}{\partial x^2}$. Equation (2.11) highlights the link between the Laplacian operator and the Fourier transform because it implies two different things:

- the eigenfunctions of the Laplacian are the Fourier modes,
- the Laplacian eigenvalues correspond to the squared pulsations.

A similar analysis can be done for the discrete case where the discrete Fourier eigenvectors are also the eigenfunctions of the discrete Laplacian (2.2).

This idea is also used to define the graph Fourier basis. In the graph case, the Laplacian $\boldsymbol{L}$ is, by construction, a symmetric positive semi-definite operator. Thus, from the spectral theorem, it possesses a complete set of orthonormal eigenvectors $\boldsymbol{U}$ and can be written as:

$$\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^*, \tag{2.12}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the Laplacian eigenvalues. $\boldsymbol{U}$ is considered to be the graph Fourier basis and the graph Fourier eigenvectors are denoted by $\{\boldsymbol{u}_\ell\}_{\ell=0,1,\dots,N-1}$. The eigenvalues somehow represent the squared pulsation, i.e., how much the eigenvectors do

[2] In turns out that the adjoint of the gradient, called divergence is $\mathrm{div} f(x) = \nabla^* f(x) = \nabla f(x)$. This can follows trivially from (2.10).

oscillate on the graph. For convenience, and without loss of generality, we order the set of eigenvalues increasingly: $0 \leq \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{N-1} = \lambda_{\max}$. This ordering associates slow pulsation with low indexes.

Connected graphs have exactly only one zero eigenvalue (pulsation/frequency) corresponding to the constant eigenvector (for the combinatorial Laplacian). Furthermore, it has been proven that the multiplicity of the zero eigenvalue is equal to the number of connected components [23]. In that case, there is no connections between the different connected components and in GSP they are usually handled independently.

Graph 5 (Path)**.** *Let us illustrate this spectral decomposition with an example. A "path" is a graph where all nodes are connected to their two neighbors except the first and the last one, i.e.,*

$$\boldsymbol{W}[i,n] = \begin{cases} 1 & \text{if } |i-n| = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Example 3 (Eigenvectors of the "path" graph)**.** *Figure 2.2 displays the first six eigenvectors, i.e., the eigenvectors associated to the lowest pulsation / frequency. As the eigenvalues increase, they naturally oscillate more. In fact, it has been proven in [125] that the basis of the path graph is a discrete cosine basis. This basis is largely used in image processing in order to handle border effects.*

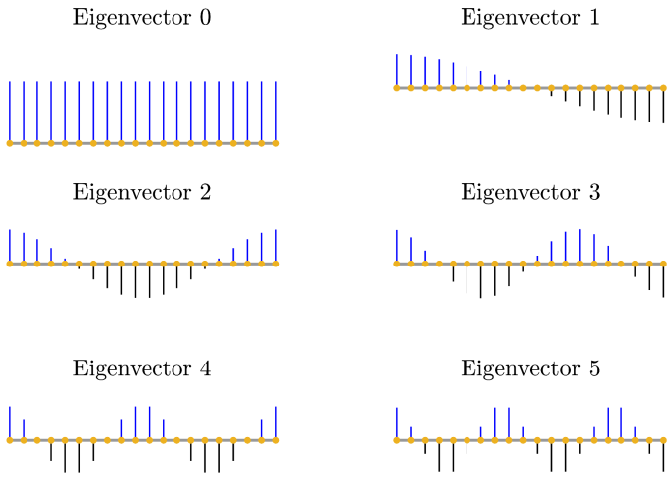


Figure 2.2 – First eigenvectors of the path graph (20 nodes).

Definition 8 (Graph Fourier transform)**.** *The graph Fourier transform $\hat{\boldsymbol{x}} \in \mathbb{R}^N$ (or $\in \mathbb{C}^N$) of a function $\boldsymbol{x} \in \mathbb{R}^N$ (or $\in \mathbb{C}^N$) defined on a graph $\mathcal{G}$ is the projection onto the orthonormal set of eigenvectors $\{\boldsymbol{u}_\ell\}_{\ell \in [0,N-1]}$ of the graph Laplacian associated with $\mathcal{G}$, i.e.,*

$$\hat{\boldsymbol{x}}[\ell] := \langle f, \boldsymbol{u}_\ell \rangle = \sum_{n=1}^{N} \boldsymbol{x}[n] \boldsymbol{u}_\ell^*[n], \quad \ell \in [0, N-1] \text{ or } \quad \hat{\boldsymbol{x}} = \boldsymbol{U}^* \boldsymbol{x}. \tag{2.13}$$

Conversely, since $\{\boldsymbol{u}_\ell\}_{\ell=0,1,\dots,N-1}$ *is an orthonormal basis, the inverse Fourier transform is*

$$\boldsymbol{x}[n] = \sum_{\ell=0}^{N-1} \hat{\boldsymbol{x}}(\ell)\boldsymbol{u}_\ell[n], \quad n \in [1,N], \quad \textit{or} \quad \boldsymbol{x} = \boldsymbol{U}\hat{\boldsymbol{x}}. \tag{2.14}$$

For any symmetric positive semidefinite matrix, there exists an infinite number of orthonormal eigenvectors bases. However, if the spectrum of the Laplacian has no eigenvalue with multiplicity, only one is entirely real. For convenience, we will, in that case, always choose that one. The definition of the Fourier transform possesses different properties described in [117]. For example, the Parseval relation holds for graphs:

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \hat{\boldsymbol{x}}, \hat{\boldsymbol{y}} \rangle. \tag{2.15}$$

The spectrum of the Laplacian replaces the squared pulsation (squared angular frequency) and is used as coordinates in the Fourier domain. However, compared to the classical frequencies, Laplacian eigenvalues are not regularly spaced and the eigenvalue distribution depends on graph characteristics.

Motivation of the Fourier definition Beside the classical analogy presented at the beginning of this section, there is another motivation to define the Fourier basis as the eigenvectors of the graph Laplacian. For a signal $\boldsymbol{x}$, the normalized norm of its gradient $\frac{\|\nabla_{\mathcal{G}}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}$ is a measure of how much the signal varies on the graph. If we were to seek for the k^{th} orthogonal less varying signals on the graph, we would solve the following problem:

$$\boldsymbol{x}_k = \underset{\boldsymbol{x}\perp\boldsymbol{x}_\ell \forall \ell<k}{\operatorname{argmin}} \frac{\boldsymbol{x}^*\boldsymbol{L}\boldsymbol{x}}{\|\boldsymbol{x}\|_2^2}. \tag{2.16}$$

Indeed, by definition of the Laplacian, we have

$$\left(\frac{\left\|\nabla_{\mathcal{G}}\boldsymbol{x}\right\|_2}{\|\boldsymbol{x}\|_2}\right)^2 = \frac{\langle \nabla_{\mathcal{G}}\boldsymbol{x}, \nabla_{\mathcal{G}}\boldsymbol{x}\rangle}{\|\boldsymbol{x}\|_2^2} = \frac{\langle \boldsymbol{x}, \operatorname{div}_{\mathcal{G}}\nabla_{\mathcal{G}}\boldsymbol{x}\rangle}{\|\boldsymbol{x}\|_2^2} = \frac{\boldsymbol{x}^*\boldsymbol{L}\boldsymbol{x}}{\|\boldsymbol{x}\|_2^2}$$

In practice for "regular graphs"[3] like rings, grids, or paths, the Fourier modes are intuitively the oscillating modes for given frequencies [114, 117]. A very important property of those modes is the fact that the higher the eigenvalue, the more the mode oscillates. As a consequence, high eigenvalues are associated to high frequencies and low eigenvalues correspond to low frequencies.

Example 4. *In Figure 2.3, we present an example of low order oscillating modes of the sensor graph presented in Figure 2.1. Since we use the combinatorial Laplacian, the first eigenvector is always constant. The second eigenvector looks like a wave (or a sinus) on the graph. The tenth eigenvector is still more or less smooth but has a lot of local minima and maxima.*

[3] Informally, the graph regularity can be seen as how much the topology of the graph varies from one node to another.

However, when the graph structure is "less regular" (with isolated vertices or groups of highly connected vertices), the Laplacian spectrum can differ greatly from the notion of frequency. Some Fourier modes can be highly concentrated, hence close to elements of the canonical basis. This will be studied in more detail in Chapter 4.

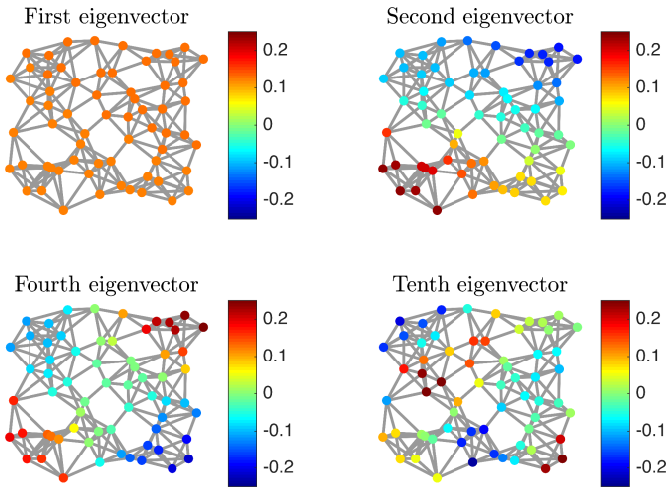


Figure 2.3 – Example of eigenvectors on a sensor graph. The first eigenvector represents the DC component of the graph. The larger the eigenvalue associated to the graph, the more the eigenvector oscillates ($N = 64$, $\lambda_0 = 0$, $\lambda_1 = 0.0488$, $\lambda_3 = 0.0996$, $\lambda_9 = 0.4198$ and $\lambda_{\max} = 3.8281$).

As we shall see in Sections 3 and 4, some of the graph spectral eigenvectors may be localized in small subsets of vertices. One simple way to know if a graph contains localized eigenvectors is to look at the modulus of the graph Fourier eigenvectors.

Definition 9 (Graph Fourier Coherence $\mu_{\mathcal{G}}$)**.** *Let $\mathcal{G}$ be a graph of N vertices. Let $\{\boldsymbol{\delta}_i\}_{i\in\{1,2,\dots,N\}}$ denote the canonical basis of $\ell^2(\mathbb{C}^N)$ of Kronecker deltas and let $\{\boldsymbol{u}_\ell\}_{\ell\in\{0,1,\dots,N-1\}}$ be the orthonormal basis of eigenvectors of the graph Laplacian of $\mathcal{G}$. The graph Fourier coherence is defined as:*

$$\mu_{\mathcal{G}} := \max_{i,\ell} |\langle \boldsymbol{\delta}_i, \boldsymbol{u}_\ell \rangle| = \max_{i,\ell} |\boldsymbol{u}_\ell[i]|.$$

2.2.1 Generalization of the classical case

It turns out that many traditional Fourier transforms correspond to particular graphs.

- The Discrete Fourier Transform (DFT) corresponds the "ring" graph.
- The Discrete Cosine Transform 2 (DCT-2) corresponds to the "path" graph.
- The two dimensional DCT corresponds to the "grid" graph.
- The two dimensional DFT corresponds to the "torus" graph.

These graphs are displayed in Figure 2.4. Strang has demonstrated in [125] the unidimensional DFT and DCT cases. The two dimensional cases are obtained through the application of [73, 74, Theorem 2.21].

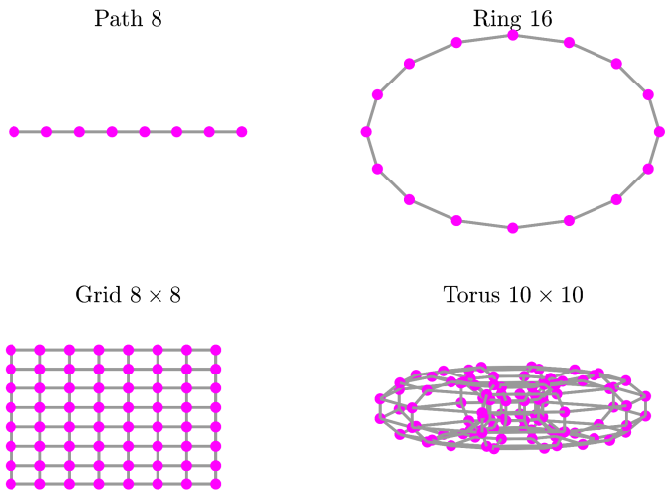


Figure 2.4 – Graphs corresponding to classical Fourier transform. The path and the grid correspond to the DCT. The ring and the torus correspond to the DFT.

The ring graph

The ring graph plays a central role in Graph Signal Processing (GSP) because it allows us to consider some results as extension from the classical framework. More precisely, we will show some equivalence between the ring graph and discrete periodic case. While we index the nodes from 1 to N, we sometimes use the index 0 for the node N.

Graph 6 (Ring graph)**.** *The weight matrix of a "ring" graph is defined as follow:*

$$\boldsymbol{W}[i,n] = \begin{cases} 1 & \text{if } |i-n| = 1 \\ 1 & \text{if } i = 1 \text{ and } n = N \\ 1 & \text{if } i = N \text{ and } n = 1 \\ 0 & \text{otherwise.} \end{cases}$$

It encodes the fact that each vertex is linked to its two neighbors. The circularity is obtained by connecting the first and the last nodes.

In this special case, the degree matrix becomes $\boldsymbol{D} = 2\boldsymbol{I}$ and the Laplacian reads:

$$\boldsymbol{L} = \begin{bmatrix} 2 & 1 & & & 1 \\ 1 & 2 & 1 & & \\ & 1 & \ddots & \ddots & \\ & & \ddots & 2 & 1 \\ 1 & & & 1 & 2 \end{bmatrix} \tag{2.17}$$

Since this matrix is circulant, first the complex exponentials (for $\ell = 0, \ldots, N-1$)

$$\boldsymbol{u}_\ell[n] = \frac{1}{\sqrt{N}} \exp\left(2\pi j \frac{\ell n}{N}\right), \tag{2.18}$$

where $j = \sqrt{-1}$, form a valid set of eigenvectors, and second the corresponding eigenvalues are given by

$$\lambda_k = 2 - 2\cos\left(2\pi \frac{nk}{N}\right). \tag{2.19}$$

Details on this computation can be found in the work of Strang [125]. Since many eigenvalues have multiplicity 2, other eigenvectors systems are possible. Additional insights on this topic can be found in [125, 49, 127].

Eigenvalues reordering

In GSP the convention is to order the eigenvalues in ascending order. In this paragraph, we derive the corresponding formulas and show the effect of this spectral folding. For $\ell = 0, \ldots N-1$, the eigenvalues in increasing order are given by

$$\lambda_\ell = \begin{cases} 2 - 2\cos\left(\frac{\pi \ell}{N}\right) & \text{if } \ell \text{ is even,} \\ 2 - 2\cos\left(\frac{\pi(\ell+1)}{N}\right) & \text{if } \ell \text{ is odd.} \end{cases} \tag{2.20}$$

Except for $\ell = 1$ and for $\ell = N$ (when N is even) all eigenvalues have a multiplicity of 2. Furthermore the associated eigenvectors are the complex exponentials of the Discrete Fourier Transform (DFT). Following the ordering of the graph eigenvalues they are given for $n = 1 \ldots N$ as

$$\boldsymbol{u}_\ell[n] = \begin{cases} \frac{1}{\sqrt{N}} \exp\left(\pi j \frac{\ell n}{N}\right) & \text{if } \ell \text{ is even,} \\ \frac{1}{\sqrt{N}} \exp\left(\pi j \frac{(2N-\ell-1)n}{N}\right) & \text{if } \ell \text{ is odd.} \end{cases} \tag{2.21}$$

Up to a reordering, this is exactly the eigenvector basis of the DFT (2.21). As shown in Figure 2.5, this reordering corresponds to flipping the negative frequencies around the Nyquist frequency. When we use the graph Laplacian operator, we lose the notion of negative and positive frequencies as only their squared modulus matter.

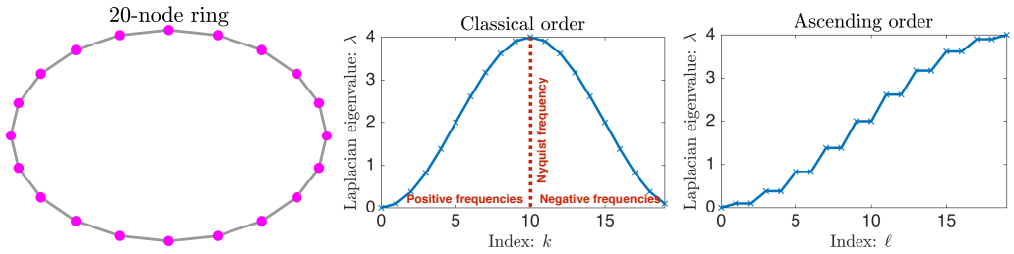


Figure 2.5 – 20-ring graph. Left: one graphical representation. Center: eigenvalues plotted with in classical order, i.e., (2.19). Right: eigenvalues plotted in ascending order, i.e., (2.20). Considering the eigenvalues in ascending order is equivalent to flipping the negative frequencies around the Nyquist frequency.

2.3 Graph filters

The graph Fourier transform plays a central role in GSP since it help us to extend the filtering operations to graph signals. In the classic setting, applying a filter on a signal is done with a convolution, i.e., a point-wise multiplication in the spectral domain. Similarly, filtering a graph signal is defined through a multiplication in the graph Fourier domain. There are two ways to define graph filters: (a) one can simply assign a coefficient to each eigenvalue, or (b) one can define a continuous function $g : \mathbb{R}_+ \to \mathbb{R}$ in the graph Fourier domain and then compute the discrete coefficients by evaluating the function at each eigenvalue. In this work, we mostly work with the second method. For $g : \mathbb{R}_+ \to \mathbb{R}$ and $\boldsymbol{x} \in \mathbb{R}^N$, the filtered signal $\boldsymbol{y}$ satisfies, in the graph spectral domain

$$\hat{\boldsymbol{y}}[\ell] = g(\lambda_\ell) \cdot \hat{\boldsymbol{x}}[\ell], \quad \text{or} \quad \hat{\boldsymbol{y}} = g(\boldsymbol{\Lambda})\hat{\boldsymbol{x}},$$

where $g(\boldsymbol{\Lambda})$ is a diagonal matrix with $g(\boldsymbol{\Lambda})[\ell, \ell] = g(\lambda_\ell)$. In the vertex domain, this expression becomes

$$\boldsymbol{y}[n] = \sum_{\ell=0}^{N-1} g(\lambda_\ell)\hat{\boldsymbol{x}}[\ell]\boldsymbol{u}_\ell[n] = \sum_{i=1}^{N} \boldsymbol{x}[n] \sum_{\ell=0}^{N-1} g(\lambda_\ell)\boldsymbol{u}_\ell^*[i]\boldsymbol{u}_\ell[n], \quad \text{or} \quad \boldsymbol{y} = g(\boldsymbol{L})\boldsymbol{x}, \tag{2.22}$$

where $g(\boldsymbol{L}) = \boldsymbol{U} g(\boldsymbol{\Lambda}) \boldsymbol{U}^*$.

Since small eigenvalues correspond to low frequencies and conversely high eigenvalues to high frequencies, filters can thus be designed to select a frequency band.

Example 5 (Low pass filtering on graph)**.** *Using a low pass filter, we can de-noise a signal on the graph. In Figure 2.6, we present an example of such an operation. Let us suppose that the signal on each node is the value returned by an inaccurate sensor that measures the temperature. Since we know that the temperature varies smoothly in space, we assume that the final temperature distribution on the graph is smooth (low-frequency) as well. To remove the noise, we thus apply a low-pass filter that removes high-frequencies.*

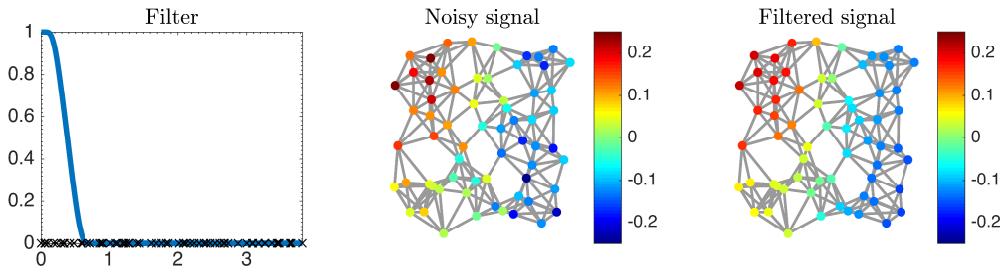


Figure 2.6 – Example of a low pass filtering operation. Fast variation between the nodes are removed while the slow variation are kept.

2.3.1 Filterbanks

Many transforms like the wavelets or the short time Fourier transform are based on multiple filters g_k.

Definition 10 (Graph filter bank)**.** *A* graph filter bank *is a sequence of kernels (or filters)* $\mathrm{g} = \{g_1, g_2, \ldots, g_K\}$*, where each* $g_k : [0, \lambda_{\max}] \to \mathbb{C}$ *is a function defined on the graph Laplacian spectrum* $[0, \lambda_{\max}]$ *of a graph* $\mathcal{G}$*.*

In Section 4, we sometimes index filter banks from 0 to $K-1$. The transforms associated to a filter bank are called *analysis* and *synthesis*. They consist in applying all filters to the signal.

Definition 11 (Analysis operator)**.** *The* analysis operator *of a graph filter bank* $\mathrm{g} = \{g_1, \ldots, g_K\}$ *applied to a graph signal* $\boldsymbol{x} \in \mathbb{C}^N$ *is given by*

$$\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}[\cdot, k] := g_k(\boldsymbol{L})\boldsymbol{x}.$$

Also we use a double indexes notation, $\boldsymbol{c} = \boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}$ is in general considered as a vector $\in \mathbb{C}^{KN}$ which is a redundant representation of $\boldsymbol{x}$. The linear operator $\boldsymbol{A}_{\mathrm{g}}$ can be thought of as a matrix of size $KN \times N$.

Definition 12 (Synthesis operator)**.** *The* synthesis operator *of a graph filter bank* $\mathrm{g} = \{g_1, \ldots, g_K\}$ *applied to a graph signal representation* $\boldsymbol{c} \in \mathbb{C}^{KN}$ *is given by*

$$\boldsymbol{S}_{\mathrm{g}}\boldsymbol{x} = \boldsymbol{A}_{\mathrm{g}}^{\mathsf{T}}\boldsymbol{c} = \sum_{k=1}^{K} g_k(\boldsymbol{L})\boldsymbol{c}[\cdot, k].$$

The synthesis operation is simply the adjoint operator of the analysis. Figure 2.7 shows a simple representation of both operations.

We need a condition to warranty that $\boldsymbol{c} = \boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}$ is a valid representation of $\boldsymbol{x}$, i.e., that no information is lost during the transformation.

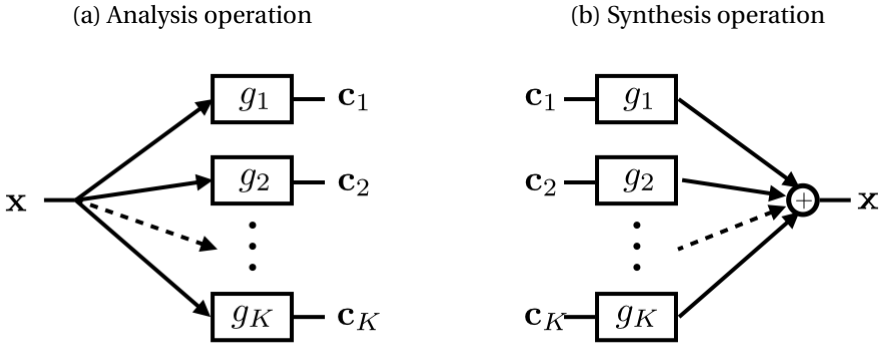


Figure 2.7 – Schematic representation of analysis and synthesis operation

Definition 13 (Graph filter bank frame)**.** *A graph filter bank* $\mathrm{g} = \{g_1, \ldots, g_K\}$ *is a* graph filter bank frame *if there exist constants A and B called the lower and upper frame bounds such that for all* $\boldsymbol{x} \in \mathbb{C}^N$*:*

$$A \|\boldsymbol{x}\|_2^2 \leq \sum_{i,k} |\boldsymbol{A}_{\mathrm{g}} \boldsymbol{x}[i,k]|^2 \leq B \|\boldsymbol{x}\|_2^2.$$

If $A = B$*, the frame is said to be a* graph filter bank tight frame.

In practice, this condition can be easily verified from the filter bank g almost independently from the graph G using the following lemma.

Lemma 1 ([115], Lemma 1)**.** *Let us consider the filter bank* $\mathrm{g} = \{g_1, \ldots, g_K\}$ *and define*

$$G(\lambda) := \sum_{k=0}^{K-1} |g_k(\lambda)|^2, \tag{2.23}$$

then the analysis operator $\boldsymbol{A}_{\mathrm{g}}$ *has lower and upper frame bounds given by*

$$A = \min_{\ell \in 0, \cdots N-1} G(\lambda), \quad \textit{and} \quad B = \max_{\ell \in 0, \cdots N-1} G(\lambda), \tag{2.24}$$

respectively. If $G(\lambda)$ *is constant over* $\lambda_\ell, \forall \ell \in 0, \ldots N-1$ *, then* g *is a* tight frame.

Examples of graph filter bank frames include the spectral graph wavelets of [52], the Meyer-like tight graph wavelet frames of [62, 63], the spectrum-adapted wavelets and vertex-frequency frames of [115], and the learned parametric dictionaries of [128]. The dictionaries constructions in [52, 115] choose the filters so that their energies are localized in different spectral bands. Different choices of filters lead to different tilings of the vertex-frequency space, and can for example lead to wavelet-like frames or vertex-frequency frames (analogous to classical windowed Fourier frames). The frame condition of Lemma 1 ensures that these filters cover the entire spectrum, so that no band of information is lost during analysis and reconstruction.

2.4 Fast filtering via Chebyshev polynomials

The graph filtering operation described above is based on the graph Fourier transform. Unfortunately, the graph Fourier basis needed for performing this transform requires the diagonalization of the graph Laplacian, which takes $O(N^3)$ operations and $O(N^2)$ memory when performed using standard techniques. This is feasible for graphs with only a few thousand vertices. To be able to tackle problems of larger size, more efficient methods are needed. We present here two possible methods of order $O(|\mathcal{E}|)$. The first one using Chebyshev polynomials is presented in [52] and the second one using the Lanczos method is proposed in [126].[4]

2.4.1 Fast filtering via Chebyshev polynomials

Filtering in the vertex domain. To avoid the Fourier transform, we perform the filtering operation in the vertex domain using only the Laplacian operator. Applying this operator corresponds to multiplying the signal in the spectral domain with the eigenvalues:

$$\widehat{\boldsymbol{L}\boldsymbol{x}} = \boldsymbol{\Lambda}\hat{\boldsymbol{x}}.$$

This is equivalent to filtering with $g(\lambda) = \lambda$. Using this relation recursively and exploiting linearity, we can apply any polynomial filter $g(\lambda) = a_0 + a_1\lambda + \cdots + a_K\lambda^M$ to a signal $\boldsymbol{x}$ with the following formula:

$$\boldsymbol{x}' = \boldsymbol{U}g(\boldsymbol{\Lambda})\boldsymbol{U}^*\boldsymbol{x} = \left(a_0\boldsymbol{I} + a_1\boldsymbol{L} + \cdots + a_M\boldsymbol{L}^M\right)\boldsymbol{x}. \tag{2.25}$$

Chebyshev polynomial approximation. The ability to apply polynomial filters efficiently suggests to approximate a given filter function with a suitable polynomial. For approximating functions on real intervals, Chebyshev polynomials are usually the preferred choice because of numerical stability considerations and the fact that they can be evaluated efficiently by three-term recurrences. We refer to [52] for a more detailed discussion on the choice of Chebyshev polynomials in signal-processing on graphs and to, e.g., [93] for an introduction to polynomial approximation.

The m^{th} Chebyshev polynomial $P_m(y)$ is generated using the recurrence relation

$$P_m(y) = 2yP_{m-1}(y) - P_{m-1}(y)$$

with $P_0(y) = 1$ and $P_1(y) = y$. For $y \in [-1,1]$, these polynomials possess the following well-known properties:

1. they admit the closed form expression $P_m(y) = \cos(m\arccos(y))$;
2. they are bounded, i.e., $P_m(y) \in [-1,1]$;

[4] Some parts of this section are slightly adapted version of [126].

3. they form an orthogonal basis of $L^2\left([-1,1], \frac{\mathrm{d}y}{\sqrt{1-y^2}}\right)$.

The third property implies that every function $h \in L^2\left([-1,1], \frac{\mathrm{d}y}{\sqrt{1-y^2}}\right)$ admits a convergent Chebyshev series

$$h(y) = \frac{1}{2}c_0 + \sum_{m=1}^{\infty} c_m P_m(y),$$

with the Chebyshev coefficients

$$c_k = \frac{2}{\pi}\int_{-1}^{1} \frac{P_m(y)h(y)}{\sqrt{1-y^2}}\mathrm{d}y = \frac{2}{\pi}\int_0^{\pi} \cos(k\theta)h(\cos(\theta))\mathrm{d}\theta.$$

Since our filter g is evaluated on the eigenvalues of the Laplacian, we need to map the interval $[-1,1]$ to the interval $[0,\lambda_{\max}]$ using the transformation $\lambda = \frac{\lambda_{\max}}{2}(y+1)$. Defining $\tilde{T}_m(\lambda) = T_m\left(\frac{2\lambda}{\lambda_{\max}} - 1\right)$ we obtain

$$g(\lambda) = \frac{1}{2}c_0 + \sum_{m=1}^{\infty} c_m \tilde{T}_m(\lambda) \tag{2.26}$$

for $\lambda \in [0,\lambda_{\max}]$, with

$$c_m = \frac{2}{\pi}\int_0^{\pi} \cos(m\theta)g\left(\frac{\lambda_{\max}}{2}(\cos(\theta)+1)\right)\mathrm{d}\theta.$$

Fast filtering algorithm. At this point, we can derive the iterative algorithm for filtering a signal $\boldsymbol{x}$ with g. The recurrence relation for the transformed Chebyshev polynomials becomes $\tilde{P}_m(\lambda) = 2\left(\frac{2x}{\lambda_{\max}} - 1\right)\tilde{P}_{m-1}(\lambda) - \tilde{P}_{m-2}(\lambda)$. On the matrix level, this yields, using (2.25):

$$\tilde{P}_m(\boldsymbol{L})\boldsymbol{x} = 2\left(\frac{2\boldsymbol{L}}{\lambda_{\max}} - \boldsymbol{I}\right)\tilde{P}_{m-1}(\boldsymbol{L})\boldsymbol{x} - \tilde{P}_{m-2}(\boldsymbol{L})\boldsymbol{x}.$$

Combined with (2.26), this finally leads to the following expression for filtering a signal $\boldsymbol{x}$:

$$\boldsymbol{x}' = g(\boldsymbol{L})\boldsymbol{x} = \frac{1}{2}c_0\boldsymbol{I}\boldsymbol{x} + \sum_{m=1}^{\infty} c_m\tilde{P}_m(\boldsymbol{L})\boldsymbol{x}.$$

When implemented, we truncate this sum at a defined order M. Assuming that $|\mathcal{E}| > N$, the computational cost of this algorithm scales linearly with the number of edges $O(M|\mathcal{E}|)$. In most applications, the Laplacian is sparse, $|\mathcal{E}| \ll N^2$, which results in a fast algorithm. Moreover, apart from storing the Laplacian, the additional memory consumed by this algorithm is only $4N$.

2.4.2 Accelerated filtering using Lanczos

A direct polynomial approximation is not the only way to approximate a graph filter. Given the graph Laplacian $\boldsymbol{L} \in \mathbb{R}^{N\times N}$ and a nonzero vector $\boldsymbol{x} \in \mathbb{R}^N$, the Lanczos method [48] shown in Algorithm 1 below computes an orthonormal basis $\boldsymbol{V}_M = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M]$ of the Krylov subspace $K_M(\boldsymbol{L}, \boldsymbol{x}) = \operatorname{span}\{\boldsymbol{x}, \boldsymbol{L}\boldsymbol{x}, \ldots, \boldsymbol{L}^{M-1}\boldsymbol{x}\}$. The computational cost of Algorithm is $O(M \cdot |\mathcal{E}|)$. The storage of the basis $\boldsymbol{V}_M$ requires MN additional memory, which can be avoided using two passes of the algorithm or restart techniques; see, e.g., [41] for more details.

Algorithm 1 Lanczos method

Input: Symmetric matrix $\boldsymbol{L} \in \mathbb{R}^{N\times N}$, vector $\boldsymbol{x} \neq \boldsymbol{0}$, $M \in \mathbb{N}$.
Output: $\boldsymbol{V}_M = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_M]$ with orthonormal columns, scalars $\alpha_1, \ldots, \alpha_M \in \mathbb{R}$ and $\beta_2, \ldots, \beta_M \in \mathbb{R}$.

$\boldsymbol{v}_1 \leftarrow \boldsymbol{x} / \|\boldsymbol{x}\|_2$
for $j = 1, 2, \ldots, M$ **do**
  $\boldsymbol{w} = \boldsymbol{L}\boldsymbol{v}_j$
  $\alpha_j = \boldsymbol{v}_j^* \boldsymbol{w}$
  $\tilde{\boldsymbol{v}}_{j+1} = \boldsymbol{w} - \boldsymbol{v}_j \alpha_j$
  if $j > 1$ **then**
    $\tilde{\boldsymbol{v}}_{j+1} \leftarrow \tilde{\boldsymbol{v}}_{j+1} - \boldsymbol{v}_{j-1}\beta_{j-1}$
  end if
  $\beta_j = \left\|\tilde{\boldsymbol{v}}_{j+1}\right\|_2$
  $\boldsymbol{v}_{j+1} = \tilde{\boldsymbol{v}}_{j+1} / \beta_j$
end for

The scalars produced by Algorithm 1 can be arranged into a symmetric tridiagonal matrix $\boldsymbol{H}_M \in \mathbb{R}^{M\times M}$ satisfying

$$\boldsymbol{V}_M^* \boldsymbol{L} \boldsymbol{V}_M = \boldsymbol{H}_M = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \beta_M \\ & & & \beta_M & \alpha_M \end{bmatrix}.$$

In floating-point arithmetics, the orthogonality of the basis produced by Algorithm 1 may get quickly lost and reorthogonalization is needed [27].

Given a continuous function $g : [0, \lambda_{\max}] \to \mathbb{R}$ and a vector $\boldsymbol{s}$, the following approximation to $g(\boldsymbol{L})\boldsymbol{x}$ was proposed by Gallopoulos and Saad in [43]:

$$g(\boldsymbol{L})\boldsymbol{x} \approx \|\boldsymbol{x}\|_2 \, \boldsymbol{V}_M g(\boldsymbol{H}_M) \boldsymbol{e}_1, \tag{2.27}$$

where $\boldsymbol{e}_1 \in \mathbb{R}^M$ is the first unit vector. Because of eigenvalue interlacing, the eigenvalues of $\boldsymbol{H}_M$ are contained in the interval $[0, \lambda_{\max}]$ and hence the expression $g(\boldsymbol{H}_M)$ is well-defined.

Typically, we have $M \ll N$, making the evaluation of $g(\boldsymbol{H}_M)$ inexpensive. The overall cost of our Lanczos-based approximation of graph-signal filtering, which consists of applying Algorithm 1 and evaluating (2.27), is therefore between $O(M \cdot |\mathcal{E}|)$ and $O\big(M \cdot |\mathcal{E}| + M^2 N\big)$, depending on how the reorthogonalization is performed.

3 Structural clustering via the graph localization operator

Convolution, stationarity, and many important transforms rely on the translation operator "that shifts a signal without changing its shape." For example, the wavelet transform is a projection onto a collection of translated and scaled versions of a mother function.

In the effort to develop GSP tools, there have been multiple attempts to generalize translation [47, 117, 72]. As we show in Appendix B.1, none of them perform what one would naturally expect for a translation, i.e., "moving some mass centered around one node to another while preserving its global shape." We believe that searching for a natural, well-defined translation operator for graphs is not the best solution to tackle GSP problems as they do not require all the properties reminiscent of the translation operator.

Instead, we can use a weaker version of translation that we call *the graph localization operator* and that has the property to localize a kernel around a selected graph vertex while conserving some global shape properties. In opposition to the classical translation both the signal shape and the signal energy are not preserved. In this thesis, we value this behavior as it exhibits an adaptation to the irregular structure of the graph and we believe that it should be leveraged for the best.

The localization operator has many important implications in GSP and will in practice replace translation. As we shall see, a) graph filters (Section 2.3) benefit from an alternative interpretation, b) localization can be used to extract local properties of the graph structure (this section), c) it plays an important role in graph uncertainty principles (Section 4), d) it naturally extends the notion of stationarity for graphs (Section 5), and e) it helps in the problem of sampling (Example 20).

3.1 The localization operator

There have been multiple generalizations of the translation operator for graphs. They are detailled in Appendix B.1. We concentrate here on the generalized translation for graph signals defined by Shuman et al.. Leveraging the graph Fourier transform, they define translation as

the convolution with a Kronecker delta [117, Equation 26]. The graph convolution $*$ being an element-wise multiplication in the spectral domain.

Definition 14. *For a graph signal $\boldsymbol{x}$ and a vertex i, the generalized graph translation operator reads:*

$$\boldsymbol{T}_i \boldsymbol{x}[n] := (\boldsymbol{x} * \boldsymbol{\delta}_i)[n] = \sum_{\ell=0}^{N-1} \hat{\boldsymbol{x}}[\ell] \boldsymbol{u}_\ell^*[i] \boldsymbol{u}_\ell[n]. \tag{3.1}$$

Unfortunately, the generalized translation operator does not perform what we would intuitively expect from it, i.e., it does not translate a signal $\boldsymbol{x}$ from node n to node i. Instead when $\hat{\boldsymbol{x}}$ changes smoothly across the frequencies (Theorem 3), then $\boldsymbol{T}_i \boldsymbol{x}$ is localized around node i, while $\boldsymbol{x}$ is in general not localized at a particular node or set of nodes.

The localization operator is very similar to the generalized translation but applies to a kernel defined in the graph spectral domain.

Definition 15. *Let $\mathcal{C}$ be the set of functions $\mathbb{R}^+ \to \mathbb{R}$. For a graph kernel $g \in \mathcal{C}$ (defined in the spectral domain) and a node i, the localization operator $\mathcal{T}_i^G : \mathcal{C} \to \mathbb{R}^N$ reads:*

$$\mathcal{T}_i^G g[n] := \sum_{\ell=0}^{N-1} g(\lambda_\ell) \boldsymbol{u}_\ell^*[i] \boldsymbol{u}_\ell[n] = (g(\boldsymbol{L})\boldsymbol{\delta}_i)[n] = g(\boldsymbol{L})[i, n]. \tag{3.2}$$

Here we use the calligraphic notation $\mathcal{T}_i^G$ to differentiate with all the translation operators $\boldsymbol{T}_i$. We first observe from (3.2) that the i^{th} line of graph filter matrix $g(\boldsymbol{L})$ is the kernel g localized at node i. Intuitively, it means

$$\big(g(\boldsymbol{L})\boldsymbol{x}\big)[i] = \langle \boldsymbol{x}, \mathcal{T}_i^G g \rangle. \tag{3.3}$$

Instead of a kernel, we could replace $g(\lambda_\ell)$ by $\hat{\boldsymbol{x}}[\ell]$ in Definition 15 and localize the discrete vector $\hat{\boldsymbol{x}}$ instead. In this case, we would match the generalized translation closely (Definition 14). Nevertheless, we prefer to work with a kernel for three reasons. 1) In practice when the graph is large, the Fourier basis cannot be computed making it impossible (or at least complicated) to localize a vector. On the other side, for a kernel g, there are techniques to approximate $\mathcal{T}_i^G g = g(\boldsymbol{L})\boldsymbol{\delta}_i$ (see Section 2.4). 2) The localization properties are theoretically easier to interpret when g is a filter. Let us suppose that g is a K order polynomial, then the support of $\mathcal{T}_i^G g$ is contained exactly in a ball of radius K centered at node i. Building on this idea, for a sufficiently regular function g, it has been proved in [117, Theorem 1 and Corollary 2] that the localization operator concentrates the kernel g around the vertex i. 3) Using a kernel ensures some stability when dealing with graph having eigenvalue multiplicity greater than 1.

3.1.1 Links with the generalized translation operator

Let us now clarify how generalized translation and localization are linked. The main difference between these two operators is the domain on which they are applied. Whereas, the translation operator acts on a discrete signal defined in the time or the vertex domain, the localization operator requires a continuous kernel or alternatively a discrete signal in the spectral domain. Both return a signal in the vertex domain. To summarize, in the classical signal processing framework, the localization operator could be seen as computing the inverse Fourier transform first and then translating the signal. For graphs, it is an operator that takes a filter from the spectral domain and localizes it at a given node i while adapting it to the graph structure. Some extra relations between these operators can be found in Appendix B.2.

3.1.2 Properties of the localization operator

In order to provide further insights on the localization operator, let us review some of its properties.

Basis independence

In many cases, the graph Fourier basis is not uniquely defined. (It happens whenever an eigenvalue has a multiplicity greater than 1.) Conveniently, it turns out that the localization operator is invariant with respect of the basis choice.

Theorem 1. *For any graph $\mathcal{G}$ and any kernel $g : \mathbb{R}_+ \to \mathbb{R}$ such that $\forall \ell, |g(\lambda_\ell)| < \infty$, the localization operator is independent of the eigenvector basis.*

The proof is given in Appendix B.3. As illustrated in Figure 3.1, this property is essential as it is a warranty that the localization operator respects graph symmetry, i.e., if two vertices are symmetric in a graph (automorphism), the result of the localization operation will also be symmetric.[1]

Generalization of the translation

In the case of a ring graph (see Graph 6), the localization operator is closely linked to the classical translation operator of equation (B.1).

Theorem 2. *Let $\mathcal{G}$ be a ring graph of N vertices, for a kernel $g : \mathbb{R}_+ \to \mathbb{R}$ such that $\forall \ell, |g(\lambda_\ell)| < \infty$, the localization operator $T_i^G g$ satisfies the following properties:*

[1] Given $\mathcal{G}'$, an automorphism of graph $\mathcal{G}$, two nodes $v_i(\mathcal{G})$ and $v_n(\mathcal{G})$ are said to be symmetric if $v_i(\mathcal{G}) = v_n(\mathcal{G}')$. Given a permuation matrix $\boldsymbol{P}$ that transform $\mathcal{G}$ in $\mathcal{G}'$, we say that $\mathcal{G}'$ is an automorphism of $\mathcal{G}$ if $\boldsymbol{PLP}^* = \boldsymbol{L}$. A signal $\boldsymbol{x}$ is symetric if $\boldsymbol{x} = \boldsymbol{Px}$.

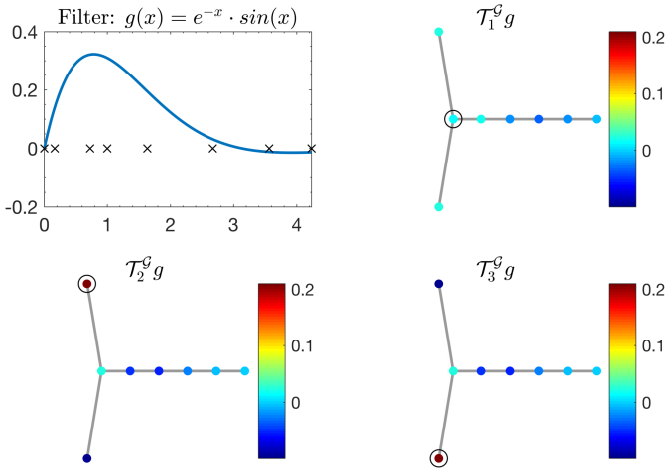


Figure 3.1 – The localization operator preserves vertex symmetry. The kernel g is localized at vertex 1,2 and 3. Because of the graph symmetry $\mathcal{T}_2^G g$ is the mirrored version of $\mathcal{T}_3^G g$. Furthermore $\mathcal{T}_2^G g[i] = \mathcal{T}_3^G g[i]$ for $i \neq 2,3$.

1. *The localized atoms are translated version of each other, i.e.,* $\forall i, n \in \{1 \dots N\}$,

$$\mathcal{T}_i^G g[n] = \mathcal{T}_N^G g[n-i]. \tag{3.4}$$

2. *The localized atoms are always real.*

3. $\mathcal{T}_i^G g[n]$ *is symmetric with respect to vertex* i*, i.e.,*

$$\mathcal{T}_i^G g[n-k] = \mathcal{T}_i^G g[n+k] \forall k. \tag{3.5}$$

4. *For any symmetric* $\boldsymbol{x}$*, a kernel* g *exists such that* $\boldsymbol{x} = \mathcal{T}_N^G g$.

The proof is given in Appendix B.4.

Localization properties

In many cases, the localization operator produces a graph signal that has energy concentrated in a small region of the graph. For example, let us suppose that the kernel g is a polynomial of order K. Then, the filter associated to g can be written as

$$g(\boldsymbol{L}) = a_0 \boldsymbol{I} + a_1 \boldsymbol{L} + a_2 \boldsymbol{L}^2 + \dots + a_K \boldsymbol{L}^K \tag{3.6}$$

Intuitively, the atom $\mathcal{T}_i^G g[n] = g(\boldsymbol{L})[i,n]$ is strictly localized in a ball of radius K around the vertex i [52, Section 5.2].

Based on this simple idea, Shuman et al. [117, Section 4.4] have shown that the smoothness[2] of the kernel g is linked to the localization property of this kernel; i.e., if a smooth kernel g is localized to center vertex i, then the magnitude of $\mathcal{T}_i^G g[n]$ decays as the hop distance (Definition 6) $h_\mathcal{G}$ between i and n increases.

It is quantified by two theorems that are slightly adapted in order to match the framework used in this thesis. The first theorem states that the decay rate of the localization operator is bounded by the approximation rate of the kernel g with a polynomial p_K.

Theorem 3 (Theorem 1 [117])**.** [3] *Let $g : [0, \lambda_{\max}] \to \mathbb{R}$ be a kernel and set $K_{in} = h_\mathcal{G}(i,n) - 1$. Then for $r, q \geq 1$ such that $\frac{1}{r} + \frac{1}{q} = 1$,*

$$\left|\mathcal{T}_i^G g[n]\right| \leq \mu_\mathcal{G}^{2\frac{q-1}{q}} \inf_{p_{K_{in}}} \left\|g - p_{K_{in}}\right\|_r, \tag{3.7}$$

where $\mu_\mathcal{G} = \max_{\ell,n} |\boldsymbol{u}_\ell[n]|$, the infimum is taken over all polynomials $p_{K_{in}}$ of order K_{in} and

$$\left\|g - p_{K_{in}}\right\|_r = \left(\sum_{\ell=0}^{N-1} \left(g(\lambda_\ell) - p_{K_{in}}(\lambda_\ell)\right)^r\right)^{\frac{1}{r}}. \tag{3.8}$$

Theorem 3 guaranties localization providing that the polynomial coefficients are decaying at a sufficient rate. Furthermore, when the kernel g is differentiable, the second and next Theorem develops the bound by introducing explicitly the error of a Chebyshev approximation.

Theorem 4 (Corollary 2 [117])**.** *Given two nodes v_i and v_n separated by the hop distance $h_\mathcal{G}(v_i, v_n)$, If g is $h_\mathcal{G}(v_i, v_n)$-times continuously differentiable on $[0, \lambda_{\max}]$, then*

$$\left|\mathcal{T}_i^G g[n]\right| \leq \left[\frac{2}{h_\mathcal{G}(i,n)!|g(0)|}\left(\frac{\lambda_{\max}}{4}\right)^{h_\mathcal{G}(v_i,v_n)}\right] \sup_{\lambda\in[0,\lambda_{\max}]} \left|g^{(h_\mathcal{G}(v_i,v_n))}(\lambda)\right|. \tag{3.9}$$

Theorem 4 provides an important insight on localization. Let us scale the kernel g by defining $g_a(x) = g(ax)$. In this case, the derivatives are given by $g_a^{(k)}(x) = a^k g^{(k)}(ax)$ and the localization bound becomes

$$\left|\mathcal{T}_i^G g[n]\right| \leq \left[\frac{2}{h_\mathcal{G}(v_i,v_n)!|g(0)|}\left(\frac{a\lambda_{\max}}{4}\right)^{h_\mathcal{G}(i,n)}\right] \sup_{\lambda\in[0,\lambda_{\max}]} \left|g^{(h_\mathcal{G}(v_i,v_n))}(\lambda)\right|. \tag{3.10}$$

If $a < 1$, the values of the k^{th} derivative decreases by a^k and the global smoothness of the kernel is improved. As a result, the bound given in (3.10) diminishes and $\mathcal{T}_i^G g$ is more localized. The relation (3.10) is an uncertainty principle that quantifies the tradeoff between localization in the spectral and in the vertex domains.

Faber polynomials can also be used to improve the localization bound. While this is beyond

[2] The polynomial coefficients of a smooth function have a rapid decay.

[3] The missing $\sqrt{N}$ term comes from differences between the definitions.

the scope of this thesis, we note that [12, Theorems 3.4 and 3.7] can be adapted to the GSP framework. Important results are also available in [13, 11]

Example 6 (Disconnected node/component)**.** *As a direct implication of Theorem 4, we can treat the case of an isolated node. Let us suppose that the vertex i is disconnected from the graph. Then the localization operator perfectly localizes the kernel on the vertex i as:*

$$\mathcal{T}_i^G g = g(0)\boldsymbol{\delta}_i.$$

More generally, if a graph $\mathcal{G}$ is made of two disconnected components $\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1, \mathcal{W}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2, \mathcal{W}_2)$, then $\forall v_i \in \mathcal{V}_1$, the support of the localization operator remains on $\mathcal{V}_1$, i.e.,

$$\mathcal{T}_i^G g[n] = 0, \quad \forall v_n \in \mathcal{V}_2. \tag{3.11}$$

Preservation of the frequency content

The shape of the localized kernel cannot be described trivially as it relies completely on the graph structure. One helpful picture is to consider the localization operator as a filtering operation with a Kronecker, i.e.,

$$\mathcal{T}_i^G g = g(\boldsymbol{L})\boldsymbol{\delta}_i. \tag{3.12}$$

As illustrated in Example 7, the graph frequency content of the Kronecker $\boldsymbol{\delta}_i$ depends on the structure around the vertex i and is, in general, not identical between two different vertices. As a result, the localized kernels may have significantly different shapes depending on where it is localized on the graph. Nevertheless, if the graph has a regular structure, a global behavior is more or less preserved.

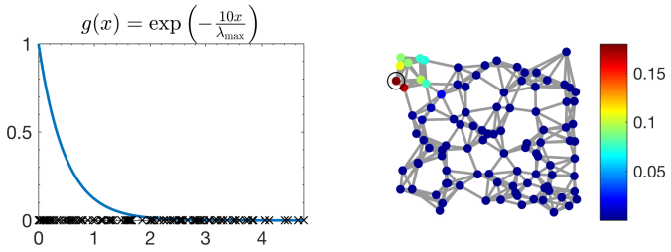
Example 7. *Figure 3.2 shows an example of localization of two kernels: a) the heat kernel and b) the Mexican hat wavelet. The shape of the localized kernels adapts to the graph topology. In the case of the heat kernel, it simply corresponds to the result of diffusing a unit of heat along the graph edges. In the case of the Mexican hat, we also observe that the general shape of the wavelet is preserved. It has large positive values around the node where it is localized. It then goes negative a few nodes further away and stabilizes at zero for nodes far away. To summarize, the localization operator preserves the global behavior of the filter while adapting to the graph topology.*

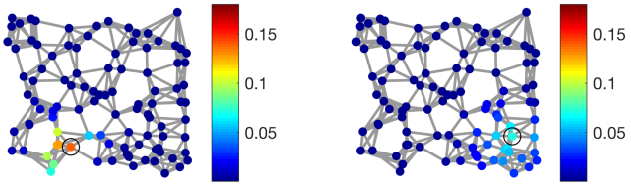
3.1.3 Norm of the localized atoms

Except for special cases (such as when $\mathcal{G}$ is a circulant graph[4] with $\mu_{\mathcal{G}} = \frac{1}{\sqrt{N}}$ and the Laplacian eigenvectors form the DFT basis) the localization operator of Definition 15 is not isometric.

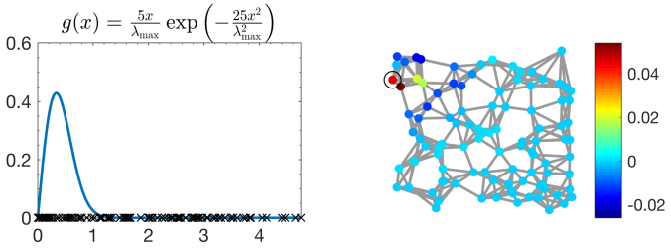
[4] a graph with a circulant weight matrix

(a) Heat kernel





(b) Mexican hat wavelet



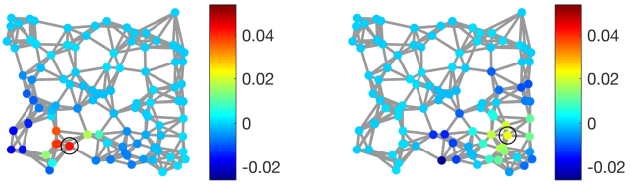


Figure 3.2 – Shape of localized kernels. The kernels are localized around three different vertices (highlighted by a black circle). We observe that the shape of the localized kernels adapts to the graph structure.

Lemma 2 ([117], Lemma 1)**.** *For any* $g \in \mathbb{C}^N$,

$$\frac{|\hat{g}(0)|}{\sqrt{N}} \leq \left\| \mathcal{T}_i^G g \right\|_2 \leq \nu_i \left\| \hat{g} \right\|_2 \leq \mu_{\mathcal{G}} \left\| \hat{g} \right\|_2, \tag{3.13}$$

where $\nu_i = \max_\ell |\boldsymbol{u}_\ell[i]|$. *It yields to the following upper bound on the operator norm of* $\mathcal{T}_i^G$*:*

$$\left\| \mathcal{T}_i^G \right\|_{op} = \sup_g \frac{\left\| \mathcal{T}_i^G g \right\|_2}{\left\| g(\boldsymbol{\lambda}) \right\|_2} \leq \nu_i \leq \mu_{\mathcal{G}},$$

It is interesting to note that although the norm is not preserved when a kernel is localized on an arbitrary graph, it is preserved on average when translated separately to every vertex on the

graph:

$$\sum_{i=1}^{N} \left\| \mathcal{T}_i^G g \right\|_2^2 = \sum_{i=1}^{N} \sum_{\ell=0}^{N-1} \left| g(\lambda_\ell) \bar{\boldsymbol{u}}_\ell[i] \right|^2 = \sum_{\ell=0}^{N-1} \left| g(\lambda_\ell) \right|^2 \sum_{i=1}^{N} |\bar{\boldsymbol{u}}_\ell[i]|^2 = \left\| g(\boldsymbol{\lambda}) \right\|_2^2. \tag{3.14}$$

The following example presents more precise insights on the interplay between the localization operator, the graph structure, and the concentration of localized functions.

Example 8. *Figure 3.3 illustrates the effect of the graph structure on the norms of localized functions. We take the kernel to be localized to be a heat kernel of the form $g(\lambda_\ell) = e^{-\tau\lambda_\ell}$, for some constant $\tau > 0$. We localize the kernel g to be centered at each vertex i of the graph with the operator $\mathcal{T}_i^G$, and we compute and plot their ℓ^2-norms $\left\| \mathcal{T}_i^G g \right\|_2$. The figure shows that when a center node i and its surrounding vertices are relatively weakly connected, the ℓ^2-norm of the localized heat kernel is large, and when the nodes are relatively well-connected, the norm is smaller. Therefore, the norm of the localized heat kernel may be seen as a measure of vertex centrality. In fact, the square norm of the localized heat kernel at vertex i is, up to constants, the average diffusion distance from i to all other vertices. It is therefore a genuine measure of centrality. Moreover, in the case of the heat kernel, we can relate the ℓ^2-norm of $\mathcal{T}_i^G g$ to its concentration*[5] *$\frac{\left\| \mathcal{T}_i^G g \right\|_2}{\left\| \mathcal{T}_i^G g \right\|_1}$. Localized heat kernels are comprised entirely of nonnegative components; i.e., $\mathcal{T}_i^G g[n] \geq 0$ for all i and n. This property comes from (i) the fact that $\mathcal{T}_i^G g[n] = \big(g(\boldsymbol{L})\big)_{in}$, and (ii) the non-trivial property that the entries of $g(\boldsymbol{L})$ are always nonnegative for the heat kernel [75]. Since $\mathcal{T}_i^G g[n] \geq 0$ for all i and n, we have*

$$\left\| \mathcal{T}_i^G g \right\|_1 = \sum_{n=1}^{N} \mathcal{T}_i^G g[n] = g(0) = 1, \tag{3.15}$$

where the second equality follows from [117, Corollary 1]. We can combine (3.13) *and* (3.15) *to derive an upper bound on the concentration of $\mathcal{T}_i^G g$:*

$$\frac{\left\| \mathcal{T}_i^G g \right\|_2}{\left\| \mathcal{T}_i^G g \right\|_1} = \left\| \mathcal{T}_i^G g \right\|_2 \leq \nu_i \left\| g(\boldsymbol{\lambda}) \right\|_2.$$

Thus, $\left\| \mathcal{T}_i^G g \right\|_2$ serves as a measure of concentration, and according to the numerical experiments of Figure 3.3, localized heat kernels centered on the relatively well-connected regions of a graph tend to be less concentrated than the ones centered on relatively less well-connected areas. Intuitively, the values of the localized heat kernels can be linked to the diffusion of a unit of energy from the center vertex to surrounding vertices over a fixed time. In the well-connected regions of the graph, energy diffuses faster, making the localized heat kernels less concentrated.

In Example 8, we have suggested that the norm of the localized heat kernel at vertex v_i is related to the global connectivity in its neighborhood. It turns out that using different kernels,

[5] In Section 4.1.4, we detail the relation between the norms and the concentrations measures.

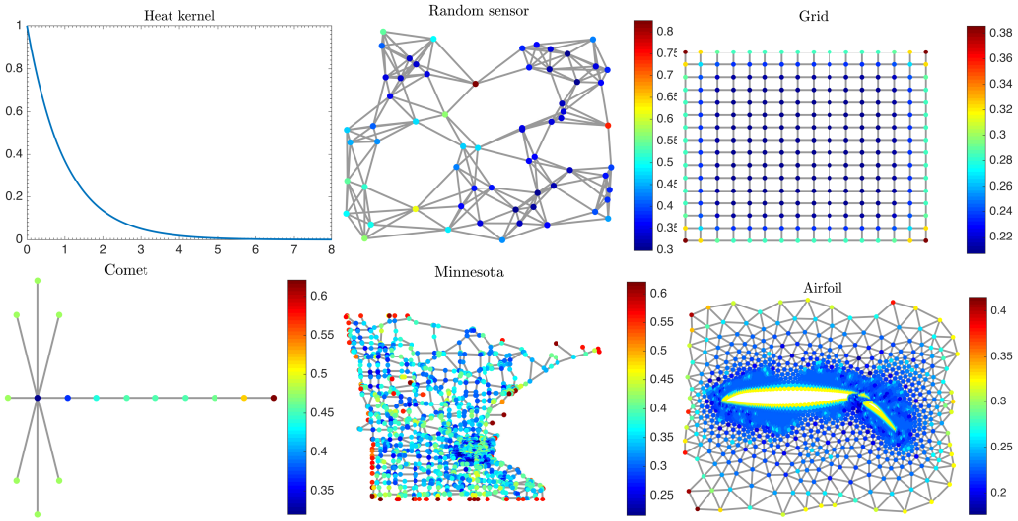


Figure 3.3 – The heat kernel $g(\lambda_\ell) = e^{-10\frac{\lambda_\ell}{\lambda_{\max}}}$ (upper left), and the norms of the localized heat kernels, $\left\{\left\|\mathcal{T}_i^G g\right\|_2\right\}_{i=1,2,\ldots,N}$, on various graphs. For each graph and each center node i, the color of vertex i is proportional to the value of $\left\|\mathcal{T}_i^G g\right\|_2$. Within each graph, the nodes i that are relatively less connected to their neighborhood seem to yield a larger norm $\left\|\mathcal{T}_i^G g\right\|_2$.

we can extract additional structural node properties.

3.2 Graph spectrograms

Traditional Fourier eigenvectors are completely un-localized, i.e., they span the entire domain with a constant modulus. On the contrary, when it comes to the graph Laplacian, some of the energy eigenvectors can be concentrated onto a small graph region or even onto a single vertex, i.e., their energy is concentrated in a region of the graph. In this case, the nodes do not contain all frequencies equally. The localization operator gives us the intuition that the frequency content of a node is highly related to the graph structure around it. Based on this idea, we propose a new graph analysis tool that captures the graph structure locally.

To this end, the non-isometric property of the localization operator is not to be seen as a curse. On the contrary, it can be used to extract the frequency content of a node. Let us suppose that the kernel g is well concentrated around a few frequencies. It turns out that the norm of the localization operator can be used to extract the local graph frequency availability:

$$\left\|\mathcal{T}_i^G g\right\|_2^2 = \sum_{\ell=0}^{N-1} g^2(\lambda_\ell)|\boldsymbol{u}_\ell|^2[i] = \langle g^2(\boldsymbol{\lambda}), |\boldsymbol{u}_\cdot|^2[i]\rangle. \tag{3.16}$$

To summarize, the norm of the localization operator can be used to access graph eigenvector localization.

3.2.1 Definition

Let us build a transform that assigns to each vertex a feature vector/function characterizing its spectral properties.

Definition 16 (Graph spectrogram)**.** *Given a graph $\mathcal{G}$ and a kernel g, the graph spectrogram is defined for $\mu \in \mathbb{R}$ as*

$$\mathcal{S}_{\mathcal{G}}(i,\mu) = \left\| \mathcal{T}_i^G g(\cdot - \mu) \right\|_2^2 = \sum_{\ell=0}^{N-1} g^2(\lambda_\ell - \mu) |\boldsymbol{u}_\ell|^2[i]. \tag{3.17}$$

The graph spectrogram satisfies a few interesting properties.

1. It provides for each vertex the same energy, i.e.,

$$\int_{\mathbb{R}} \mathcal{S}_{\mathcal{G}}^2(i,\mu)\mathrm{d}\mu = \sum_{\ell=0}^{N-1} |\boldsymbol{u}_\ell|^2[i] \int_{\mathbb{R}} g^2(\lambda_\ell - \mu)\mathrm{d}\mu = \|g\|_2^2 \sum_{\ell=0}^{N-1} |\boldsymbol{u}_\ell|^2[i] = \|g\|_2^2 \tag{3.18}$$

2. If the kernel g is chosen to be concentrated around 0, then the graph spectrogram estimates the frequency content of the graph nodes. Indeed, from (3.16), we know that $\mathcal{S}_{\mathcal{G}}(i,\mu)$ will be large if the node i contains some frequency around μ.
3. If the kernel g is chosen to be normalized ($\int_{\mathbb{R}} g^2(x)\mathrm{d}x = 1$) and concentrated around 0, then the cumulative graph spectrogram $\sum_i \mathcal{S}_{\mathcal{G}}^2(i,\mu)$ provides an estimator of the eigenvalue density around μ:

$$\sum_{i=1}^{N} \mathcal{S}_{\mathcal{G}}^2(i,\mu) = \sum_{i=1}^{N} \sum_{\ell=0}^{N-1} g^2(\lambda_\ell - \mu) |\boldsymbol{u}_\ell|^2[i] = \sum_{\ell=0}^{N-1} g^2(\lambda_\ell - \mu) = \left\| g(\boldsymbol{\lambda} - \mu \mathbf{1}) \right\|_2^2. \tag{3.19}$$

4. Using the combinatorial Laplacian, if the weights of a graph are scaled by a constant factor, then the frequency axis of the graph spectrogram is equivalently scaled, i.e., let $\mathcal{G}_1(\mathcal{V},\mathcal{E},\mathcal{W})$ and $\mathcal{G}_2(\mathcal{V},\mathcal{E},a\mathcal{W})$, then

$$\mathcal{S}_{\mathcal{G}_2}(i,\mu) = \mathcal{S}_{\mathcal{G}_1}(i,a\mu) \tag{3.20}$$

This last property indicates the importance of the scale of the edges weights, especially, when one uses graph spectrograms to compare different graphs.

3.2.2 Parameter selection and efficient computation

Discretization

In practice, the graph spectrogram is computed only at K equally spaced values between 0 and $\lambda_{\max}$.

Definition 17. *Given $k = 1,\ldots,K$, let us denote $\mu_k = \frac{(k-1)\lambda_{\max}}{K-1}$ and define the window $g_k(x) =$*

$g(x-\mu_k)$. *The discretized graph spectrogram is defined as*

$$\boldsymbol{S}_{\mathcal{G}}[i,k] = \mathcal{S}_{\mathcal{G}}(i,\mu_k) = \left\| \mathcal{T}_i^G g_k \right\|_2^2 \tag{3.21}$$

Because of the discretization, the mass associated with each node may not be constant (i.e., $\sum_k \boldsymbol{S}_{\mathcal{G}}[i,k] \neq c, \forall i$). One solution is to design the mother kernel g such that the set of filter $\{g_k(\boldsymbol{L})\}_{k=1,\dots,K}$ forms a tight frame, i.e., from Lemma 1 we need $\forall \lambda \in [0, \lambda_{\max}]$

$$\sum_{k=1}^{K} g_k(\lambda) = \sum_{k=1}^{K} g^2\left(\lambda - \frac{(k-1)\lambda_{\max}}{K-1}\right) = A. \tag{3.22}$$

When a tight filter bank is used, then the mass of the discretized graph is constant over the vertex set, i.e.,

$$\sum_k \boldsymbol{S}_{\mathcal{G}}[i,k] = A, \quad \forall v_i \in \mathcal{V}. \tag{3.23}$$

a property resembling (3.17).

Itersine construction

There exists different constructions of kernel g satisfying (3.22). For convenience, we develop in this subsection a particular one that we use extensively in this thesis. Our solution is based on the function

$$\text{itersine}(\lambda) = \begin{cases} \sin\left(\frac{1}{2}\pi \cos^2(\pi\lambda)\right) & \text{if } \lambda \in \left[-\frac{1}{2}, \frac{1}{2}\right] \\ 0 & \text{otherwise.} \end{cases} \tag{3.24}$$

Depending on the overlap o and of K, the mother kernel g is defined as

$$g(\lambda) = \frac{2}{o}\text{itersine}\left(\frac{K-o+1}{o\lambda_{\max}}\right). \tag{3.25}$$

Using this construction, the set of filters $\{g_k(\boldsymbol{L})\}_{k=1,\dots,K}$ forms a tight filter bank when the overlap o is even. The overlap parameter allows to tune the width of the kernel g.

Efficient computation

A direct computation of the graph spectrogram requires a cubic amount of computation. Indeed there are NK filtering operations to be done, making the complete computation scale as $O\left(N^3K\right)$ for exact filtering or $O(NEKO_c)$ if the Chebyshev approximation is used (See Section 2.4). A direct computation is thus clearly impossible for large graphs (more than a few thousands nodes).

To overcome this issue, we present an approximation with a linear complexity with the number

of edges E and the number of filters K. The central idea is summarized in the following Lemma.

Lemma 3. *Let $\mathcal{G}$ be a graph of size N, g a kernel and $\boldsymbol{w} \sim \mathcal{D}(\mathbf{0}_N, \boldsymbol{I}_N)$ a vector of i.i.d. random variables, then we have*

$$\mathbb{E}\left[\left(\langle \boldsymbol{w}, \mathcal{T}_i^G g\rangle\right)^2\right] = \left\|\mathcal{T}_i^G g\right\|_2^2. \tag{3.26}$$

Proof. A direct computation shows:

$$\mathbb{E}\left[\left|\langle \boldsymbol{w}, \mathcal{T}_i^G g\rangle\right|^2\right] = \mathbb{E}\left[(\mathcal{T}_i^G g)^* \boldsymbol{w}\boldsymbol{w}^* \mathcal{T}_i^G g\right] = (\mathcal{T}_i^G g)^* \mathbb{E}\left[\boldsymbol{w}\boldsymbol{w}^*\right] \mathcal{T}_i^G g = \left\|\mathcal{T}_i^G g\right\|_2^2.$$

□

Lemma 3 tells us that $\left\|\mathcal{T}_i^G g\right\|_2^2$ can be estimated for all vertices v_i at once using M graph filtering operation. Based on this idea, we propose Algorithm 2 that has a complexity $O(KMEO_c)$, where O_c is the order of the Chebyschev approximation and M the number of random signals to be filtered. The parameter M controls the trade-off between accuracy and complexity as detailed by Theorem 5.

Algorithm 2 Fast Graph Spectrogram

1: INPUT: $\mathcal{G}$, K, g, M, $\mathcal{D}(\mathbf{0}, \boldsymbol{I})$.
2: Draw M random i.i.d signals $\boldsymbol{w}_m$ according to the distribution $\mathcal{D}$.
3: Filter all random signals for all kernels: $\boldsymbol{x}_{k,m} = g_k(\boldsymbol{L})\boldsymbol{w}_m$.
4: Average the squared filtered signals: $\dot{\boldsymbol{S}}_{\mathcal{G}}[i,k] = \frac{1}{M}\sum_{m=1}^{M}(\boldsymbol{x}_{k,m}[i])^2$.

Theorem 5. *For every distribution $\mathcal{D}$ with a zero first moment $m_1 = 0$, a second moment $m_2 = 1$ and bounded fourth order moments m_4, the sample Fast Graph Spectrogram estimator $\boldsymbol{S}_{\mathcal{G}}[\dot{i}, k]$*

(a) is unbiased,

$$\mathbb{E}\left[\dot{\boldsymbol{S}}_{\mathcal{G}}[i,k]\right] = \boldsymbol{S}_{\mathcal{G}}[i,k], \tag{3.27}$$

(b) and has variance

$$\mathbf{Var}\left[\dot{\boldsymbol{S}}_{\mathcal{G}}[i,k]\right] = \frac{1}{M}\left(\left\|\mathcal{T}_i^G g_k\right\|_4^4 (m_4 - 3) + 2\left\|\mathcal{T}_i^G g_k\right\|_2^4\right). \tag{3.28}$$

The proof is given in Appendix B.5.

Theorem 5 characterizes our estimator and provides a good insight to choose a sampling distribution for the second step of Algorithm 2. For example, if a Gaussian distribution is selected, i.e., $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, then $m_4 = 3$ and the variance becomes simply:

$$\mathbf{Var}\left[\dot{\boldsymbol{S}}_{\mathcal{G}}[i,k]\right] = \frac{2}{M}\left\|\mathcal{T}_i^G g\right\|_2^4.$$

To minimize the variance, a distribution with negative excess kurtosis should be chosen. The optimal one is a centered Bernoulli distribution with the parameter $p = 0.5$:

$$\boldsymbol{w}[n] = \begin{cases} 1 & \text{with probability 0.5 and} \\ -1 & \text{with probability 0.5.} \end{cases} \tag{3.29}$$

In this case, $m_4 = 1$ and the variance from (3.28) is minimized

$$\mathbf{Var}[\boldsymbol{S}_{\mathcal{G}}[i,k]] = \frac{2}{M}\left(\left\|\mathcal{T}_i^G g\right\|_2^4 - \left\|\mathcal{T}_i^G g_k\right\|_4^4\right). \tag{3.30}$$

Furthermore the factor $\left\|\mathcal{T}_i^G g\right\|_2^4 - \left\|\mathcal{T}_i^G g_k\right\|_4^4$ depends on the concentration level of the vector $\mathcal{T}_i^G g$. The sparser $\mathcal{T}_i^G g$, the smaller the difference and vice versa. In the extreme case where $g(x) = 1$ and thus $\mathcal{T}_i^G g = \boldsymbol{\delta}_i$, the variance is 0 and the estimator is exact. In the worst case where $g(x) = \sqrt{N}\delta_0(x)$ and thus $\mathcal{T}_i^G g = \frac{1}{\sqrt{N}}$, $\left\|\mathcal{T}_i^G g\right\|_2^4 - \left\|\mathcal{T}_i^G g_k\right\|_4^4 = \left\|\mathcal{T}_i^G g\right\|_2^4 \left(1 - \frac{1}{N}\right)$. Note that in general, the concentration level of an atom $\mathcal{T}_i^G g$ does not depend on the graph size.

3.3 Application: structural clustering

In practice, to compute a graph spectrogram, M different values of μ are selected and hence each vertex is assigned with a M-dimensional feature vector that characterizes its surrounding structure. In this context, a natural application is to classify graph nodes according to their role. The approach we will follow in this section is very different from traditional clustering methods where nodes are classified according to their position.

3.3.1 Spectral clustering

Traditional spectral clustering [131, 132] is done by computing the first k eigenvectors (associated with the lowest eigenvalues) of the (normalized) graph Laplacian. These eigenvectors form a low dimensional embedding for the graph vertices. Thus we can compute a distance between the vertices. The final clusters are obtained using k-means on the embedding. Figure 3.4 shows an example of a few graphs.

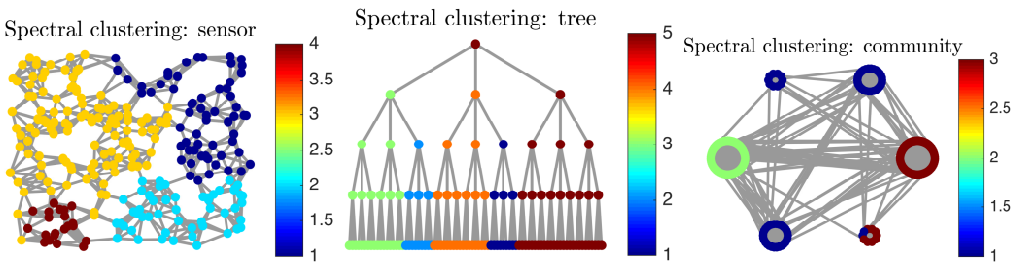


Figure 3.4 – Traditional clustering of various graphs. Nearby nodes are classified together.

This intuition behind spectral clustering is that the first graph eigenvectors also encode the smallest variations on the graph. Indeed, from (2.16), we know that the lowest eigenvectors minimize $\frac{\|\nabla_{\mathcal{G}}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2}$. As a result, close vertices tend to have similar values on the lowest eigenvectors.

3.3.2 Roots of structural clustering

It turns out that the Laplacian eigenvectors contain much more information than node locality. Associated with the eigenvalues, they can reconstruct the Laplacian and thus contain the full graph structure. There is only little literature characterizing relations between the graph structure and the graph eigenvectors. Most of the literature focus on the graph spectra [36, 71, 28] or characterize the Perron-Frobenius vector. Nevertheless, a few simple observations can be made.

- *Eigenvector localization:* Completely regular structures such as "path, ring, grid and torus" have also completely delocalized eigenvectors. Some random graphs with low diameter, such as "Erdos-Renyi" [38, 37] (and probably "D-regular") also have usually delocalized eigenvectors. In most of the other cases, at least some eigenvectors are localized. For example, sensor networks usually have localized eigenvectors.

- *Composition of graphs:* If two graphs are weakly connected together to form a third larger graph, the Fourier basis of the third graph is a perturbed version of the two original Fourier bases. This often creates localized eigenvectors. As an illustration, see the "comet" and the "community" (generated from the stochastic block model) graphs displayed in Figure 3.8 . (They are respectively referred to Graphs 8 and 9).

- *Symmetry in graphs:* Graph symmetry (also referred as "automorphism") creates eigenvalue multiplicity. More information can be found in [4, 18, 15].

- *Node degree:* When the combinatorial Laplacian is used, the degree plays an important role in the shape/localization of the eigenvectors. Node with higher degrees seems to contain higher frequency components. This is illustrated in Figure 3.5.

- *Isolated nodes:* An isolated vertex leads to Kronecker eigenvector. For an illustration, please refer to Figure 4.4 and Graph 10.

3.3.3 Definition of structural clustering

In order to construct a structural clustering algorithm, we need to define meaningful features that characterize graph vertices according to graph structure. In harmonic analysis, the eigenvectors are resonance modes of the graph. Our idea is simple:

Node with similar structures will resonate at the same frequencies.

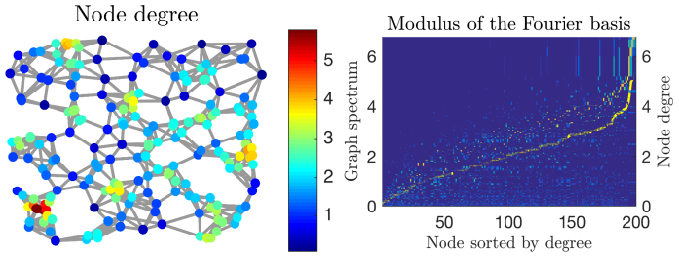


Figure 3.5 – Correlation between the nodes' degree and large values of the graph eigenvectors. Left: Random sensor graph (the signal is the degree of the nodes). Right: Modulus of the Fourier basis with the nodes sorted according to their degree. The degree is plotted in yellow. Node with higher degrees also have higher frequency components.

Since the eigenvectors form an orthonormal basis, one can see their squared modulus as a distribution over the graph spectrum. The distribution for the vertex i is defined as follows

$$\phi_i(x) = \sum_{\ell=0}^{N-1} \boldsymbol{u}_\ell^2[i]\delta_{\lambda_\ell}(x), \tag{3.31}$$

where

$$\delta_\lambda(x) = \begin{cases} 1 & \text{if } x = \lambda, and \\ 0 & \text{otherwise.} \end{cases} \tag{3.32}$$

The spectral distance between two vertices v_i and v_n is defined as the Earth Mover's Distance (EMD) [105] between the two distributions $\phi_i(x)$ and $\phi_n(x)$. The EMD distance represents the minimum mass that needs to be moved in order to transform one distribution to another one. In our case it can be computed as the difference of the Cumulative Density Functions (CDF) Φ_i.

Definition 18. *The spectral distance between vertices v_i and v_n is defined as:*

$$d_S(v_i, v_n) := \int_0^{\lambda_{\max}} |\Phi_i(x) - \Phi_n(x)|\, dx \tag{3.33}$$

where

$$\Phi_i(x) = \int_0^x \phi_i(y)\mathrm{d}y = \sum_{\substack{\ell=0 \\ \lambda_\ell \le x}}^{N-1} \boldsymbol{u}_\ell^2[i]. \tag{3.34}$$

The function Φ_i can be seen as a feature function associated to the node i. In Figure 3.6, we observe that this function has some correlation with the degree of the node. In fact, the degree of the node can be seen as a first order structural feature for graph vertices.

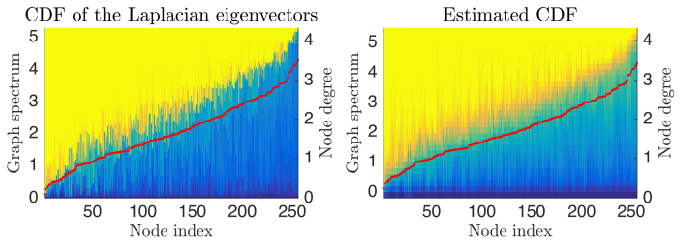


Figure 3.6 – Cumulative density function of the eigenvector energy for the sensor graph displayed in Figure 3.5. Here we use an "Itersine" construction with an overlap of $o = 4$ and $K = 30$. The nodes are ordered by degree (plotted in red). The estimated CDF is a smoothed version of the exact CDF.

Fast approximation of the structural distance

It turns out that using the graph spectrogram, we can approximate the structural distance. This avoids the diagonalization of the Laplacian, which is computationally expensive.

Let us first approximate the CDF ϕ_i. Given K kernels g_k satisfying (3.22) with $A = 1$ and $\mu_k = \frac{k-1}{K-1}\lambda_{\max}$, the CDF function $\Phi_i(\mu_k)$ is approximated by

$$\dot{\Phi}_i(\mu_k) := \sum_{l=1}^{k} \boldsymbol{S}_G[i,l] = \sum_{l=1}^{k} \left\| \mathcal{T}_i^G g_l \right\|_2^2 = \sum_{\ell=0}^{N-1} \boldsymbol{u}_\ell^2[i] \sum_{l=1}^{k} g_l^2(\lambda_\ell). \tag{3.35}$$

If the mother kernel g is localized around 0, the term $\sum_{l=1}^{k} g_l^2(\lambda_\ell)$ is a step function equal to 1 for $\lambda_\ell \ll \mu_k$ and equal to 0 for $\lambda_\ell \gg \mu_k$. The bias of the estimator is

$$\left|\Phi_i(\mu_k) - \dot{\Phi}_i(\mu_k)\right| = \left| \sum_{\ell=0}^{N-1} \left(s_{\mu_k}(\lambda_\ell) - \sum_{l=1}^{k} g_l^2(\lambda_\ell) \right) \boldsymbol{u}_\ell^2[i] \right| \leq \sum_{\ell=0}^{N-1} \left| s_{\mu_k}(\lambda_\ell) - \sum_{l=1}^{k} g_l^2(\lambda_\ell) \right| \boldsymbol{u}_\ell^2[i], \tag{3.36}$$

where

$$s_{\mu_k}(\lambda) = \begin{cases} 1 & \text{if } \lambda \leq \mu_k \\ 0 & \text{otherwise.} \end{cases}$$

The bias is characterized by how much $\sum_{l=1}^{k} g_l^2$ approximates the step down function s_{μ_k}. The term $\boldsymbol{u}_\ell^2[i]$ weights the error according to the localization of the frequency content of the vertex i. We build now our estimator of the structural distance.

Definition 19. *Given K kernels g_k satisfying* (3.22) *with $A = 1$, the estimated structural distance between nodes v_i and v_n is defined as:*

$$\dot{d}_S(v_i, v_n) := \frac{1}{K} \sum_{k=1}^{K} \left| \dot{\Phi}_i(\mu_k) - \dot{\Phi}_n(\mu_k) \right| = \frac{1}{K} \sum_{k=1}^{K} \left| \sum_{\ell=0}^{N-1} \left(\boldsymbol{u}_\ell^2[i] - \boldsymbol{u}_\ell^2[n] \right) \sum_{l=1}^{k} g_l^2(\lambda_\ell) \right| \tag{3.37}$$

Figures 3.6 and 3.7 compare the exact and approximate CDF and the structural distances for

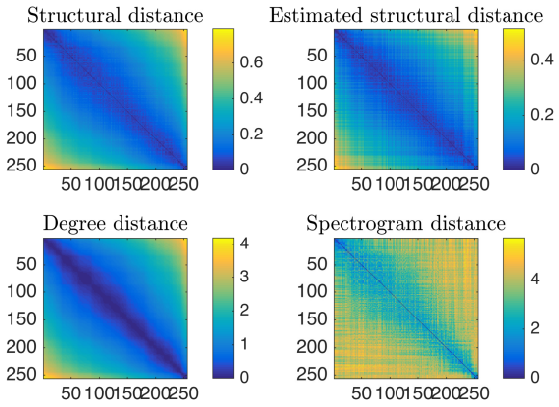


Figure 3.7 – Different distances between the vertices of a sensor graph ("Itersine" construction with an overlap of $o = 4$ and $K = 30$, sensor graph displayed in Figure 3.5). The nodes are sorted by degree.

the sensor graph. It turns out that the structural distance is highly correlated with the vetex degrees. The structural distance captures mostly local structure variations that are due to random sampling. Furthermore, these variations are well encoded by the degree. The degree can be seen as a first order structural feature, but is in general not sufficient to determine the role of the graph node.

At this point, we are ready to build a simple structural clustering algorithm. We simply consider $\dot{\Phi}_i$ as the vertex feature for the node i and apply the k-means algorithm to find N_c clusters (Algorithm 3). The algorithm complexity is governed by the computation of the graph spectrogram, which is $O(EMKO_c)$.

Algorithm 3 Structural Clustering

1: INPUT: $\mathcal{G}$, K, g, N_c.
2: Compute $\boldsymbol{S}_{\mathcal{G}}$ or its estimate using Algorithm 2.
3: Compute $\dot{\Phi}_i[k] := \sum_{l=1}^{k} \boldsymbol{S}_{\mathcal{G}}[i,l]$, $k = 1,\dots,K$
4: Use $\dot{\Phi}_i$ as a feature vector for node i and use k-means to obtain N_c clusters

3.3.4 Numerical experiments

Example 9 (Graph spectrogram and clustering)**.** *We compute the spectrogram and perform structural clustering on four different graphs, three of them are described below.*

Graph 7 (Symmetric 3-tree)**.** *A symmetric* 3*-tree is a tree graph where the root and each leaf have exactly* 3 *leaves for a specified amount of layers. In this example, we consider five-layers symmetric* 3*-tree. All nodes from the same layer are structurally identical.*

Graph 8 (Comet)**.** *A comet graph is made of a star with k vertices connected to a center vertex and a single branch of length greater than one extending from one neighbor of the center vertex.*

This graph is studied in [106, 76]. All leaves are structurally identical. In this example, we use $k = 11$ *and* $n = 20$.

Graph 9 (Community)**.** *A community graph is made of communities connected according to the stochastic block model, i.e., intra-community and extra-community connections have probability* p_{in} *and* p_{out}*. In this example we use* 6 *communities respectively of size* $10, 30, 60, 10, 30, 60$ *for a total of* $N = 200$ *nodes. We connect the nodes randomly inside a community with a probability* $p_{in} = 0.5$ *and outside of the community with probability* $p_{out} = 0.005$.

In Figure 3.8, we compute the Graph spectrogram for these four graphs (Graphs 4, 7, 8 and 9, i.e., sensor $N = 256$*, symmetric* 3*-tree, comet and community) and we perform structural clustering of Algorithm 3 ("Itersine" filter bank with an overlap of* $o = 4$ *and* $K = 30$*, k-means with* 10 *random initializations, we input the correct amount of cluster for each example). Nodes with the same* role *are classified together. The clustering results of both the tree and the comet graphs are what one would naturally expect, i.e., nodes with similar roles are grouped together. For the community graph, the algorithm gathers together communities of the same size. It is expected as they have statistically the same structure. A perfect clustering would be obtained if the structural features would have been filtered using the graph (see Example 10). In a sensor graph, the nodes are mostly clusterized according to their degree, which makes sense, since it is probably the most relevant structural information for this type of graph.*

Comparison with the degree *The degree plays an important role in the CDF(see middle column in Figure 3.8). However it is in general not sufficient and too simple to perform structural clustering. As a simple example, it is insufficient to characterize for the comet and the tree graph. Furthermore, even for the community graph where the node structure is mostly characterized by its degree, as shown in Figure 3.9, degree clustering fails.*

Eigenvalue density estimation *From property* (3.19)*, computing the spectrogram also provides us with an estimation of the eigenvalue density. Figure 3.10 compares the obtained approximations with histograms. The precision of this estimation depends on the bandwidth of the mother window* g*: the smaller the more precise. Our estimator reads*

$$\dot{p}_{\mathcal{G}}(\mu_k) = \frac{\sum_{i=1}^{N} \boldsymbol{S}_{\mathcal{G}}(i,k)}{NA} = \frac{\left\|\boldsymbol{\lambda} - \mu_k \mathbf{1}\right\|_2^2}{NA}. \tag{3.38}$$

Example 10 (Unsupervised image segmentation example)**.** *Let us illustrate further how structural clustering can be used in image processing. From the "peppers" image (*512×512 *pixels) we create a graph where each pixel is a node. The graph is created by comparing patches of pixels of size* 5×5*, i.e., we connect two nodes if their neighborhood is similar. This operation results in a graph of* $262'144$ *vertices.*

The processing is done as follows: a) we compute the spectrogram using Algorithm 2, b) we remove some noise from the spectrogram using a graph smoothing and c) we perform structural

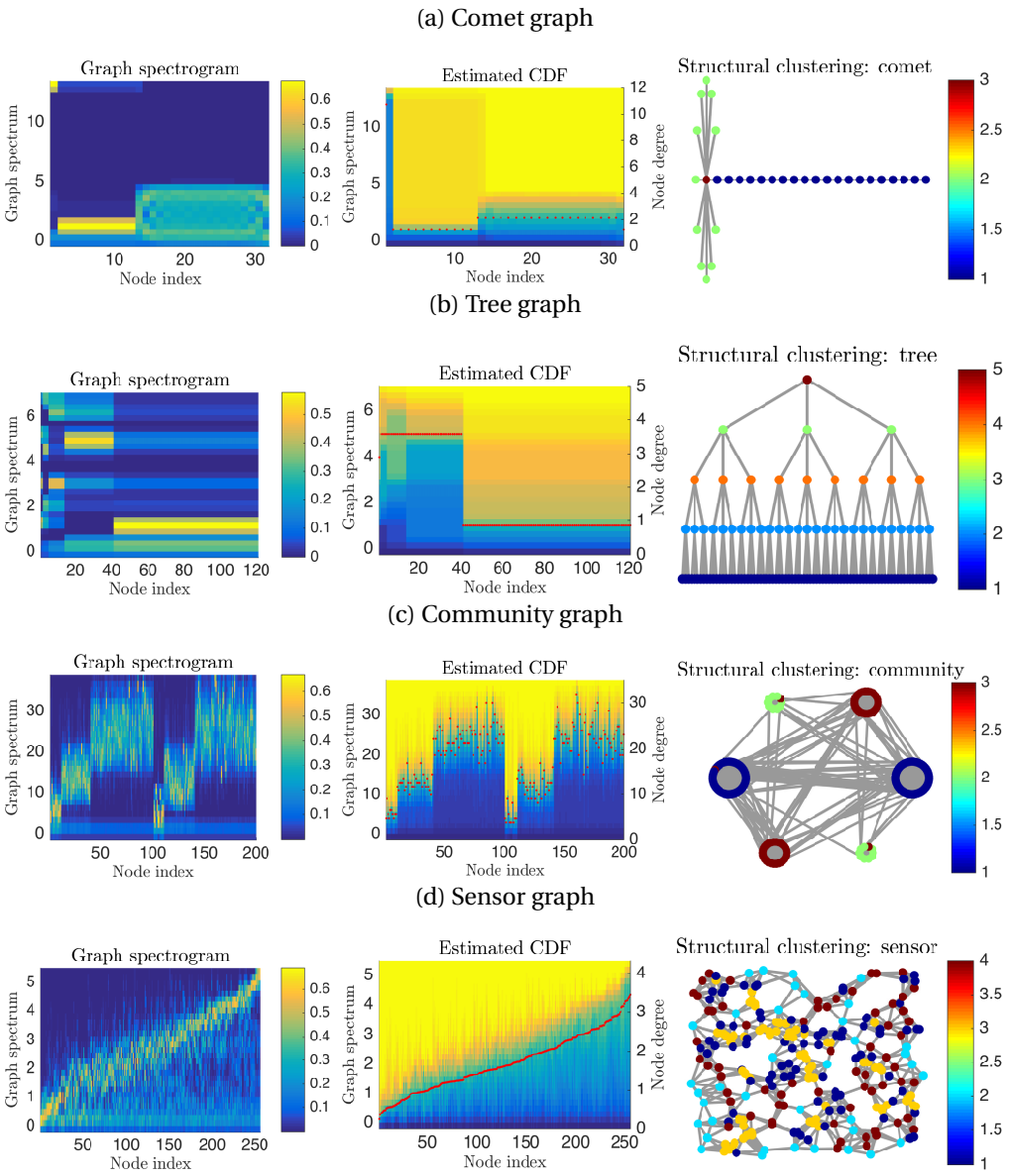


Figure 3.8 – Graph spectrograms, vertices features and structural clustering for various graphs. Structural clustering successfully classifies the node according to their role in the graph.

clustering. The denoising operation is done since we assume, for this example, that pixels with similar properties are connected together.

In Figure 3.11 top, we observe that the algorithm separates the image according to the degree of smoothness of the images: constant parts are depicted in yellow and parts with large gradients are in dark blue. On the bottom of Figure 3.11 we displayed the 4 *first bands of the graph spectrogram before and after the smoothing operation. These features captures different structural properties of the image.*

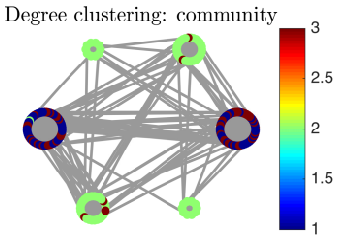


Figure 3.9 – Clustering the community graph according to the degree. Using this simple feature, the clustering algorithm is not able to cluster the communities according to their sizes.

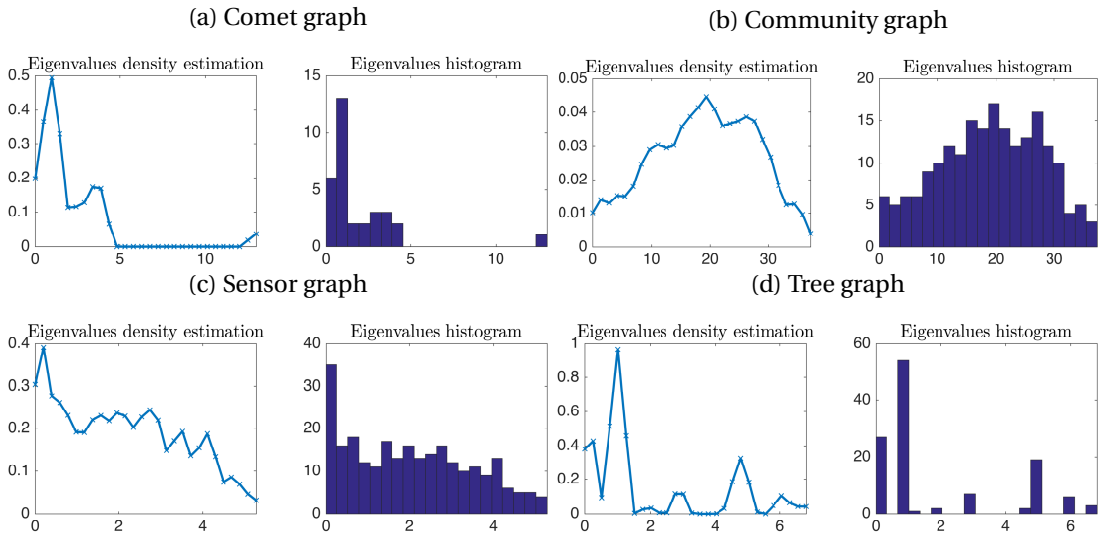


Figure 3.10 – Estimation of the eigenvalue density. Equation (3.38) successfully approximates the eigenvalue density function.

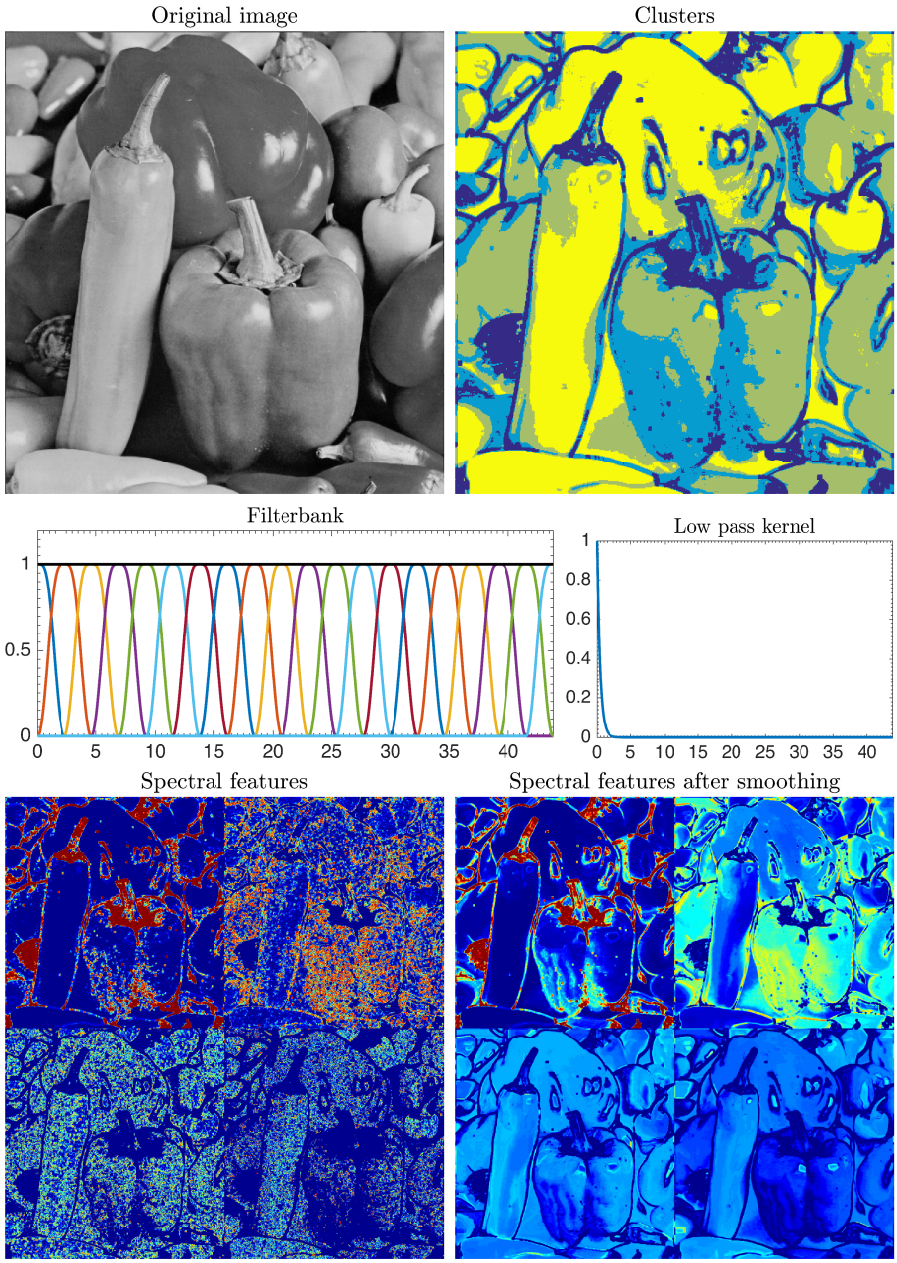


Figure 3.11 – Unsupervised pixel classification based on the image structure. Pixel are classified according to their level of smoothness in the image. The spectral features, corresponding to the first four bands of the filter bank (blue, red, orange, and purple), are carrying the structure of the image.

4 Global and local uncertainty principles for graph signals

4.1 Introduction

4.1.1 What is an uncertainty principle?

An uncertainty principle is a relation that limits the concentration of a function (signal) in one or two domains simultaneously. For example, one version of the Heisenberg Uncertainty principle states that a function f cannot be arbitrarily concentrated simultaneously in time and in frequency. The relation characterizes the mean

$$M(f) = \int_{\mathbb{R}} t|f(t)|^2 \mathrm{d}t,$$

and the variance

$$V(f) = \int_{\mathbb{R}} (t - M(f))^2 |f(t)|^2 \mathrm{d}t.$$

The Heisenberg's uncertainty principle states that any function f satisfies

$$V(f)V(\hat{f}) \geq C$$

where $C = \frac{1}{2\sqrt{2\pi}}$ or $C = \frac{1}{2}$ depending on the choice of the Fourier definition.

We emphasize that the domain $\mathbb{R}$ is regular, i.e., the spreading both in Fourier and in time is independent of the function localization. A function $f : \mathbb{R} \to \mathbb{R}$ and its shifted version $f_b(t) = f(t-b)$, have the same spreadings in both the time and the Fourier domains:

$$V\left(f\right) = V\left(f_b\right), \quad V\left(\hat{f}\right) = V\left(\hat{f}_b\right), \quad \forall b \in \mathbb{R}.$$

As a consequence, the Heisenberg uncertainty principle is also independent of the localization of f and we can characterize the entire domain $\mathbb{R}$ with the single number C.

Unfortunately, due to the inhomogeneous structure of graphs, the spreading of a signal

depends on its localization and a new technique is required to accurately bound its measure: a local uncertainty principle, i.e., a bound adapting to each graph node.[1]

4.1.2 Why study graph uncertainty principles?

In signal processing, uncertainty principles such as the ones presented in [31, 30, 34, 51, 16, 100] form an important tool to design and evaluate linear transforms for processing "classical" signals such as audio signals, time series, and images residing on Euclidean domains. It is desirable that the dictionary atoms are jointly localized in time and frequency, and uncertainty principles characterize the resolution tradeoff between these two domains. Moreover, while **"the uncertainty principle is [often] used to show that certain things are impossible,"** Donoho and Stark [31] present **"examples where the generalized uncertainty principle shows something unexpected is *possible*; specifically, the recovery of a signal or image despite significant amounts of missing information."** In particular, uncertainty principles can provide guarantees that, if a signal has a sparse decomposition in a dictionary of incoherent atoms, this is indeed a unique representation that can be recovered via optimization [30, 34].

Many of the multiscale transforms designed for graph signals attempt to leverage intuition from signal processing techniques designed for signals on Euclidean data domains by generalizing fundamental operators and transforms to the graph setting (e.g., by checking that they correspond on a ring graph). While some intuition, such as the notion of filtering with a Fourier basis of functions that oscillate at different rates, carries over to the graph setting, the irregular structure of the graph domain often restricts our ability to generalize ideas. One prime example is the lack of a shift-invariant notion of translation of a graph signal that is discussed in Chapter 3. As shown in [70, 106] and discussed in [117, Section 3.2], the concentration of the Fourier basis functions is another example where the intuition does not carry over directly. Complex exponentials, the basis functions for the classical Fourier transform, have global support across the real line. On the other hand, the eigenvectors of the combinatorial or normalized graph Laplacians, which are most commonly used as the basis functions for a graph Fourier transform, are sometimes localized to small regions of the graph. Because the incoherence between the Fourier basis functions and the standard normal basis underlies many uncertainty principles, we demonstrate this issue with a short example.

Example 11 (Part I: Laplacian eigenvector localization)**.** *Let us consider the two manifolds (surfaces) embedded in* $\mathbb{R}^3$ *and shown in the first row of Figure 4.1. The first one is a flat square. The second is identical except for the center where it contains a spike. We sample both of these manifolds uniformly across the* x*-*y *plane and create a graph by connecting the* 8 *nearest neighbors with weights depending on the distance (*$\mathcal{W}(v_i, v_n) = e^{-d_{in}/\sigma}$*). The energy of each Laplacian eigenvector of the graph arising from the first manifold is not concentrated on any particular vertex; i.e.,* $\max_{i,\ell} |\boldsymbol{u}_\ell[i]| \ll 1$*, where* $\boldsymbol{u}_\ell$ *is the eigenvector associated with eigenvalue* λ_ℓ*.*

[1] This chapter is availlable with small modifications in [89].

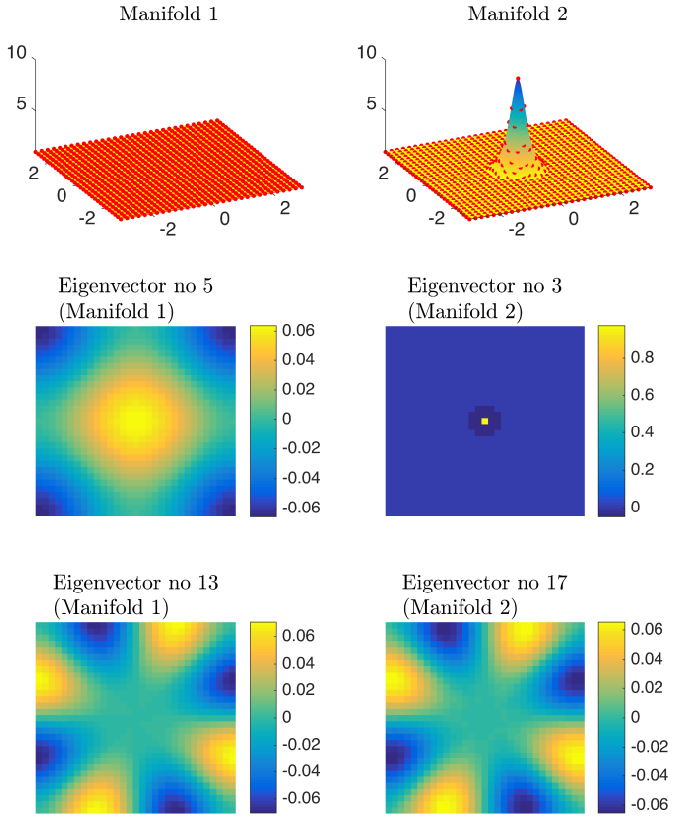


Figure 4.1 – Concentration of graph Laplacian eigenvectors. We discretize two different manifolds by sampling uniformly across the x-y plane. Due its bumpy central part, the graph arising from manifold 2 has a graph Laplacian eigenvector (shown in the middle row of the right column) that is highly concentrated in both the vertex and graph spectral domains. However, the eigenvectors of this graph whose energy primarily resides in the flatter parts of the manifold (such as the one shown in the bottom row of the right column) are less concentrated, and some closely resemble the Laplacian eigenvectors of the graph arising from the flat manifold 1 (such as the corresponding eigenvector shown in the bottom row of the left column.

However, the graph arising from the second manifold does have a few eigenvectors, such as eigenvector 3 *shown in the middle row Figure 4.1, whose energy is highly concentrated on the region of the spike; i.e.,* $\max_{i,\ell} |\boldsymbol{u}_\ell[i]| \approx 1$. *Yet, the Laplacian eigenvectors of this second graph whose energy resides primarily on the flatter regions of the manifold, such as eigenvector* 17 *shown in the bottom row of Figure 4.1, are not too concentrated on any single vertex. Rather, they more closely resemble some of the Laplacian eigenvectors of the graph arising from the first manifold.*

4.1.3 Classification of uncertainty principle and related work

Uncertainty principles can be divided into three different families that we detail in this section. Furthermore, the majority of uncertainty principles are independent of the signal localization. We call them "global" as opposed to "local" when the signal localization matters.

Family A: Concentration around a reference point

The first family of uncertainty principles measure the spreading *around some reference point*, usually the mean position of the energy contained in the signal. The well-known Heisenberg uncertainty principle [40, 68] presented in the beginning of this section belongs to this family. It views the modulus square of the signal in both the time and Fourier domains as energy probability density functions, and takes the variance of those energy distributions as measures of the spreading in each domain. The uncertainty principle states that the product of variances in the time and in the Fourier domains cannot be arbitrarily small. The generalization of this uncertainty principle to the graph setting is complex since there does not exist a simple formula for the mean value or the variance of graph signals, in either the vertex or the graph spectral domains.

For unweighted graphs, Agaskar and Lu [1, 2, 3] also view the square modulus of the signal in the vertex domain as an energy probability density function and use the geodesic graph distance (shortest number of hops) to define the spread of a graph signal around a given center vertex. For the spread of a signal $\boldsymbol{x}$ in the graph spectral domain, Agaskar and Lu use the normalized variation $\frac{\boldsymbol{x}^* \boldsymbol{L} \boldsymbol{x}}{\|\boldsymbol{x}\|_2^2}$, which captures the smoothness of a signal. They then specify uncertainty curves that characterize the tradeoff between the smoothness of a graph signal and its localization in the vertex domain. This idea is generalized to weighted graphs in [80].

As pointed out in [3], the tradeoff between smoothness and localization in the vertex domain is intuitive as a signal that is smooth with respect to the graph topology cannot feature values that decay too quickly from the peak value. However, as shown in Figure 4.1 (and subsequent examples in Table 4.1), graph signals can indeed be simultaneously highly localized or concentrated in both the vertex domain and the graph spectral domain. This discrepancy is because the normalized variation used as the spectral spread in [3] is one method to measure the spread of the spectral representation around the eigenvalue 0, rather than around some mean of that signal in the graph spectral domain.

In fact, using the notion of spectral spread presented in [3], the graph signal with the highest spectral spread on a graph $\mathcal{G}$ is the graph Laplacian eigenvector associated with the highest eigenvalue. The graph spectral representation of that signal is a Kronecker delta whose energy is completely localized at a single eigenvalue. One might argue that its *spread* should in fact be zero. So, in summary, while there does exist a tradeoff between the smoothness of a graph signal and its localization around any given center vertex in the vertex domain, the classical idea that a signal cannot be simultaneously localized in the time and frequency domains does

not always carry over to the graph setting. While certainly an interesting avenue for continued investigation, we do not discuss uncertainty principles based on spreads in the vertex and graph spectral domains any further in this thesis.

Family B: Absolute concentration

The second family of uncertainty principles involve the absolute sparsity or concentration of a signal. The key quantities are typically either support measures counting the number of non-zero elements, or concentration measures, such as ℓ^p-norms. An important distinction is that these sparsity and concentration measures are not localization measures. They can give the same values for different signals, independent of whether the predominant signal components are clustered in a small region of the vertex domain or spread across different regions of the graph. An example of a recent work from the graph signal processing literature that falls into this family is [130], in which Tsitsvero et al. propose an uncertainty principle that characterizes how jointly concentrated graph signals can be in the vertex and spectral domains. Generalizing prolate spheroidal wave functions [118], their notion of concentration is based on the percentage of energy of a graph signal that is concentrated on a given set of vertices in the vertex domain and a given set of frequencies in the graph spectral domain.

Since we can interpret signals defined on graphs as finite dimensional vectors with well-defined ℓ^p-norms, we can also apply directly the results of existing uncertainty principles for finite dimensional signals. As one example, the Elad-Bruckstein uncertainty principle of [34] states that if $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the coefficients of a vector $\boldsymbol{x} \in \mathbb{R}^N$ in two different orthonormal bases, then

$$\frac{\|\boldsymbol{\alpha}\|_0 + \|\boldsymbol{\beta}\|_0}{2} \geq \sqrt{\|\boldsymbol{\alpha}\|_0 \cdot \|\boldsymbol{\beta}\|_0} \geq \frac{1}{\mu}, \tag{4.1}$$

where μ is the maximum magnitude of the inner product between any vector in the first basis with any vector in the second basis.

Family C: Joint representation

The third family of uncertainty principles characterizes a single joint representation of time and frequency. The short-time Fourier transform (STFT) is an example of a time-frequency representation that projects a signal $\boldsymbol{x}$ onto a set of translated and modulated copies of a function g. Usually, g is a function localized in the time-frequency plane, for example a Gaussian, vanishing away from some known reference point in the joint time and frequency domain. Hence this transformation reveals *local properties in time and frequency* of $\boldsymbol{x}$ by separating the time-frequency domain into regions where the translated and modulated copies of g are localized. This representation obeys an uncertainty principle: the STFT coefficients cannot be arbitrarily concentrated. This can be shown by estimating the different ℓ^p-norms of this representation (note that the concentration measures of the second family of

uncertainty principles are used). For example, Lieb [64] proves a concentration bound on the ambiguity function (e.g., the STFT coefficients of the STFT atoms). Lieb's approach is more general than the Heisenberg uncertainty principle, because it handles the case where the signal is concentrated around multiple different points (see, e.g., the signal $\boldsymbol{x}_3$ in Figure 4.2).

Local uncertainty principles

While the family B of uncertainty principles above in general yields *global uncertainty principles*, we can generalize family C to the graph setting in a way that yields *local uncertainty principles*. In the classical Euclidean setting, the underlying domain is homogeneous, and thus uncertainty principles apply to all signals equally, regardless of where on the real line they are concentrated. **However, in the graph setting, the underlying domain is irregular, and a change in the graph structure in a single small region of the graph can drastically affect the uncertainty bounds.** For instance, the second family of uncertainty principles all depend on the coherence between the graph Laplacian eigenvectors and the standard normal basis of Kronecker deltas, which is a global quantity in the sense that it incorporates local behavior from all regions of the graph. To see how this can limit the usefulness of such global uncertainty principles, we return to the motivating example 11.

Example 12 (Part II: Global versus local uncertainty principles)**.** *In Section 4.2.2, we show that a direct application of a result from [100] to the graph setting yields the following uncertainty relationship, which falls into the second family described above, for any signal $\boldsymbol{x} \in \mathbb{R}^N$:*

$$\left(\frac{\|\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_1}\right)\left(\frac{\|\hat{\boldsymbol{x}}\|_2}{\|\hat{\boldsymbol{x}}\|_1}\right) \leq \max_{i,\ell} |\boldsymbol{u}_\ell[i]|. \tag{4.2}$$

Each fraction in the left-hand side of (4.2) *is a measure of concentration that lies in the interval* $[\frac{1}{\sqrt{N}}, 1]$ *(N is the number of vertices), and the coherence between the graph Laplacian eigenvectors and the Kronecker deltas on the right-hand side lies in the same interval. On the graph arising from manifold 1 (Figure 4.1), the coherence is close to* $\frac{1}{\sqrt{N}}$*, and* (4.2) *yields a meaningful uncertainty principle. However, on the graph arising from manifold 2 (Figure 4.1), the coherence is close to 1 due to the localized eigenvector 3 in Figure 4.1,* (4.2) *is trivially true for any signal in* $\mathbb{R}^n$ *from the properties of vector norms, and thus the uncertainty principle is not particularly useful. Nevertheless, far away from the spike, signals should behave similarly on manifold 2 to how they behave on manifold 1. Part of the issue here is that the uncertainty relationship holds for any graph signal $\boldsymbol{x}$, even those concentrated on the spike, which we know can be jointly localized in both the vertex and graph spectral domains. An alternative approach is to develop a local uncertainty principle that characterizes the uncertainty in different regions of the graph on a separate basis. Then, if the energy of a given signal is concentrated on a more homogeneous part of the graph, the concentration bounds will be tighter.*

4.1.4 Concentration measures

In this thesis, we focus on Families B and C. Thus we need to introduce some concentration/sparsity measures. **Throughout the thesis, we use the terms *sparsity* and *concentration* somewhat interchangeably, but we reserve the term *spread* to describe the spread of a function around some mean or center point (Family C).**

- The first concentration measure is the support measure of $\boldsymbol{x}$, denoted $\|\boldsymbol{x}\|_0$, which counts the number of non-zero elements of $\boldsymbol{x}$.
- The second concentration measure is the Shannon entropy, which is used often in information theory and physics:

$$H(\boldsymbol{x}) = -\sum_n |\boldsymbol{x}[n]|^2 \ln |\boldsymbol{x}[n]|^2, \tag{4.3}$$

  where the variable n has values in $\{1,2,\dots,N\}$ for functions on graphs and in $\{0,1,\dots,N-1\}$ in the graph Fourier representation.
- Another class of concentration measures is the ℓ^p-norms, with $p \in [1,\infty]$. For $p \neq 2$, the sparsity of $\boldsymbol{x}$ may be measured using the following quantity: p-concentration measure

$$s_p(\boldsymbol{x}) = \begin{cases} \frac{\|\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_p}, & \text{if } 1 \leq p \leq 2 \\ \frac{\|\boldsymbol{x}\|_p}{\|\boldsymbol{x}\|_2}, & \text{if } 2 < p \leq \infty. \end{cases} \tag{4.4}$$

  For any vector $\boldsymbol{x} \in \mathbb{C}^N$ and any $p \in [1,\infty]$, $s_p(\boldsymbol{x}) \in \left[N^{-\left|\frac{1}{p}-\frac{1}{2}\right|}, 1\right]$. If $s_p(\boldsymbol{x})$ is high (close to 1), then $\boldsymbol{x}$ is sparse, and if $s_p(\boldsymbol{x})$ is low, then $\boldsymbol{x}$ is not concentrated.

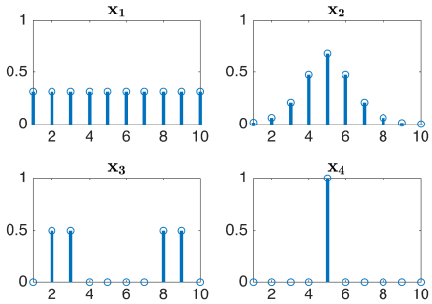
Figure 4.2 uses some basic signals to illustrate this notion of concentration, for different values of p. In addition to sparsity, one can also relate ℓ^p-norms to the Shannon entropy via Renyi entropies (see, e.g., [99, 101] for more details).

4.2 Generalization of traditional uncertainty principles

In this section, we introduce some notation and illustrate further how certain intuitionw from signal processing on Euclidean spaces do not carry over to the graph setting.

4.2.1 Concentration of the graph Laplacian eigenvectors

The spectrum of the graph Laplacian replaces the frequencies as coordinates in the Fourier domain. For the special case of shift-invariant graphs with circulant graph Laplacians [49, Section 5.1], the Fourier eigenvectors can still be viewed as pure oscillations. However, for more general graphs (i.e., all but the most highly structured), the oscillatory behavior of the Fourier eigenvectors must be interpreted more broadly. For example, [114, Figure 3] displays



p	$s_p(\boldsymbol{x}_1)$	$s_p(\boldsymbol{x}_2)$	$s_p(\boldsymbol{x}_3)$	$s_p(\boldsymbol{x}_4)$
1.00	0.32	0.46	0.50	1.00
1.33	0.56	0.69	0.71	1.00
2.00	1.00	1.00	1.00	1.00
4.00	0.56	0.75	0.71	1.00
∞	0.32	0.68	0.50	1.00

Figure 4.2 – The concentration $s_p(\cdot)$ of four different example signals (all with 2-norm equal to 1), for various values of p. Note that the position of the signal coefficients does not matter for this concentration measure. Different values of p lead to different notions of concentration; for example, $\boldsymbol{x}_2$ is more concentrated than $\boldsymbol{x}_3$ if $p = \infty$ (it has a larger maximum absolute value), but less concentrated if $p = 1$.

the number of zero crossings of each eigenvector; that is, for each eigenvector, the number of pairs of connected vertices where the signs of the values of the eigenvector at the connected vertices are opposite. It is generally the case that the graph Laplacian eigenvectors associated with larger eigenvalues contain more zero crossings, yielding a notion of frequency to the graph Laplacian eigenvalues. However, despite this broader notion of frequency, the graph Laplacian eigenvectors are not always globally-supported pure oscillations like the complex exponentials. In particular, they can feature sharp peaks, meaning that some of the Fourier basis elements can be much more similar to an element of the canonical basis of Kronecker deltas on the vertices of the graph. As we will see, uncertainty principles for signals on graphs are highly affected by this phenomenon.

One way to compare a graph Fourier basis to the canonical basis is to compute the coherence between these two representations:

$$\mu_{\mathcal{G}} = \max_{i,\ell} |\langle \boldsymbol{\delta}_i, \boldsymbol{u}_\ell \rangle| = \max_{i,\ell} |\boldsymbol{u}_\ell[i]| = \max_{\ell} s_\infty(\boldsymbol{u}_\ell).$$

This quantity measures the similarity between the two sets of vectors. If the sets possess a common vector, then $\mu_{\mathcal{G}} = 1$ (the maximum possible value for $\mu_{\mathcal{G}}$). If the two sets are maximally incoherent, such as the canonical and Fourier bases in the standard discrete setting, then $\mu_{\mathcal{G}} = 1/\sqrt{N}$ (the minimum possible value).

Because the graph Laplacian matrix encodes the weights of the edges of the graph, the coherence $\mu_{\mathcal{G}}$ clearly depends on the structure of the underlying graph. It remains an open question exactly how structural properties of weighted graphs such as the regularity, clustering, modularity, and other spectral properties can be linked to the concentration of the graph Laplacian eigenvectors. For certain classes of random graphs [29, 33, 129] or large regular graphs [14], the eigenvectors have been shown to be non-localized, globally oscillating functions (i.e., $\mu_{\mathcal{G}}$ is low). Yet, empirical studies such as [70] show that graph Laplacian eigenvectors can be

highly concentrated (i.e., $\mu_{\mathcal{G}}$ can be close to 1), particularly when the degree of a vertex is much higher or lower than the degrees of other vertices in the graph. The following example illustrates how $\mu_{\mathcal{G}}$ can be influenced by the graph structure.

Example 13. *In this example, we discuss two classes of graphs that can have graph Fourier coherences. The first, called comet graphs (Graph 8), are studied in [106, 76]. They are composed of a star with k vertices connected to a center vertex, and a single branch of length greater than one extending from one neighbor of the center vertex (see Figure 4.3, top). If we fix the length of the longer branch (it has length 10 in Figure 4.3), and increase k, the number of neighbors of the center vertex, the graph Laplacian eigenvector associated with the largest eigenvalue approaches a Kronecker delta centered at the center vertex of the star. As a consequence, the coherence between the graph Fourier and the canonical bases approaches 1 as k increases.*

The second class consists in modified path graphs, which we use several times in this chapter.

Graph 10 (Modified path)**.** *We start with a standard path graph of* 10 *nodes equally spaced (all edge weights are equal to one) and we move the first node out to the left; i.e., we reduce the weight between the first two nodes (see Figure 4.3, bottom). The weight is related to the distance by $\boldsymbol{W}_{12} = {}^{1}/_{d_{12}}$ with d_{12} being the distance between the nodes 1 and 2.*

When the weight between nodes 1 and 2 decreases, the eigenvector associated with the largest eigenvalue of the Laplacian becomes more concentrated, which increases the coherence $\mu_{\mathcal{G}}$. These two examples of simple families of graphs illustrate that the topology of the graph can impact the graph Fourier coherence, and, in turn, uncertainty principles that depend on the coherence.

In Figure 4.4, we display the eigenvector associated with the largest graph Laplacian eigenvalue for a modified path graph of 100 nodes, for several values of the weight $\boldsymbol{W}_{12}$. Observe that the shape of the eigenvector has a sharp local change at node 1.

Example 13 demonstrates an important point to keep in mind. A small local change in the graph structure can greatly affect the behavior of one eigenvector, and, in turn, a global quantity such as $\mu_{\mathcal{G}}$. However, intuitively, a small local change in the graph should not drastically change the processing of signal values far away, for example in a denoising or inpainting task. For this reason, in Section 4.4.1, we introduce a notion of local uncertainty that depicts how the graph is behaving locally.

Note that not only special classes of graphs or pathological graphs yield highly localized graph Laplacian eigenvectors. Rather, graphs arising in applications such as sensor or transportation networks, or graphs constructed from sampled manifolds (such as the graph sampled from manifold 2 in Figure 4.1) can also have graph Fourier coherences close to 1 (see, e.g., [117, Section 3.2] for further examples).

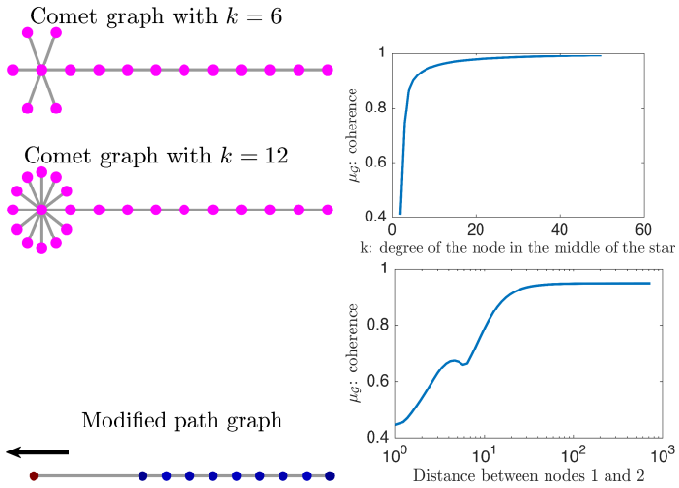


Figure 4.3 – Coherence between the graph Fourier basis and the canonical basis for the graphs described in Example 13. Top left: Comet graphs with $k = 6$ and $k = 12$ branches, all of length one except for one of length ten. Top right: Evolution of the graph Fourier coherence $\mu_{\mathcal{G}}$ with respect to k. Bottom left: Example of a modified path graph with 10 nodes. Bottom right: Evolution of the coherence of the modified path graph with respect to the distance between nodes 1 and 2. As the degree of the comet's center vertex increases or the first node of the modified path is pulled away, the coherence $\mu_{\mathcal{G}}$ tends to the limit value $\sqrt{\frac{N-1}{N}}$.

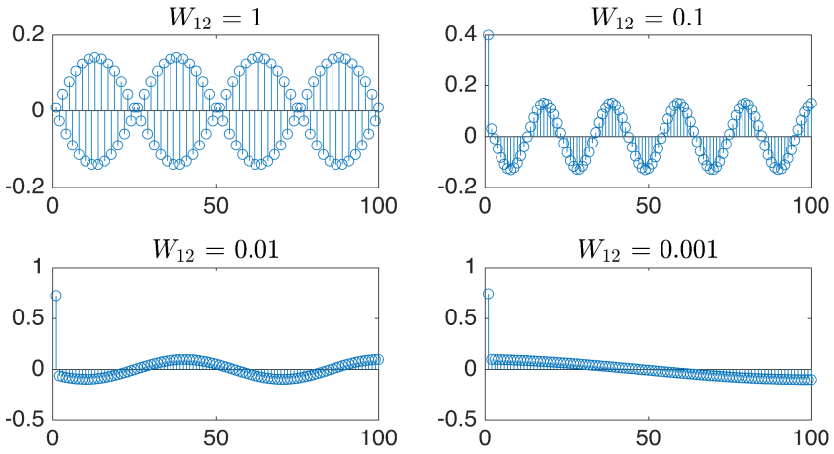


Figure 4.4 – Eigenvectors associated with the largest graph Laplacian eigenvalue of the modified path graph with 100 nodes, for different values of $\boldsymbol{W}_{12}$. As the distance between the first two nodes increases, the eigenvector becomes sharply peaked.

4.2.2 Direct applications of uncertainty principles for discrete signals

We start by applying three known uncertainty principles for discrete signals to the graph setting.

Theorem 6. *Let $\boldsymbol{x} \in \mathbb{C}^N$ be a nonzero signal defined on a connected, weighted, undirected graph $\mathcal{G}$, let $\{\boldsymbol{u}_\ell\}_{\ell=0,1,\ldots,N-1}$ be a graph Fourier basis for $\mathcal{G}$, and let $\mu_\mathcal{G} = \max_{i,\ell} |\langle \boldsymbol{\delta}_i, \boldsymbol{u}_\ell \rangle|$. We have the following four uncertainty principles:*

(i) the support uncertainty principle [34]

$$\frac{\|\boldsymbol{x}\|_0 + \|\hat{\boldsymbol{x}}\|_0}{2} \geq \sqrt{\|\boldsymbol{x}\|_0 \, \|\hat{\boldsymbol{x}}\|_0} \geq \frac{1}{\mu_\mathcal{G}}, \tag{4.5}$$

(ii) the ℓ^p-norm uncertainty principle [100]

$$\|\boldsymbol{x}\|_p \, \|\hat{\boldsymbol{x}}\|_p \geq \mu_\mathcal{G}^{1-\frac{2}{p}} \, \|\boldsymbol{x}\|_2^2, \qquad p \in [1,2], \tag{4.6}$$

(iii) the entropic uncertainty principle [67]

$$H(\boldsymbol{x}) + H(\hat{\boldsymbol{x}}) \geq -2 \ln \mu_\mathcal{G}. \tag{4.7}$$

(iv) the 'local' uncertainty principle [40]

$$\sum_{i \in \mathcal{V}_S} |\boldsymbol{x}[i]|^2 \leq |\mathcal{V}_S| \, \|\boldsymbol{x}\|_\infty^2 \leq |\mathcal{V}_S| \mu_\mathcal{G}^2 \, \|\hat{\boldsymbol{x}}\|_1^2 \tag{4.8}$$

for any subset $\mathcal{V}_S$ of the vertices $\mathcal{V}$ in the graph $\mathcal{G}$.

The first uncertainty principle is given by a direct application of the Elad-Bruckstein inequality [34]. It states that the sparsity of a function in one representation limits the sparsity in a second representation. As displayed in (4.1), the work of [34] holds for representations in any two bases. As we have seen, if we focus on the canonical basis $\{\boldsymbol{\delta}_i\}_{i=1,\ldots,N}$ and the graph Fourier basis $\{\boldsymbol{u}_\ell\}_{\ell=0,\ldots,N-1}$, the coherence $\mu_\mathcal{G}$ depends on the graph topology. For the ring graph, $\mu_\mathcal{G} = \frac{1}{\sqrt{N}}$, and we recover the result from the standard discrete case (regular sampling, periodic boundary conditions). However, for graphs where $\mu_\mathcal{G}$ is closer to 1, the uncertainty principle (4.5) is much weaker and therefore less informative. For example, $\|\hat{\boldsymbol{x}}\|_0 \, \|\boldsymbol{x}\|_0 \geq \frac{1}{\mu_\mathcal{G}^2} \approx 1$ is trivially true of nonzero signals.

The same caveat applies to (4.6) and (4.7), which follow directly from [100] and [67], respectively, by once again specifying the canonical and graph Fourier bases. The last inequality (4.8) is an adaptation [40, Eq. (4.1)] to the graph setting, using the Hausdorff-Young inequality of Theorem 7 (see next section). It states that the energy of a function in a subset of the domain is bounded from above by the size of the selected subset and the sparsity of the function in the Fourier domain. If the subset $\mathcal{V}_S$ is small and the function is sparse in the graph Fourier domain, this uncertainty principle limits the amount of energy of $\boldsymbol{x}$ that fits inside of the subset of $\mathcal{V}_S$. Because $\mathcal{V}_S$ can be chosen to be a local region of the domain (the graph vertex domain in our case), Folland and Sitaram [40] refer to such principles as "local uncertainty inequalities." However, the term $\mu_\mathcal{G}$ in the uncertainty bound is not local in the sense that it

depends on the whole graph structure and not just on the topology of the subgraph containing vertices in $\mathcal{V}_S$.

The following example illustrates the relation between the graph, the concentration of a specific graph signal, and one of the uncertainty principles from Theorem 6. We return to this example in Section 4.2.4 to discuss further the limitations of these uncertainty principles featuring $\mu_{\mathcal{G}}$.

Example 14. *Figure 4.5 shows the computation of the quantities involved in* (4.6)*, with* $p = 1$ *and different* $\mathcal{G}$*'s taken to be the modified path graphs of Example 13, with different distances between the first two vertices. We show the left-hand side of* (4.6) *for two different Kronecker deltas, one centered at vertex 1, and one centered at vertex 10. We have seen in Figure 4.3 that, as the distance between the first two vertices increases, the coherence increases, and therefore the lower bound on the right-hand side of* (4.6) *decreases. For* $\boldsymbol{\delta}_1$*, the uncertainty quantity on the left-hand side of* (4.6) *follows a similar pattern. The intuition behind this is that, as the weight between the first two vertices decreases, a few of the eigenvectors start to have local jumps around the first vertex (see Figure 4.4). As a result, we can sparsely represent* $\boldsymbol{\delta}_1$ *as a linear combination of those eigenvectors and* $\left\|\widehat{\boldsymbol{\delta}_1}\right\|_1$ *is reduced. However, since there are not any eigenvectors that are localized around the last vertex in the path graph, we cannot find a sparse linear combination of the graph Laplacian eigenvectors to represent* $\boldsymbol{\delta}_{10}$*. Therefore, its uncertainty quantity on the left-hand side of* (4.6) *does not follow the behavior of the lower bound.*

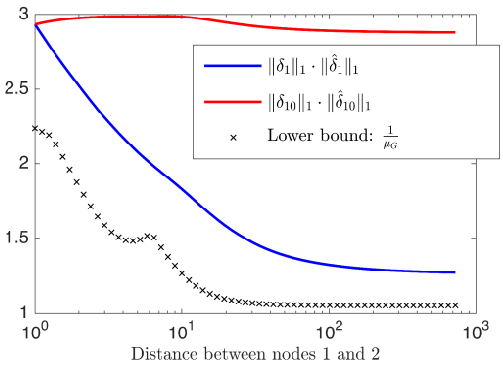


Figure 4.5 – Numerical illustration of the ℓ^p-norm uncertainty principle on a sequence of modified path graphs with different mutual coherences between the canonical basis of deltas and the graph Laplacian eigenvectors. For each modified path graph, the weight $\boldsymbol{W}_{12}$ of the edge between the first two vertices is the reciprocal of the distance shown on the horizontal axis. The black crosses show the lower bound on the right-hand side of (4.6), with $p = 1$. The blue and red lines show the corresponding uncertainty quantity on the left-hand side of (4.6), for the graph signals $\boldsymbol{\delta}_1$ and $\boldsymbol{\delta}_{10}$, respectively.

4.2.3 The Hausdorff-Young inequalities for signals on graphs

The classical Hausdorff-Young inequality [98, Section IX.4] is a fundamental harmonic analysis result behind the intuition that a high degree of concentration of a signal in one domain (time or frequency) implies a low degree of concentration in the other domain. This relation is used in the proofs of the entropy and ℓ^p-norm uncertainty principles in the continuous setting. In this section, as we continue to explore the role of $\mu_{\mathcal{G}}$ and the differences between the Euclidean and graph settings, we extend the Hausdorff-Young inequality to graph signals.

Theorem 7. *Let $\mu_{\mathcal{G}}$ be the coherence between the graph Fourier and canonical bases of a graph $\mathcal{G}$. Let $p, q > 0$ be such that $\frac{1}{p} + \frac{1}{q} = 1$. For any signal $\boldsymbol{x} \in \mathbb{C}^N$ defined on $\mathcal{G}$ and $1 \leq p \leq 2$, we have*

$$\|\hat{\boldsymbol{x}}\|_q \leq \mu_{\mathcal{G}}^{1-\frac{2}{q}} \|\boldsymbol{x}\|_p . \tag{4.9}$$

Conversely, for $2 \leq p \leq \infty$, we have

$$\|\hat{\boldsymbol{x}}\|_q \geq \mu_{\mathcal{G}}^{1-\frac{2}{q}} \|\boldsymbol{x}\|_p . \tag{4.10}$$

The proof of Theorem 7, given in Sec. C.1, is an extension of the classical proof using the Riesz-Thorin interpolation theorem. In the classical (infinite dimensional) setting, the inequality only depends on p and q. On a graph, it depends on $\mu_{\mathcal{G}}$ and hence on the structure of the graph.

Dividing both sides of each inequality in Theorem 7 by $\|\boldsymbol{x}\|_2$ leads to bounds on the concentrations (or sparsity levels) of a graph signal and its graph Fourier transform.

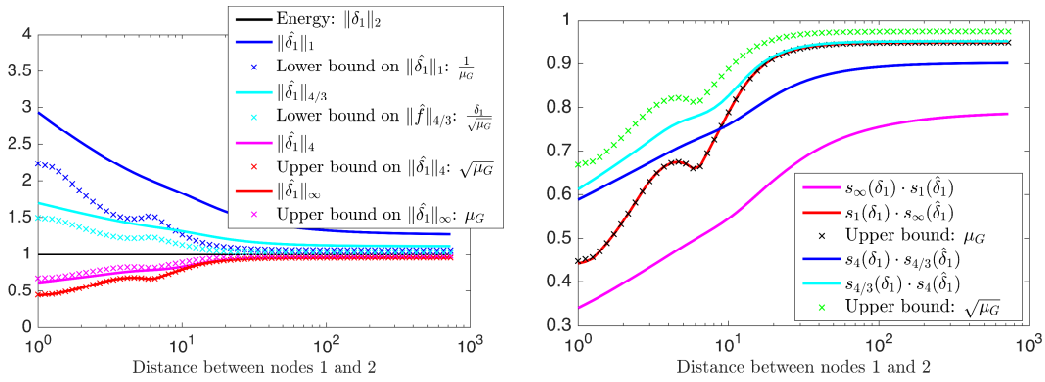
Corollary 1. *Let $p, q > 0$ be such that $\frac{1}{p} + \frac{1}{q} = 1$. For any signal $\boldsymbol{x} \in \mathbb{C}^N$ defined on the graph $\mathcal{G}$, we have*

$$s_p(\boldsymbol{x}) s_q(\hat{\boldsymbol{x}}) \leq \mu_{\mathcal{G}}^{\left|1-\frac{2}{q}\right|}.$$

Theorem 7 and Corollary 1 assert that the concentration or sparsity level of a graph signal in one domain (vertex or graph spectral) limits the concentration or sparsity level in the other domain. However, once again, if the coherence $\mu_{\mathcal{G}}$ is close to 1, the result is not particularly informative as $s_p(\boldsymbol{x}) s_q(\hat{\boldsymbol{x}})$ is trivially upper bounded by 1. The following numerical experiment illustrates the quantities involved in the Hausdorff-Young inequalities for graph signals. We again see that as the graph Fourier coherence increases, signals may be simultaneously concentrated in both the vertex domain and the graph spectral domain.

Example 15. *Continuing with the modified path graphs of Examples 13 and 14, we illustrate the bounds of the Hausdorff-Young inequalities for graph signals in Figure 4.6. For this example, we take the signal $\boldsymbol{x}$ to be $\boldsymbol{\delta}_1$, a Kronecker delta centered on the first node of the modified path graph. As a consequence, $\|\boldsymbol{\delta}_1\|_p = 1$ for all p, which makes it easier to compare the quantities*

involved in the inequalities. For this example, the bounds of Theorem 7 are fairly close to the actual values of $\|\hat{\boldsymbol{\delta}}_1\|_q$.



(a) (b)

Figure 4.6 – Illustration of the bounds of the Hausdorff-Young inequalities for graph signals on the modified path graphs with $\boldsymbol{x} = \boldsymbol{\delta}_1$. (a) The quantities in (4.9) and (4.10) for $q = 1, \frac{4}{3}, 4$, and ∞. (b) The quantities in Corollary 1 for the same values of q.

Sharpness of the graph Hausdorff-Young inequalities. For $p = q = 2$, (4.9) and (4.10) becomes equalities. Moreover, for $p = 1$ or $p = \infty$, there is always at least one signal for which the inequalities (4.9) and (4.10) become equalities, respectively. Let i_1 and ℓ_1 satisfy $\mu_{\mathcal{G}} = \max_{i,\ell} |\boldsymbol{u}_\ell[i]| = |\boldsymbol{u}_{\ell_1}(i_1)|$. For $p = 1$, let $\boldsymbol{x} = \boldsymbol{\delta}_{i_1}$. Then $\|\boldsymbol{x}\|_1 = 1$, and $\|\hat{\boldsymbol{x}}\|_\infty = \max_\ell |\langle \boldsymbol{\delta}_{i_1}, \boldsymbol{u}_\ell \rangle| = \mu_{\mathcal{G}}$, and thus (4.9) is tight. For $p = \infty$, let $\boldsymbol{x} = \boldsymbol{u}_{\ell_1}$. Then $\|\boldsymbol{x}\|_\infty = \mu_{\mathcal{G}}$, $\|\hat{\boldsymbol{x}}\|_1 = \left\|\widehat{\boldsymbol{u}_{\ell_1}}\right\|_1 = 1$, and thus (4.10) is tight. The red curve and its bound in Figure 4.6 show the tight case for $p = 1$ and $q = \infty$.

4.2.4 Limitations of global concentration-based uncertainty principles in the graph setting

The motivation for this sub-section was twofold. First, we want to derive the uncertainty principles for graph signals analogous to some of those that are so fundamental for signal processing on Euclidean domains. However, we also want to highlight the limitations of this approach (the family B of uncertainty principles described in Section 4.1.3) in the graph setting. The graph Fourier coherence is a global parameter that depends on the topology of the entire graph. Hence, it may be greatly influenced by a small localized changes in the graph structure. For example, in the modified path graph examples above, a change in a single edge weight leads to an increased coherence, and in turn significantly weakens the uncertainty principles characterizing the concentrations of the graph signal in the vertex and spectral domains. Such examples call into question the ability of such global uncertainty principles for graph signals to accurately describe phenomena in inhomogeneous graphs. This is the primary motivation for our investigation into local uncertainty principles in Section 4.4.1. However, before getting there, we consider global uncertainty principles from the family C of uncertainty principles

described in Section 4.1.3 that bound the concentration of the analysis coefficients of a graph signal in a time-frequency transform domain.

4.3 Global uncertainty principles

4.3.1 Some definitions

As mentioned in Section 4.1, uncertainty principles can inform dictionary design. In the next section, we present uncertainty principles characterizing the concentration of the analysis coefficients of graph signals in different transform domains. We focus on three different classes of dictionaries for graph signal analysis: (i) frames, (ii) graph filter-bank frames, and (iii) graph Gabor filter bank frames, where graph filter bank frames are a subclass of frames, and graph Gabor filter bank frames are a subclass of graph filter bank frames. In this section, we define these different classes of dictionaries, and highlight some of their mathematical properties. Note that our notation uses dictionary atoms that are doubly indexed by i and k, but these could be combined into a single index m for the most general case.

Definition 20 (Frame)**.** *A dictionary $\mathcal{D} = \{\boldsymbol{g}_{i,k}\}$ is a frame if there exist constants A and B called the lower and upper frame bounds such that for all $\boldsymbol{x} \in \mathbb{C}^N$:*

$$A\|\boldsymbol{x}\|_2^2 \leq \sum_{i,k} |\langle \boldsymbol{x}, \boldsymbol{g}_{i,k}\rangle|^2 \leq B\|\boldsymbol{x}\|_2^2.$$

If $A = B$, the frame is said to be a tight frame.

For more properties of frames, see, e.g., [20, 59, 60]. Most of the recently proposed dictionaries for graph signals are either orthogonal bases (e.g., [24, 77, 107]) , which are a subset of tight frames, or overcomplete frames (e.g., [52, 117, 115, 128]).

A frame is a graph filter bank frame if it is generated using the localization operator of multiples graph filters, i.e.,

$$\mathcal{D}_{\mathrm{g}} = \{\boldsymbol{g}_{i,k}\} = \{\mathcal{T}_i^G g_k\} \tag{4.11}$$

In order to be a frame g has to satisfy the criteria of Lemma 1, more specifically

$$G(\lambda_\ell) = \sum_{k=1}^{K} g_k(\lambda_\ell) > 0, \quad \ell = 0, 1, \ldots, N-1 \tag{4.12}$$

If the graph spectrum is not known, one can simply check the stronger condition $G(\lambda) > 0$ on the interval $[0, \lambda_{\max}]$. Note that, the frame constant of a graph filter bank frame defined as (4.11) are $A = \min_\ell G(\lambda_\ell)$ and $B = \max_\ell G(\lambda_\ell)$.

In order to generalize classical windowed Fourier frames, we often use a graph filter bank where the kernels are uniform translates, which we refer to as a graph Gabor filter bank.

Definition 21 (Graph Gabor filter bank)**.** *When the K kernels used to generate the graph filter bank frame are uniform translates of each other, we refer to the resulting dictionary as a* graph Gabor filter bank *or a* graph Gabor filter frame. *If we use the warping technique of [115] on these uniform translates, we refer to the resulting dictionary as a* spectrum-adapted graph Gabor filter frame.

Graph Gabor filter banks are generalizations of the short time Fourier transform. When g is smooth, the atoms are localized in the vertex domain. In this contribution, for all graph Gabor filter frames, we use the "itersine" mother window of (3.25). A few desirable properties of this choice of window are (a) it is perfectly localized in the spectral domain in $[-0.5, 0.5]$, (b) it is smooth enough to be approximated by a low order polynomial, and (c) the frame formed by uniform translates (with an even overlap) is tight.

The analysis operator of a dictionary $\mathcal{D} = \{\boldsymbol{g}_{i,k}\}$ to a signal $\boldsymbol{x} \in \mathbb{C}^N$ is referred as

$$(\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x})[i,k] = \langle \boldsymbol{x}, \boldsymbol{g}_{i,k} \rangle.$$

When $\mathcal{D} = \{\boldsymbol{g}_{i,k}\} = \{\mathcal{T}_i^G g_k\}$ is a graph filter bank frame, from Definition 11 we denote it with

$$\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x} = \langle \boldsymbol{x}, \mathcal{T}_i^G g_k \rangle. \tag{4.13}$$

In all cases, we view $\boldsymbol{A}_{\mathcal{D}}$ as a linear operator from $\mathbb{C}^N$ to $\mathbb{C}^{|\mathcal{D}|}$ (we use bold notation as we consider the matrix operator), and thus we use $\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_p$ (or $\left\|\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}\right\|_p$) to denote a vector norm of the analysis coefficients.

4.3.2 Discrete version of Lieb's uncertainty principle

Lieb's uncertainty principle in the continuous one-dimensional setting [64] states that the cross-ambiguity function of a signal cannot be too concentrated in the time-frequency plane. In the following, we transpose these statements to the discrete periodic setting, and then generalize them to frames and signals on graphs. The following discrete version of Lieb's uncertainty principle is partially presented in [39, Proposition 2].

Theorem 8. *Define the discrete Fourier transform (DFT) as*

$$\hat{\boldsymbol{x}}[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \boldsymbol{x}[n] \exp\left(\frac{-j2\pi kn}{N}\right),$$

and the discrete windowed Fourier transform (or discrete cross-ambiguity function) as (see, e.g., [68, Section 4.2.3])

$$\boldsymbol{A}_{\mathcal{D}_{DWFT}}\boldsymbol{x}[u,k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \boldsymbol{x}[n]\boldsymbol{g}^*[n-u] \exp\left(\frac{-j2\pi kn}{N}\right).$$

For two discrete signals of period N, we have for $2 \leq p < \infty$

$$\left\| \boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x} \right\|_p = \left(\sum_{u=1}^{N} \sum_{k=0}^{N-1} |\boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x}[u,k]|^p \right)^{\frac{1}{p}} \leq N^{\frac{1}{p}-\frac{1}{2}} \|\boldsymbol{x}\|_2 \left\| \boldsymbol{g} \right\|_2, \tag{4.14}$$

and for $1 \leq p \leq 2$

$$\left\| \boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x} \right\|_p = \left(\sum_{u=1}^{N} \sum_{k=0}^{N-1} |\boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x}[u,k]|^p \right)^{\frac{1}{p}} \geq N^{\frac{1}{p}-\frac{1}{2}} \|\boldsymbol{x}\|_2 \left\| \boldsymbol{g} \right\|_2. \tag{4.15}$$

These inequalities are proven in Appendix C.2.2. Note that the minimizers of this uncertainty principle are the so-called "picket fence" signals, trains of regularly spaced diracs.

4.3.3 Generalization of Lieb's uncertainty principle to frames

Theorem 9. *Let $\mathcal{D} = \{\boldsymbol{g}_{i,k}\}$ be a frame of atoms in $\mathbb{C}^N$, with lower and upper frame bounds A and B, respectively. For any signal $\boldsymbol{x} \in \mathbb{C}^N$ and any $p \geq 2$, we have*

$$\|\boldsymbol{A}_{\mathcal{D}} \boldsymbol{x}\|_p \leq B^{\frac{1}{p}} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{1-\frac{2}{p}} \|\boldsymbol{x}\|_2. \tag{4.16}$$

For any signal $\boldsymbol{x} \in \mathbb{C}^N$ and any $1 \leq p \leq 2$, we have

$$\|\boldsymbol{A}_{\mathcal{D}} \boldsymbol{x}\|_p \geq A^{\frac{1}{p}} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{1-\frac{2}{p}} \|\boldsymbol{x}\|_2. \tag{4.17}$$

Combining (4.16) *and* (4.17)*, for any $p \in [1,\infty]$, we have*

$$s_p(\boldsymbol{A}_{\mathcal{D}} \boldsymbol{x}) \leq \frac{B^{\min\{\frac{1}{2},\frac{1}{p}\}}}{A^{\max\{\frac{1}{2},\frac{1}{p}\}}} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{\left|1-\frac{2}{p}\right|}. \tag{4.18}$$

When $\mathcal{D}$ is a tight frame with frame bound A, (4.18) *reduces to*

$$s_p(\boldsymbol{A}_{\mathcal{D}} \boldsymbol{x}) \leq A^{-\left|\frac{1}{2}-\frac{1}{p}\right|} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{\left|1-\frac{2}{p}\right|}.$$

A proof is included in Section C.2.1 of the Appendix. The proof of Theorem 8 in Section C.2.2 of the Appendix also demonstrates that this uncertainty principle is indeed a generalization of the discrete periodic variant of Lieb's uncertainty principle.

4.3.4 Lieb's uncertainty principle for graph filter bank frames

Lemma 2 implies that $\max_{i,k} \left\| \mathcal{T}_i^G g_k \right\|_2 \leq \mu_{\mathcal{G}} \max_k \left\| g_k(\boldsymbol{\lambda}) \right\|_2$. Therefore the following is a corollary to Theorem 9 for the case of graph filter bank frames.

Theorem 10. *Let $\mathcal{D}_{\mathrm{g}} = \{\boldsymbol{g}_{i,k}\} = \{\mathcal{T}_i^G g_k\}$ be a graph filter bank frame of atoms on a graph $\mathcal{G}$ generated from the sequence of filters $\mathrm{g} = \{g_1, g_2, \ldots, g_K\}$. For any signal $\boldsymbol{x} \in \mathbb{C}^N$ on $\mathcal{G}$ and for any $p \in [1, \infty]$, we have*

$$s_p(\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}) \leq \frac{B^{\min\{\frac{1}{2},\frac{1}{p}\}}}{A^{\max\{\frac{1}{2},\frac{1}{p}\}}} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{\left|1-\frac{2}{p}\right|} \leq \frac{B^{\min\{\frac{1}{2},\frac{1}{p}\}}}{A^{\max\{\frac{1}{2},\frac{1}{p}\}}} \left(\mu_{\mathcal{G}} \max_k \left\| g_k(\boldsymbol{\lambda}) \right\|_2 \right)^{\left|1-\frac{2}{p}\right|}, \tag{4.19}$$

where $A = \min_\ell G(\lambda_\ell)$ is the lower frame bound and $B = \max_\ell G(\lambda_\ell)$ is the upper frame bound. When $\mathcal{D}$ is a tight frame with frame bound A, (4.19) *reduces to*

$$s_p(\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}) \leq A^{-\left|\frac{1}{2}-\frac{1}{p}\right|} \left(\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 \right)^{\left|1-\frac{2}{p}\right|} \leq A^{-\left|\frac{1}{2}-\frac{1}{p}\right|} \left(\mu_{\mathcal{G}} \max_k \left\| g_k(\boldsymbol{\lambda}) \right\|_2 \right)^{\left|1-\frac{2}{p}\right|}. \tag{4.20}$$

The bounds depend on the frame bounds A and B, which are fixed with the design of the filter bank. However, in the tight frame case, we can choose the filters in a manner such that the bound A does not depend on the graph structure. For example, if the g_k are defined continuously on the interval $[0, \lambda_{\max}]$ and $\sum_{k=0}^{M-1} \left| g_k(\lambda) \right|^2$ is equal to a constant for all λ, A is not affected by a change in the values of the Laplacian eigenvalues, e.g., from a change in the graph structure. The second quantity, $\max_{i,k} \left\| \boldsymbol{g}_{i,k} \right\|_2 = \max_{i,k} \left\| \mathcal{T}_i^G g_k \right\|_2$, reveals the influence of the graph. The maximum ℓ_2-norm of the atoms depends on the filter design, but also, as discussed previously in Section 3, on the graph topology. However, the bound is not local as it depends on the maximum $\left\| \mathcal{T}_i^G g_k \right\|_2$ over all localizations i and filters k, which takes into account the entire graph structure.

The second bounds in (4.19) and (4.20) also suggest how the filters can be designed in order to improve the uncertainty bound. The quantity $\left\| g_k(\boldsymbol{\lambda}) \right\|_2 = \left(\sum_\ell |g_k(\lambda_\ell)|^2 \right)^{\frac{1}{2}}$ depends on the distribution of the eigenvalues λ_ℓ, and, as consequence, on the graph structure. However, the distribution of the eigenvalues can be taken into account when designing the filters in order to reduce or cancel this dependency [115].

In the following example, we compute the first uncertainty bound in (4.20) for different types of graphs and filters. It provides some insight on the influence of the graph topology and filter bank design on the uncertainty bound.

Example 16. *We use the techniques of [115] to construct four tight localized spectral graph filter frames for each of eight different graphs. Figure 4.7 shows an examples of the four sets of filters for a 64 node sensor network. For each graph, two of the sets of filters, (b) and (d) in Figure 4.7, are adapted via warping to the distribution of the graph Laplacian eigenvalues so that each filter contains an appropriate number of eigenvalues (roughly equal in the case of translates and roughly logarithmic in the case of wavelets). The warping excludes filters containing zero*

or very few eigenvalues at which the filter has a nonzero value. These tight frames are designed such that $A = 1$*, and thus Theorem 10 yields*

$$s_\infty(\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}) = \frac{\left\|\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}\right\|_\infty}{\left\|\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}\right\|_2} \leq \max_{i,k} \left\|\mathcal{T}_i^G g_k\right\|_2 \leq \mu_G \max_k \left\|g_k(\boldsymbol{\lambda})\right\|_2 .$$

Table 4.1 displays the values of the first concentration bound $\max_{i,k} \left\|\mathcal{T}_i^G g_k\right\|_2$ *for each graph and frame pair. The uncertainty bound is largest when the graph is far from a regular lattice (ring or path). As expected, the worst cases are for highly inhomogeneous graphs like the comet graph or a modified path graph with one isolated vertex. The choice of the filter bank may also decrease or increase the bound, depending on the graph.*

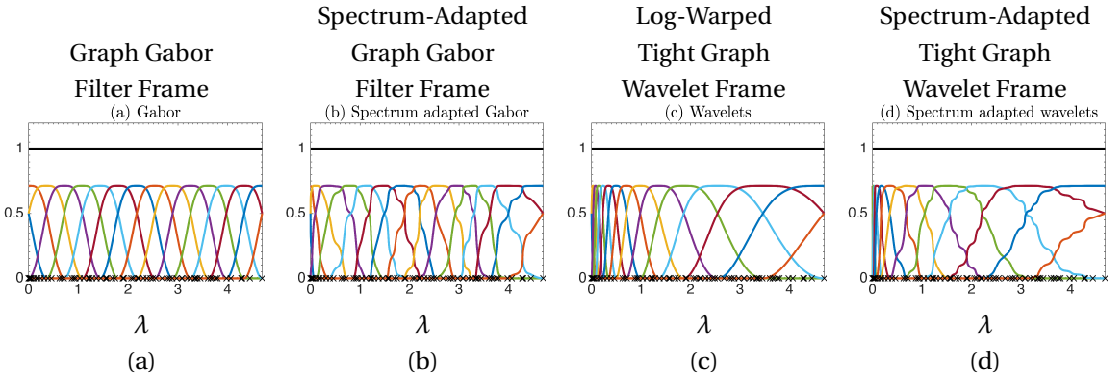


Figure 4.7 – Four different filter bank designs of [115], shown for a random sensor network with 64 nodes. Each colored curve is a filter defined continuously on $[0, \lambda_{\max}]$, and each filter bank has 16 such filters. They are designed such that $G(\lambda) = 1$ for all λ (black line), and thus all four designs yield tight localized spectral graph filter frames. The frame bounds here are $A = B = N$.

Graph	μ_G	Uniform Translate Graph Gabor	Spectrum-Adapted Graph Gabor	Log-Warped Wavelets	Spectrum-Adapted Wavelets
Ring	0.12	0.33	**0.28**	0.44	0.45
Random sensor network	0.90	0.70	0.69	**0.68**	0.69
Random regular	0.43	0.41	**0.40**	0.57	0.53
Erdos Renyi	0.93	0.68	0.68	0.68	**0.67**
Comet	0.98	0.70	0.70	0.70	0.70
Path	0.18	0.45	**0.38**	0.51	0.51
Modified path: $W_{12} = 0.1$	0.48	0.69	0.66	**0.57**	0.58
Modified path: $W_{12} = 0.01$	0.70	0.71	0.68	0.70	**0.65**

Table 4.1 – Numerical values of the uncertainty bound $\max_{i,k} \left\|\mathcal{T}_i^G g_k\right\|_2$ of Example 16 for various graphs of 64 nodes.

The uncertainty principle in Theorem 10 bounds the concentration of the graph Gabor transform coefficients. In the next example, we examine these coefficients for a series of signals with different vertex and spectral domain localization properties.

Example 17 (Concentration of the graph Gabor coefficients for signals with varying vertex and spectral domain concentrations.)**.** *In Figure 4.8, we analyze a series of signals on a random sensor network of 100 vertices. Each signal is created by localizing a kernel* $h_\tau(\lambda) = e^{-\frac{\lambda^2}{\lambda_{max}^2}\tau^2}$ *to be centered at vertex 1 (circled in black). To generate the four different signals, we vary the value of the parameter* τ *in the heat kernel. We plot the four localized kernels in the graph spectral and vertex domains in the first two columns, respectively. The more we "compress" h in the graph spectral domain (i.e., we reduce its spectral spreading by increasing* τ*), the less concentrated the localized atom becomes in the vertex domain. The joint vertex-frequency representation* $|(\boldsymbol{A}_g \mathcal{T}_1^G h_\tau)[i,k]|$ *of each signal is shown in the third column, which illustrates the trade-off between concentration in the vertex and the spectral domains. The concentration of these graph Gabor transform coefficients is the quantity bounded by the uncertainty principle presented in Theorem 10. In the last row of the Figure 4.8,* $\tau = \infty$*, which leads to a Kronecker delta for the kernel and a constant on the vertex domain. On the contrary, when the kernel is constant, with* $\tau = 0$ *(top row), the energy of the graph Gabor coefficients stays concentrated around one vertex but spreads along all frequencies.*

4.4 Local uncertainty principles

In the previous section, we defined a global bound for the concentration of the localized spectral graph filter frame analysis coefficients. In the classical setting, such a global bound is also local in the sense that each part of the domain has the same structure, due to the regularity of the underlying domain. However, this is not the case for the graph setting where the domain is irregular. Example 13 shows that a "bad" structure (a weakly connected node) in a small region of the graph reduces the uncertainty bound even if the rest of the graph is well behaved. Functions localized near the weakly connected node can be highly concentrated in both the vertex and frequency domains, whereas functions localized away from it are barely impacted. Importantly, *the worst case determines the global uncertainty bound.* As another example, suppose one has two graphs G_1 and G_2 with two different structures, each of them having a different uncertainty bound. The uncertainty bound for the graph $\mathcal{G}$ that is the union of these two disconnected graphs is the minimum of the uncertainty bounds of the two disconnected graphs, which is suboptimal for one of the two graphs.

In this section, we ask the following questions. Where does this worse case happen? Can we find a local principle that characterizes more accurately the uncertainty in other parts of the graph? In order to answer this question, we investigate the concentration of the analysis coefficients of the frame atoms, which are localized signals in the vertex domain. This technique is used in the classical continuous case by Lieb [64], who defines the (cross-) ambiguity function, the STFT of a short-time Fourier atom. The result is a joint time-frequency uncertainty principle that does not depend on the localization in time or in frequency of the analyzed atom.

Thus, we start by generalizing the definition of ambiguity (or cross-ambiguity) functions from

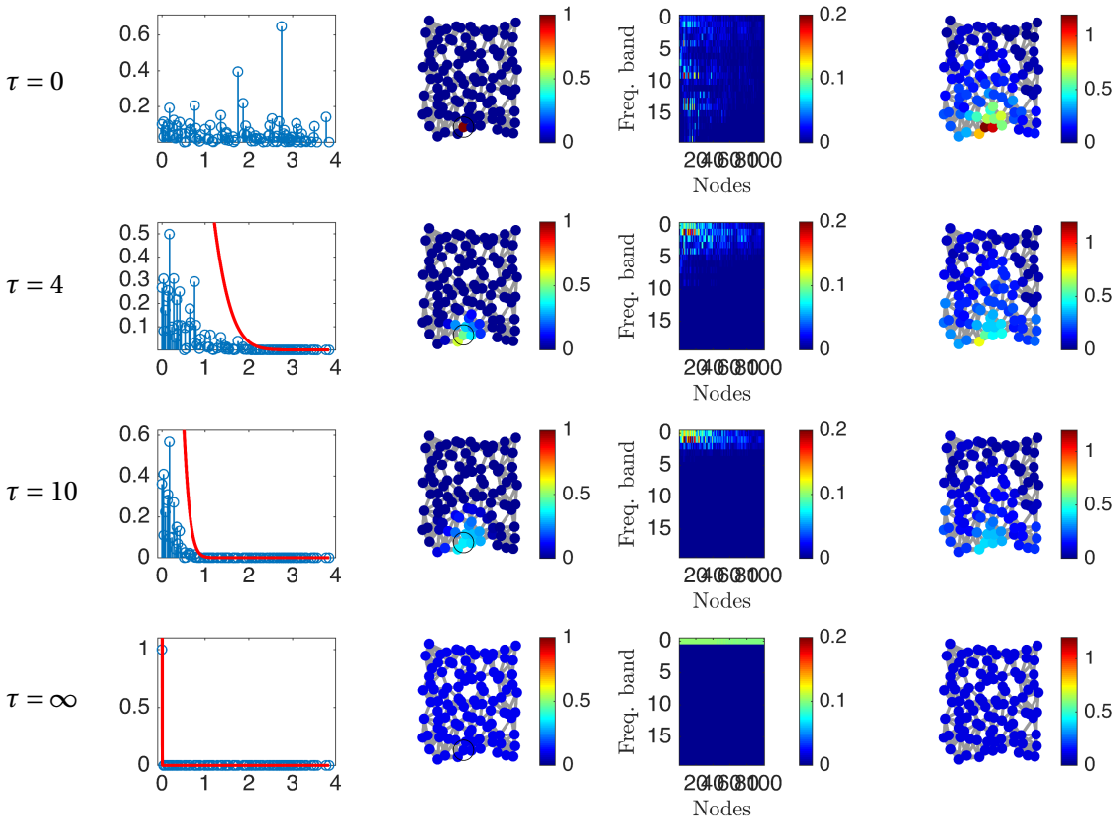


Figure 4.8 – Graph Gabor transform of four different signals $\boldsymbol{x}_\tau = \mathcal{T}_1^G h_\tau$, with each row corresponding to a signal with a different value of the parameter τ. Each of the signals is a kernel localized to vertex 1, with the kernel to be localized equal to $h_\tau(\lambda) = e^{-\frac{\lambda^2}{\lambda_{\max}^2}\tau^2}$. The underlying graph is a random sensor network of 100 vertices. First column: the kernel $h_\tau(\lambda)$ is shown in red and the localized kernel $\widehat{\boldsymbol{x}_\tau}$ is shown in blue, both in the graph spectral domain. Second column: the signal $\boldsymbol{x}_\tau$ in the vertex domain (the center vertex 1 is circled). Third column: $|(\boldsymbol{A}_g \mathcal{T}_1^G h_\tau)[i,k]|$, the absolute value of the Gabor transform coefficients for each vertex i and each of the 20 frequency bands k. Fourth column: since it is hard to see where on the graph the transform coefficients are concentrated when the nodes are placed on a line in the third column, we display the value $\sum_{k=0}^{19} |(\boldsymbol{A}_g \mathcal{T}_1^G h_\tau)[i,k]|$ on each vertex i in the network. This figure illustrates the tradeoff between the vertex and the frequency concentration.

time-frequency analysis of one-dimensional signals to the graph setting.

Definition 22 (Ambiguity function)**.** *The ambiguity function of a localized spectral frame* $\mathcal{D} = \{\boldsymbol{g}_{i,k}\} = \{\mathcal{T}_i^G g_k\}$ *is defined as:*

$$\mathbb{A}_g[i_0, k_0, i, k] = (\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0})[i,k] = \langle \mathcal{T}_{i_0}^G g_{k_0}, \mathcal{T}_i^G g_k \rangle$$

When the kernels $\{g_k\}_{k=0,1,\ldots,M-1}$ are appropriately warped uniform translates, the operator $\boldsymbol{A}_g$ becomes a generalization of the short time Fourier transform. Additionally, the ambiguity

function assesses the degree of coherence (linear dependence) between the atoms $\mathcal{T}_{i_0}^G g_{k_0}$ and $\mathcal{T}_i^G g_k$. In the following, we use this ambiguity function to probe *locally* the structure of the graph, and derive local uncertainty principles.

4.4.1 Local uncertainty principle

In order to probe the local uncertainty of a graph, we take a set of localized kernels in the graph spectral domain and center them at different local regions of the graph in the vertex domain. The atoms resulting from this construction are jointly localized in both the vertex and graph spectral domains, where "localized" means that the values of the function are zero or close to zero away from some reference point. By ensuring that the atoms are localized or have support within a small region of the graph, we focus on the properties of the graph in that region. In order to get a local uncertainty principle, we apply the frame operator to these localized atoms, and analyze the concentration of the resulting coefficients. In doing so, we develop an uncertainty principle relating these concentrations to the local graph structure.

To prepare for the theorem, we first state a lemma that gives a hint to how the scalar product of two localized functions depends on the graph structure and properties. In the following, we multiply two kernels g and h in the graph spectral domain. For notation, we represent the product of these two kernels in vertex domain as $g \cdot h$.

Lemma 4. *For two kernels g, h and two nodes i, n, the localization operator satisfies*

$$\langle \mathcal{T}_i^G g, \mathcal{T}_n^G h \rangle \quad = \quad \mathcal{T}_i^G (g \cdot h)[n] \tag{4.21}$$

and

$$\left(\sum_i \left| \langle \mathcal{T}_i^G g, \mathcal{T}_n^G h \rangle \right|^p \right)^{\frac{1}{p}} = \left\| \mathcal{T}_n^G (g \cdot h) \right\|_p . \tag{4.22}$$

The proof is deferred to Appendix C.3.1 Equation (4.21) shows more clearly the conditions on the kernels and nodes under which the scalar product is small. Let us take two examples. First, suppose g and h have a compact support on the spectrum and do not overlap (kernels localized in different places), then $g \cdot h$ is zero everywhere on the spectrum, and therefore the scalar product on the left-hand side of (4.21) is also equal to zero. Second, assume i and n are distant from each other. Then $|\mathcal{T}_i^G (g \cdot h)[n]|$ is small if g and h are reasonably smooth. In other words, the two atoms $\mathcal{T}_i^G g$ and $\mathcal{T}_n^G h$ must be localized both in the same area of graph in the vertex domain and the same spectral region in order for the scalar product to be large. This localization depends on the atoms, but also on the graph structure.

The following theorem provides inequalities giving a local uncertainty principle. The local bound depends of the localization of the atom $\mathcal{T}_{i_0}^G g_{k_0}$ both in the graph and spectral domains.

The center vertex i_0 and kernel g_{k_0} can be chosen to be any vertex and kernel; however, the locality property of the uncertainty principle appears when ${\mathcal{T}_{i_0}}^G g_{k_0}$ is concentrated around node i_0 in the vertex domain and around a small portion of the spectrum in the graph spectral domain. Once again, we measure the concentration with ℓ^p-norms.

Theorem 11 (Local uncertainty)**.** *Let $\{{\mathcal{T}_i}^G g_k\}_{i\in[1,N],k\in[0,M-1]}$ be a localized spectral graph filter frame with lower frame bound A and upper frame bound B. For any $i_0 \in [1,N]$, $k_0 \in [0,M-1]$ such that $\left\| {\mathcal{T}_{i_0}}^G g_{k_0} \right\|_2 > 0$, the quantity*

$$\left\| \boldsymbol{A}_{\mathrm{g}} {\mathcal{T}_{i_0}}^G g_{k_0} \right\|_p = \left(\sum_{k=1}^{M} \sum_{i=1}^{N} \left| \langle {\mathcal{T}_i}^G g_k, {\mathcal{T}_{i_0}}^G g_{k_0} \rangle \right|^p \right)^{\frac{1}{p}} = \left(\sum_{k=1}^{M} \left\| {\mathcal{T}_{i_0}}^G (g_{k_0} \cdot g_k) \right\|_p^p \right)^{\frac{1}{p}} \tag{4.23}$$

satisfies, for $p \in [1,\infty]$,

$$s_p\left(\boldsymbol{A}_{\mathrm{g}} {\mathcal{T}_{i_0}}^G g_{k_0} \right) \leq \frac{B^{\min\{\frac{1}{p},1-\frac{1}{p}\}} \left\| {\mathcal{T}_{\tilde{i}_{i_0,k_0}}}^G g_{\tilde{k}_{i_0,k_0}} \right\|_2^{\left|1-\frac{2}{p}\right|}}{A^{\frac{1}{2}}} \leq \frac{B^{\min\{\frac{1}{p},1-\frac{1}{p}\}} \left(\nu_{\tilde{i}_{i_0,k_0}} \left\| g_{\tilde{k}_{i_0,k_0}} \right\|_2 \right)^{\left|1-\frac{2}{p}\right|}}{A^{\frac{1}{2}}}, \tag{4.24}$$

where ν_i is defined in Lemma 2,

$$\tilde{k}_{i_0,k_0} = \operatorname*{argmax}_k \left\| {\mathcal{T}_{i_0}}^G (g_{k_0} \cdot g_k) \right\|_\infty, \text{ and } \tilde{i}_{i_0,k_0} = \operatorname*{argmax}_i \left| {\mathcal{T}_{i_0}}^G (g_{k_0} \cdot g_{\tilde{k}_{i_0,k_0}})[i] \right|.$$

The proof is deferred in Appendix C.3.2.

The bound in (4.24) is local, because we get a different bound for each (i_0, k_0) pair. For each such pair, the bound depends on the quantities $\tilde{i}_{i_0,k_0}, \tilde{k}_{i_0,k_0}$, which are maximizers over a set of all vertices and kernels, respectively; however, as we discuss in Example 18 below, $\tilde{i}_{i_0,k_0}$ is typically close to i_0, and $\tilde{k}_{i_0,k_0}$ is typically close to k_0. For this reason, this bound typically depends only on local quantities.

The next corollary shows that in many cases, the local uncertainty inequality (4.24) is sharp (i.e., it becomes an equality). To obtain this, we require that the frame $\boldsymbol{A}_{\mathrm{g}}$ is tight and that $|\langle {\mathcal{T}_i}^G g_k, {\mathcal{T}_{i_0}}^G g_{k_0} \rangle|$ is maximized when $k = k_0$ and $i = i_0$.

Corollary 2. *Under the assumptions of Theorem 11 and, assuming additionally that*

1. *$\boldsymbol{A}_{\mathrm{g}}$ is a tight frame with frame-bound A,*
2. *$k_0 = \operatorname{argmax}_k \left\| {\mathcal{T}_{i_0}}^G (g_k \cdot g_{k_0}) \right\|_\infty$, and*
3. *$i_0 = \operatorname{argmax}_n |{\mathcal{T}_{i_0}}^G g_{k_0}^2[n]|$,*

we have

$$s_\infty\left(\boldsymbol{A}_{\mathrm{g}} {\mathcal{T}_{i_0}}^G g_{k_0} \right) = \frac{\left\| \boldsymbol{A}_{\mathrm{g}} {\mathcal{T}_{i_0}}^G g_{k_0} \right\|_\infty}{\left\| \boldsymbol{A}_{\mathrm{g}} {\mathcal{T}_{i_0}}^G g_{k_0} \right\|_2} = \frac{\left\| {\mathcal{T}_{i_0}}^G g_{k_0} \right\|_2}{A^{\frac{1}{2}}}. \tag{4.25}$$

The proof is deferred to Appendix C.3.3.

Corollary 3. *Under the assumptions of Theorem 11, we have*

$$s_\infty(\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}) = \frac{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty}{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_2} \geq \frac{\left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2}{B^{\frac{1}{2}}}, \tag{4.26}$$

which is a lower bound on the concentration measure.

The proof is given in Appendix C.3.4. Together, Theorem 11 and Corollary 3 yield lower and upper bounds on the local sparsity levels $s_\infty(\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0})$:

$$\frac{\left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}}\right\|_2}{A^{\frac{1}{2}}} \geq s_\infty(\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}) = \frac{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty}{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_2} \geq \frac{\left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2}{B^{\frac{1}{2}}}.$$

4.4.2 Illustrative examples

In order to better understand the above local uncertainty principle, we illustrate it with some examples.

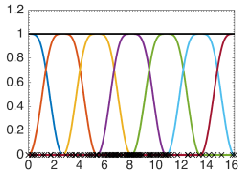
Example 18 (Local uncertainty on a sensor network)**.** *Let us concentrate on the case where $p = \infty$. Theorem 11 tells us that*

$$\frac{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty}{\left\|\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}\right\|_2} \leq \frac{\left\|\mathcal{T}_{\tilde{i}_{i_0,k_0}}^G g_{\tilde{k}_{i_0,k_0}}\right\|_2}{A^{\frac{1}{2}}} \leq \frac{\sqrt{N}\nu_{\tilde{i}_{i_0,k_0}}\left\|g_{\tilde{k}_{i_0,k_0}}\right\|_2}{A^{\frac{1}{2}}}, \tag{4.27}$$

meaning that the concentration of $\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}$ is limited by $\frac{1}{\left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}_{i_0,k_0}}\right\|_2}$. One question is to what extent this quantity is local or reflects the local behavior of the graph. As a general illustration for this discussion, we present in Figure 4.9 quantities related to the local uncertainty of a random sensor network of 100 *nodes evaluated for two different values of k (one in each column) and all nodes i.*

The first row (not counting the top figure) shows the local sparsity levels of $\boldsymbol{A}_\mathrm{g}\mathcal{T}_{i_0}^G g_{k_0}$ in terms of the ℓ^∞-norm (left hand side of (4.27)*) at each node of the graph. The second row shows the values of the upper bound on local sparsity for each node of the graph (middle term of* (4.27)*). The values of both rows are strikingly close. Note that for this type of graph, local sparsity/concentration is lowest where the nodes are well connected.*

We focus now on the values of $\tilde{k}$ and $\tilde{i}$ as they are crucial in Theorem 11. We also give insights that explain when a tight bound is obtained, as stated in Corollary 2. There is not a simple way to determine the value of $\tilde{k}$, because it depends not only on the node i_0 and the filters g_k, but also on the graph Fourier basis. However, the definition $\tilde{k} = \mathrm{argmax}_k \left\|\mathcal{T}_{i_0}^G(g_k \cdot g_{k_0})\right\|_\infty$ implies that



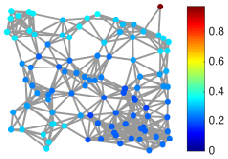
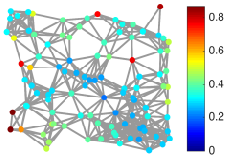
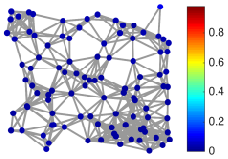
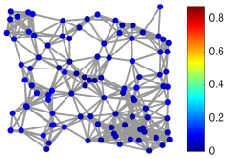
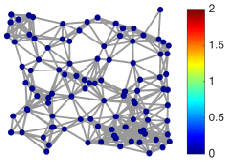
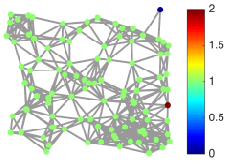
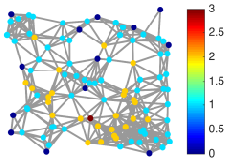
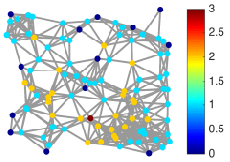
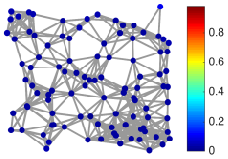
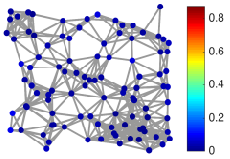
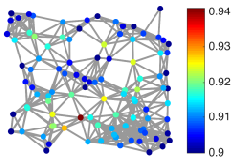
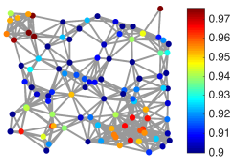
	First filter $k_0 = 0$ (blue)	Second Filter $k_0 = 1$ (orange)
Graph Gabor filter bank		
Local sparsity level: $s_\infty(\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}) = \frac{\left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty}{\left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_2}$		
Upper bound on local sparsity from Theorem 11: $\frac{\left\|\mathcal{T}_{\tilde{i}_{i_0,k_0}}^G g_{\tilde{k}_{i_0,k_0}}\right\|_2}{A^{\frac{1}{2}}}$		
Maximizing filter index: $\tilde{k}_{k_0,i_0}$		
Hop distance between $\tilde{i}_{k_0,i_0}$ and i_0		
Lower bound on the local sparsity from Corollary 3: $\frac{\left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2}{B^{\frac{1}{2}}}$		
Relative error between $s_\infty(\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0})$ and $B^{-\frac{1}{2}}\left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2$: $\frac{s_\infty(\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}) - B^{-\frac{1}{2}}\left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2}{s_\infty(\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0})}$		

Figure 4.9 – Illustration of Theorem 11 and related variables $\tilde{i}$ and $\tilde{k}$ for a random sensor graph of 100 nodes. Top figure: the 8 uniformly translated kernels $\{g_k\}_k$ (in 8 different colors) defined on the spectrum and giving a tight frame. Each row corresponds to quantities related to the local uncertainty principle. The first column concerns the kernel (filter) in blue on the top figure, the second is associated with the orange one. On a sensor graph, the local uncertainty level (inversely proportional to the local sparsity level plotted here) is far from constant from one node to another or from one frequency band to another.

the two kernels $g_{\tilde{k}}$ and g_{k_0} have to overlap "as much as possible" in the graph Fourier domain in order to maximize the infinity-norm. In the case of a Gabor filter bank like the one presented in the first line of Figure 4.9, $k_0 = \tilde{k}$ for most of the nodes. This happens because the filters g_k and g_{k_0} do not overlap much if $k \neq k_0$, i.e when

$$\left\| g_{k_0}^2(\boldsymbol{\lambda}) \right\|_2^2 = \sum_\ell \left(g_{k_0}^2(\lambda_\ell) \right)^2 \text{»} \sum_\ell \left(g_{k_0}(\lambda_\ell) g_k(\lambda_\ell) \right)^2 = \left\| (g_k \cdot g_{k_0})(\boldsymbol{\lambda}) \right\|_2^2.$$

In fact, in the case of Figure 4.9, $\tilde{k}$ is bounded between $k_0 - 1$ and $k_0 + 1$ because there is no overlap with the other filters. In Figure 4.9, we plot $\tilde{k}[i]$ for $k_0 = 0$ and $k_0 = 1$. For the first filter, we have $\tilde{k}_{i_0,k_0} = k_0$ for all vertices i_0. The second filter follows the same rule except for two nodes. The isolated node on the north east is less connected to the rest and there is a Laplacian eigenvector that is well localized on it. As a consequence, the localization on the graph is affected in a counter-intuitive manner.

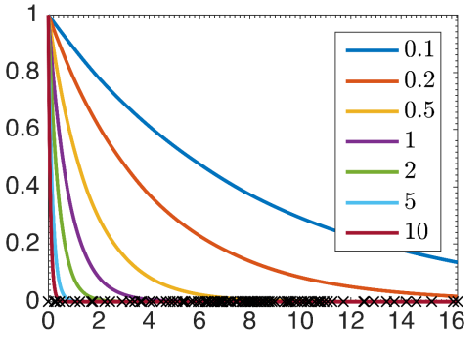
Let us now concentrate on the second important variable: $\tilde{i}$. Under the assumption that the kernels g_k are smooth, the energy of localized atoms $\mathcal{T}_{i_0}^G g_k$ reside inside a ball centered at i_0 [117]. Thus, the node n maximizing $|\mathcal{T}_{i_0}^G (g_{k_0} g_{\tilde{k}})[n]|$ cannot be far from the node i_0. Let us use the hop distance $h_{\mathcal{G}}(v_i, v_n)$ (See Definition 6). If the kernels g_k are polynomial functions of order K, the localization operator $\mathcal{T}_{i_0}^G g$ concentrates all of the energy of $\mathcal{T}_{i_0}^G g_k$ inside a K-radius ball centered in i_0. Since the resulting kernel $g_{k_0} g_{\tilde{k}}$ is a polynomial of order $2K$, $\tilde{i}$ will be at a distance of at most $2K$ hops from the node i_0. In general, $\tilde{i}$ is close to i_0. In fact, the distance $h_{\mathcal{G}}(v_{i_0}, v_{\tilde{i}})$ is related to the smoothness of the kernel $g_{k_0} g_{\tilde{k}}$ [117]. To illustrate this effect, we present in Figure 4.10 the average and maximum hop distance $h_{\mathcal{G}}(v_{i_0}, v_{\tilde{i}})$. In this example, we control the concentration of a kernel g with a dilation parameter a: $g_a(x) = g(ax)$. Increasing a compresses the kernel in the Fourier domain and increases the spread of the localized atoms in the vertex domain. Note that even for high spectral compression, the hop distance $h_{\mathcal{G}}(v_{i_0}, \tilde{i})$ remains low. Additionally, we also compute the mean relative error between $\left\| \mathcal{T}_{i_0}^G g^2 \right\|_\infty$ and $|\mathcal{T}_{i_0}^G g^2[i_0]|$. This quantity asserts how well $\left\| \mathcal{T}_{i_0}^G g \right\|_2^2$ estimates $\left\| \mathcal{T}_{i_0}^G g^2 \right\|_\infty$.[2] Returning to Figure 4.9, the fourth row shows the hop distance between i_0 and $\tilde{i}$. It never exceeds 3 for both the first and the second filter, which is a good sign of locality.

In practice we can not always determine the values of $\tilde{k}$ and $\tilde{i}$, but as we have seen, the quantity $B^{-\frac{1}{2}} \left\| \mathcal{T}_i^G g_{k_0} \right\|_2$ may still be a good estimate of the local sparsity level. Row 5 of Figure 4.9 shows these estimates, and the last row shows the relative error between these estimates and the actual local sparsity levels. We observe that for the first kernel, the estimate gives a sufficiently rough approximation of the local sparsity levels. For the second kernel, the approximation error is low for most of the nodes, but not all.

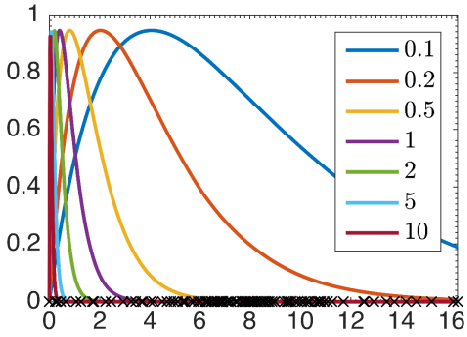
In the next example, we compare the local and global uncertainty principles on a modified path graph.

[2] From Lemma 4, when $\left\| \mathcal{T}_{i_0}^G g^2 \right\|_\infty = |\mathcal{T}_{i_0}^G g^2[i_0]|$, then $\left\| \mathcal{T}_{i_0}^G g^2 \right\|_\infty = \left\| \mathcal{T}_{i_0}^G g \right\|_2^2$.

(a) Heat kernel



(b) Wavelet kernel



a	mean$_i$ $\frac{\|\mathcal{T}_i^G g\|_\infty - \|\mathcal{T}_i^G g[i]\|}{\|\mathcal{T}_i^G g\|_\infty}$ in %	mean$_i$ $h_{\mathcal{G}}(v_{\tilde{i}}, v_i)$	max$_i$ $h_{\mathcal{G}}(v_{\tilde{i}}, v_i)$
Heat kernel			
0.1	0	0	0
0.2	0	0	0
0.5	2.01	0.28	1
1	5.87	0.89	2
2	7.45	1.39	3
5	8.59	2.04	4
10	2.63	2.08	4
Wavelet kernel			
0.1	0	0	0
0.2	0	0	0
0.5	9.03	0.62	1
1	10.99	1.07	2
2	17.69	1.67	3
5	29.67	2.07	4
10	33.45	2.48	6

Figure 4.10 – Localization experiment using the sensor graph of Figure 4.9. The heat kernel is defined as $g(ax) = e^{-\frac{10 \cdot ax}{\lambda_{\max}}}$ and the wavelet kernel $g(ax) = \sqrt{40} \cdot ax \cdot e^{-\frac{40 \cdot ax}{\lambda_{\max}}}$. For a smooth kernel g, the hop distance $h_{\mathcal{G}}$ between i and $\tilde{i} = \text{argmax}_n |\mathcal{T}_i^G g[n]|$ is small.

Example 19. *On a* 64*-node modified path graph (see Example 13 for details), we compute the graph Gabor transform of the signals* $\boldsymbol{x}_1 = \mathcal{T}_1^G g_0$ *and* $\boldsymbol{x}_2 = \mathcal{T}_{64}^G g_0$*. In Figure 4.11, we show the evolution of the graph Gabor transforms of the two signals with respect to the distance* $d = 1/W_{12}$ *from the first to the second vertex in the graph. As the first node is pulled away, a localized eigenvector appears centered on the isolated vertex. Because of this, as this distance increases, the signal* $\boldsymbol{x}_1$ *becomes concentrated in both the vertex and graph spectral domains, leading to graph Gabor transform coefficients that are highly concentrated (see the top right plot in Figure 4.11). However, since the graph modification is local, it does not drastically affect the graph Gabor transform coefficients of the signal* $\boldsymbol{x}_2$ *(middle row of Figure 4.11), whose energy is*

concentrated on the far end of the path graph.

In Figure 4.12, we plot the evolution of the uncertainty bounds as well as the concentration of the Gabor transform coefficients of $\boldsymbol{x}_1$ *and* $\boldsymbol{x}_2$*. The global uncertainty bound from Theorem 10 tells us that*

$$s_1(\boldsymbol{A}_{\mathrm{g}}\boldsymbol{x}) \leq \max_{i,k} \left\| \mathcal{T}_i^G g_k \right\|_2, \text{ for any signal } \boldsymbol{x}.$$

The local uncertainty bound from Theorem 11 tells us that

$$s_1(\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^G g_{k_0}) \leq \left\| T_{\tilde{i}_{i_0,k_0}} g_{\tilde{k}_{i_0,k_0}} \right\|_2, \text{ for all } i_0 \text{ and } k_0.$$

Thus, we can view the global uncertainty bound as an upper bound on all of the local uncertainty bounds. In fact the bumps in the global uncertainty bound in Figure 4.12 correspond to the local bound with $i_0 = 1$ *and different frequency bands* k_0*. We plot the local bounds for* $i_0 = 1$ *and* $k_0 = 0$ *and* $k_0 = 2$*.*

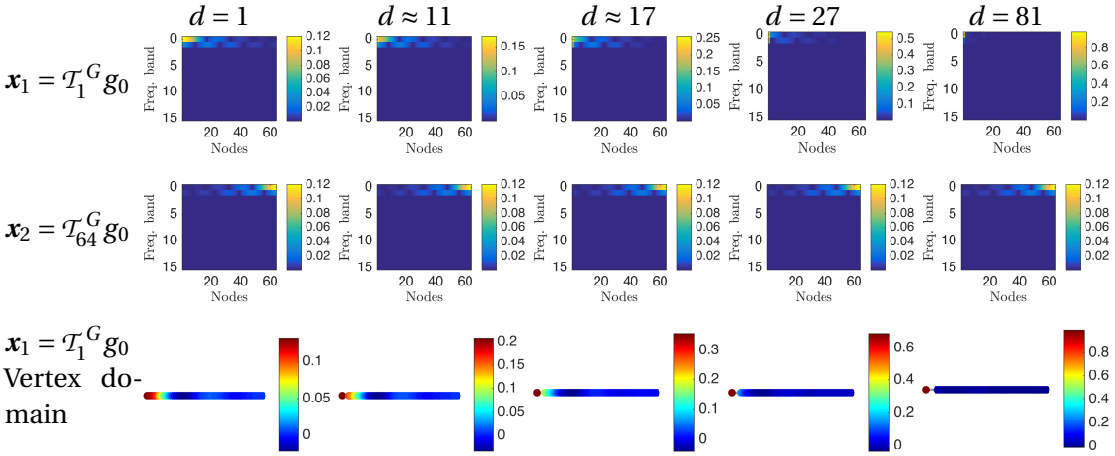


Figure 4.11 – Graph Gabor transforms of $\boldsymbol{x}_1 = \mathcal{T}_1^G g_0$ and $\boldsymbol{x}_2 = \mathcal{T}_{64}^G g_0$ for 5 different distances between vertices 1 and 2 of the modified path graph. The distance $d = 1/W_{12}$ is the inverse of the weight of the edge connecting the first two vertices in the path. The node 64 is not affected by the change in the graph structure, because its energy is concentrated on the opposite side of the path graph. The graph Gabor coefficients of $\boldsymbol{x}_1$, however, become highly concentrated as a graph Laplacian eigenvector becomes localized on vertex 1 as the distance increases. The bottom row shows that, as the distance between the first two vertices increases, the atom $\mathcal{T}_1^G g_0$ also converges to a Kronecker delta centered on vertex 1.

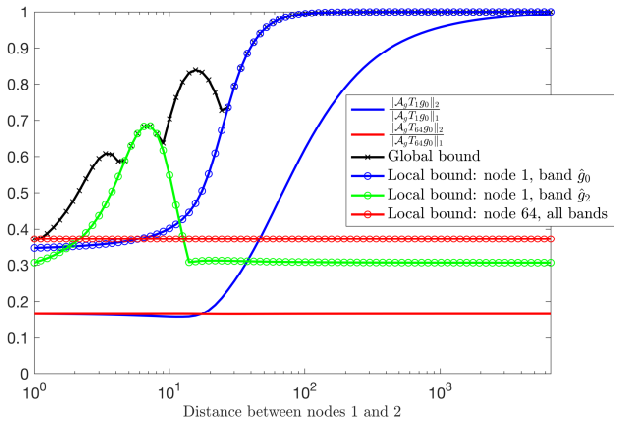


Figure 4.12 – Concentration of the graph Gabor coefficients of $\boldsymbol{x}_1 = \mathcal{T}_1^G g_0$ and $\boldsymbol{x}_2 = \mathcal{T}_{64}^G g_0$ with respect to the distance between the first two vertices in the modified path graph, along with the upper bounds on this concentration from Theorem 10 (global uncertainty) and Theorem 11 (local uncertainty). Each bump of the global bound corresponds to a local bound of a given spectral band of node 1. For clarity, we plot only the bands g_0 and g_2 for the node 1. For the node 64, the local bound is barely affected by the change in graph structure, and the sparsity levels of the graph Gabor transform coefficients of $\mathcal{T}_{64}^G g_0$ also do not change much.

4.4.3 Single kernel analysis

Let us focus on the case where we analyze a single kernel g. Such an analysis is relevant when we model the signal as a linear combination of different localizations of a single kernel:

$$\boldsymbol{x}[n] = \sum_{i=1}^{N} w_i \mathcal{T}_i^G g[n]$$

This model has been proposed in different contributions [82, 42, 138], and has also been used as an interpolation model, e.g., in [91] and [116, Section V.C]. In this case, we could ask the following question. If we measure the signal value at node n, how much information do we get about w_n? We can answer this by looking at the overlap between the atom $\mathcal{T}_n^G g$ and the other atoms. When $\mathcal{T}_n^G g$ has a large overlap with the other atoms, the value of $\boldsymbol{x}[n]$ does not tell us much about w_n. However, in the case where $\mathcal{T}_n^G g$ has a very small overlap with the other atoms (an isolated node for example), knowing $\boldsymbol{x}[n]$ gives an excellent approximation for the value of w_n. The following theorem uses the sparsity level of $g(\boldsymbol{L})\mathcal{T}_n^G g$ to analyze the overlap between the atom $\mathcal{T}_n^G g$ and the other atoms.

Theorem 12. *For a kernel g, the overlap between the atom localized on the center vertex n and the other atoms satisfies*

$$O_p[n] = \frac{\left(\sum_i \left|\langle \mathcal{T}_i^G g, \mathcal{T}_n^G g\rangle\right|^p\right)^{\frac{1}{p}}}{\left(\sum_i \left|\langle \mathcal{T}_i^G g, \mathcal{T}_n^G g\rangle\right|^2\right)^{\frac{1}{2}}} = \frac{\left\|g(\boldsymbol{L})\mathcal{T}_n^G g\right\|_p}{\left\|g(\boldsymbol{L})\mathcal{T}_n^G g\right\|_2} = \frac{\left\|\mathcal{T}_n^G g\right\|_p^2}{\left\|\mathcal{T}_n^G g\right\|_2^2}$$

Proof. This result follows directly from the application of (4.22) in Lemma 4. □

4.5 Illustrative application: non-uniform sampling

Example 20 (Non-uniform sampling for graph inpainting)**.** *In order to motivate Theorem 12 from a practical signal processing point of view, we use it to optimize the sampling of a signal over a graph. To assess the quality of the sampling, we solve a small inpainting problem where only a part of a signal is measured and the goal is to reconstruct the entire signal. Assuming that the signal varies smoothly in the vertex domain, we can formulate the inverse problem as:*

$$\underset{x}{\operatorname{argmin}}\, x^T \boldsymbol{L} x \quad s.\,t. \quad y = Mx, \tag{4.28}$$

where y is the observed signal, M the inpainting masking operator and $x^T \boldsymbol{L} x$ the graph Tikhonov regularizer ($\boldsymbol{L}$ being the Laplacian). In order to generate the original signal, we filter Gaussian noise on the graph with a low pass kernel h. The frequency content of the resulting signal will be close to the shape of the filter h. For this example, we use the low pass kernel $h(x) = \frac{1}{1+\frac{100}{\lambda_{\max}} x}$ to generate the smooth signal.

For a given number of measurements, the traditional idea is to randomly sample the graph. Under that strategy, the measurements are distributed across the network. Alternatively, we can use our local uncertainty principles to create an adapted mask. The intuitive idea that nodes with less uncertainty (higher local sparsity values) should be sampled with higher probability because their value can be inferred less easily from other nodes. Another way to picture this fact is the following. Imagine that we want to infer a quantity over a random sensor network. In the more densely populated parts of the network, the measurements are more correlated and redundant. As result, a lower sampling rate is necessary. On the contrary, in the parts where there are fewer sensors, the information has less redundancy and a higher sampling rate is necessary. The heat kernel $g(x) = e^{-\tau x}$ is a convenient choice to probe the local uncertainty of a graph, because $g^2(x) = e^{-2\tau x}$ is also a heat kernel, resulting in a sparsity level depending only on $\left\| \mathcal{T}_n^G g^2 \right\|_2$. Indeed we have $\left\| \mathcal{T}_n^G g^2 \right\|_1 = 1$. The local uncertainty bound of Theorem 12 becomes:

$$O_1[n] = \frac{\left\| \mathcal{T}_n^G g^2 \right\|_1}{\left\| \mathcal{T}_n^G g^2 \right\|_2} = \frac{1}{\left\| \mathcal{T}_n^G g^2 \right\|_2}.$$

Based on this measure, we design a second random sampled mask with a probability proportional to $\left\| \mathcal{T}_i^G g^2 \right\|_2$; that is, the higher the overlap level at vertex n, the smaller the probability that vertex n is chosen as a sampling point, and vice-versa. For each sampling ratio, we performed 100 *experiments and averaged the results. For each experiment, we also randomly generated new graphs. The experiment was carried out using open-source code: the UNLocBoX [87] and the GSPBox [86]. Figure 4.13 presents the result of this experiment for a sensor graph and a community graph. In the sensor graph, we observe that our local measure of uncertainty varies smoothly on the graph and is higher in the more dense part. Thus, the likelihood of*

sampling poorly connected vertices is higher than the likelihood of sampling well connected vertices. In the community graph, we observe that the uncertainty is highly related to the size of the community. The larger the community, the larger the uncertainty (or, equivalently, the smaller the local sparsity value). In both cases, the adapted, non-uniform random sampling performs better than random uniform sampling.

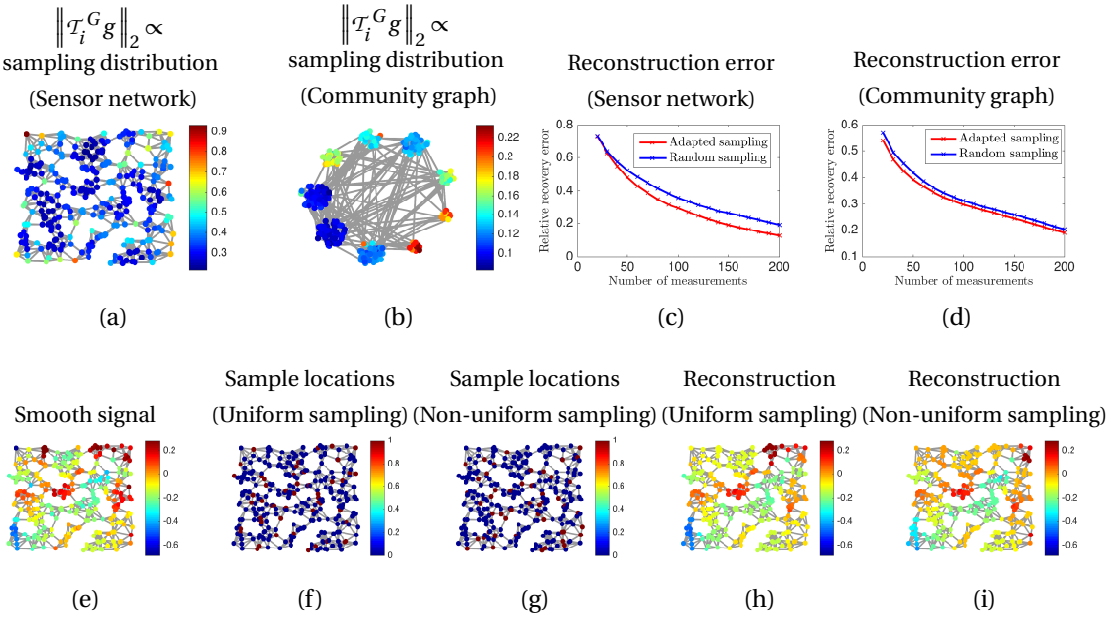


Figure 4.13 – Comparison of random uniform sampling and random non-uniform sampling according to a distribution based on the local sparsity values. Top row: (a)-(b) The random non-uniform sampling distribution is proportional to $\left\|\mathcal{T}_i^G g\right\|_2$ (for different values of i), which is shown here for sensor and community graphs with 300 vertices. (c)-(d) the errors resulting from using the different sampling methods on each graph, with the reconstruction in (4.28). Bottom row: an example of a single inpainting experiment. (e) the smooth signal, (f)-(g) the locations selected randomly according to the uniform and non-uniform sampling distributions, (h)-(i) the reconstructions resulting from the two different sets of samples.

Other works are also starting to use uncertainty principles to develop sampling theory for signals on graphs. In [96], the cumulative coherence is used to optimize the sampling distribution. This can be seen as sampling proportionally to $\left\|\mathcal{T}_i^G g\right\|_2^2$, where g is a specific rectangular kernel, in order to minimize the cumulative coherence of band-limited signals. In [130], Tsitsvero et al. make a link between uncertainty and sampling to obtain a non-probabilistic sampling method. While non-uniform random sampling is only an illustrative example in this paper, we are currently working on a separate contribution that uses our uncertainty theory to optimize sampling.

Part II

Machine learning contributions using graphs

5 Stationary signal processing on graphs

5.1 Introduction

Graphs have been used in Machine learning problems for decades. The general idea is to propagate information between similar pieces of data, an operation done in two steps: 1) the construction of a graph by connecting samples with similar features, and 2), the propagation the labels through the edges.[1] More recently, given $\boldsymbol{x}$ a vector of labels, graphs have been used extensively as regularizers in optimization problems of the form

$$\underset{\boldsymbol{x}}{\operatorname{argmin}}\, f(\boldsymbol{x}) + \boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x}, \tag{5.1}$$

where $f(\boldsymbol{x})$ is usually a data fidellity term such as a ℓ^2 loss [122, 7, 110, 55]. The general idea of the regularizer $\boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x}$ is that it encodes the sum of all variations of the signal $\boldsymbol{x}$, i.e.,

$$\boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} = \left\| \nabla_{\mathcal{G}} \boldsymbol{x} \right\|_2^2 = \frac{1}{2} \sum_{i=1}^{N} \sum_{n=1}^{N} \boldsymbol{W}[i,n] \left(\boldsymbol{x}[n] - \boldsymbol{x}[i] \right)^2. \tag{5.2}$$

The solution of Problem 5.1 is in general more accurate than a k-nearest neighbor average because it takes into account of the non labeled samples to help in the learning process. In fact the graph is often considered as an approximation of a hidden structure from where the data is sampled (i.e., the Manifold). An example is shown in Fig 5.1.

In practice, the optimization scheme of Problem 5.1 has proved to be efficient, providing the following assumption is satisfied: *Samples with similar features have similar labels.* Unfortunately, this assumption limits the class of signals that can be recovered. For example, it is impossible to recover oscillating signals. Figure 5.2 illustrates how traditional techniques fails to recover the signal of an oscillating signal.

In the two previous chapters, we have observed that the behavior of the localization operator adapts to the graph structure. In this chapter, we use this key feature to generalize graph

[1] This technique is usually called *label propagation.*

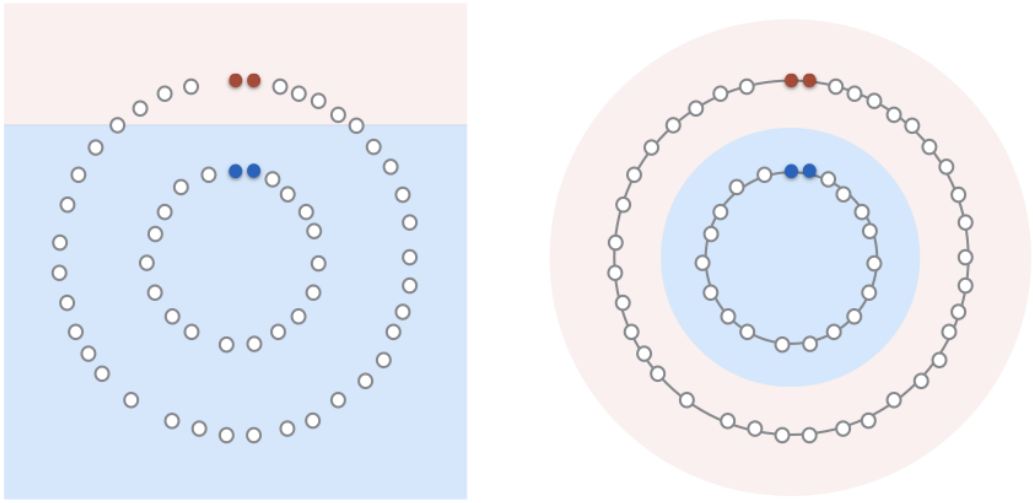


Figure 5.1 – Difference between a kNN classifier (left) and a graph classifier (right). The graph is created by connecting the $k = 2$ nearest neighbors and captures the structure of the data inherent of the non-labeled samples. As the label propagates through the graph edges, the obtain classifier adapts to the structure of the data.

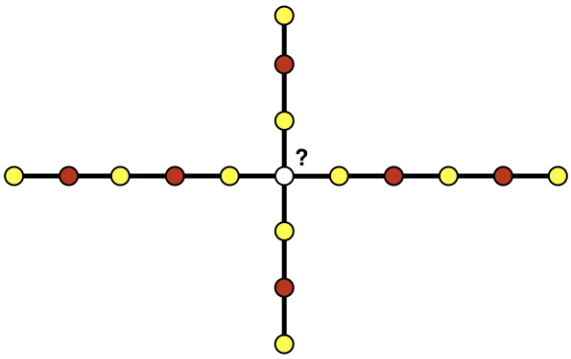


Figure 5.2 – Traditional techniques fails to recover oscillating signals. Here, the central node should be flag as red. However label propagations techniques assign the yellow class.

regularization for a much broader class of signal that includes oscillations . This is done using a probabilistic hypothesis called *stationarity*.[2]

[2]This chapter is availlable with small modifications in [82].

5.1.1 What is stationarity?

Stationarity is a traditional hypothesis in signal processing used to represent a special type of statistical relationship between samples of a temporal signal. The most commonly used is wide-sense stationarity, which assumes that the first two statistical moments are invariant under translation, or equivalently that the correlation between two samples depends only on their time difference. Stationarity is a corner stone of many signal analysis methods. The expected frequency content of stationary signals, called Power Spectral Density (PSD), provides an essential source of information used to build signal models, generate realistic surrogate data or perform predictions. In Figure 5.3, we present an example of a stationary process (blue curve) and two predictions (red and green curves). As the blue signal is a realization of a stationary process, the red curve is more probable than the green one because it respects the frequency content of the observed signal.

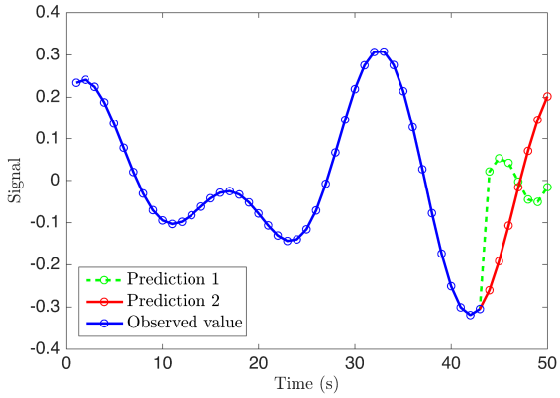


Figure 5.3 – Signal prediction. The red curve is more likely to occur than the green curve because it respects the frequency statistics of the blue curve.

5.1.2 Generalizing stationarity to graph signals

Classical stationarity is a statement of statistical regularity under arbitrary translations and is based on a regular structure (often “time”). However many signals do not live on such a regular structure. For instance, imagine that instead of having one sensor returning a temporal signal, we have multiple sensors living in a two-dimensional space, each of which delivers only one value. In this case (see Figure 5.4 left), the signal support is no longer regular. Since there exists an underlying continuum in this example (2D space), one could assume the existence of a 2D stationary field and use Kriging [136] to interpolate observations to arbitrary locations, thus generalizing stationarity for a regular domain but irregularly spaced samples.

On the contrary, in this thesis we generalize stationarity for an irregular domain that is represented by a graph, *without resorting to any underlying regular continuum.* Our approach is to use a weak notion of translation invariance, the localization operator, that captures the

structure (if any) of the data. Whereas classical stationarity means that correlations are computed by translating the auto-correlation function, here correlations are given by localizing a common graph kernel, which is a generalized notion of translation as detailed in Section 3.1.

Figure 5.4 (left) presents an example of random multivariate variable living in a 2-dimensional space. Seen as scattered samples of an underlying 2D stochastic function, one would (rightly) conclude it is not stationary. However, under closer inspection, the observed values look stationary *within* the spiral-like structure depicted by the graph in Figure 5.4 (right). The traditional Kriging interpolation technique would ignore this underlying structure and conclude that there are always rapid two dimensional variations in the underlying continuum space. This problem does not occur in the graph case, where the statistical relationships inside the data follow the graph edges resulting in this example in signals oscillating smoothly over the graph. A typical example of a stationary signal on a graph would be the result of a survey

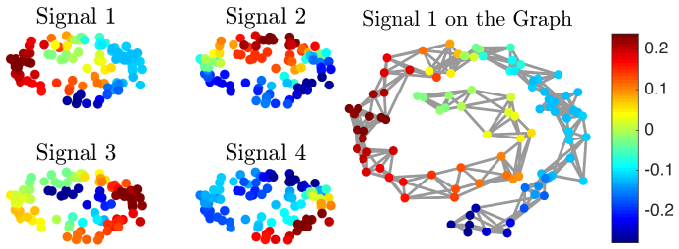


Figure 5.4 – Example of stationary graph signals. The graph connections express relationships between the different elements of one signal. In this case, the signal varies smoothly along the snail shape of the graph.

performed by the users of a social network. If there is a relationship between a user's answer and those of his neighbours, this relationship is expected to be constant among all users. Using stationarity on the graph, we could predict the most probable answer for users that never took the survey.

Outline We use spectral graph theory to extend the notion of stationarity to a broader class of signals. Leveraging the graph localization operator, we establish the theoretical basis of this extension in Section 5.4. We show that the resulting notion of stationarity is equivalent to the proposition of Girault [44, Definition 16], although the latter is not defined in terms of a localisation operator. Localisation is a very desirable feature, since it naturally expresses the scale at which samples are strongly correlated.

Since our framework depends on the power spectral density (PSD), we generalize the Welch method [133, 5] in Section 5.5 and obtain a scalable and robust way to estimate the PSD. It improves largely the covariance estimation when the number of signals is limited.

Based on the generalization of Wiener filters, we propose a new regularization term for graph signal optimization instead of the traditional Dirichlet prior, that depends on the noise level

and on the PSD of the signal. The new optimization scheme presented in Section 5.6 has three main advantages: 1) it permits to deal with an arbitrary regularization parameter, 2) it adapts to the data optimally as we prove that the optimization model is a Maximum A Posteriori (MAP) estimator, and 3) it is more scalable and robust than a traditional Gaussian estimator.

Finally, in Section 5.7, we show experimentally that common datasets such as USPS follow our stationarity assumption. In section 5.8, we exploit this fact to perform missing data imputation and we show how stationarity improves over classical graph models and Gaussian MAP estimator.

5.2 Related work

Graphs have been used for regularization in data applications for more than a decade [114, 122, 139, 92] and two of the most used models will be presented in Section D.1. The idea of graph filtering was hinted at by the machine learning community [122] but developed for the spectral graph wavelets proposed by Hammond et al. [52] and extended by Shuman et al. in [117]. While in most cases, graph filtering is based on the graph Laplacian, Moura et al. [108] have suggested to use the adjacency matrix instead.

We note that a probabilistic model using Gaussian random fields has been proposed in [42, 138]. In this model, signals are automatically graph stationary with an imposed covariance matrix. Our model differentiates itself from these contributions because it is based on a much less restrictive hypothesis and uses the point of view of stationarity. A detailed explanation is given at the end of Section 5.4.

Finally, stationarity on graphs has been recently proposed in [45, 44] by Girault et al. These contributions use a different translation operator (presented in Appendix B.1), promoting energy preservation over localization. While seemingly different, we show that our approach and Girault's result in the same graph spectral characterisation of stationary signals. Girault et al [46] have also shown that using the Laplacian as a regularizer in a de-noising problem (Tikhonov) is equivalent to applying a Wiener filter adapted to a precise class of graph signals. In [44, pp 100], an expression of graph Wiener filter can be found.

After the publication of this work in [82], some additional work were done on the topic. First some PSD estimation methods were proposed in [69, 19]. Then stationarity has been extended to time evolving signals on graphs in [66, 90].

5.3 Stationarity for temporal signals

Let $\boldsymbol{x}[t]$ be a time indexed stochastic process. We use $\overline{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x}]$ to denote the expected value of $\boldsymbol{x}$. In this section, we work with the periodic discrete case.

Definition 23 (Time Wide-Sense Stationarity)**.** *A signal is Time Wide-Sense Stationary (WSS)*

if its first two statistical moments are invariant under translation, i.e.,

1. $\overline{\boldsymbol{x}}[t] = \mathbb{E}[\boldsymbol{x}[t]] = c \in \mathbb{R}$,
2. $\mathbb{E}\left[(\boldsymbol{x}[t] - \overline{\boldsymbol{x}}[t])(\boldsymbol{x}[s] - \overline{\boldsymbol{x}}[s])^*\right] = \boldsymbol{\eta}_{\boldsymbol{x}}[t-s]$,

where $\boldsymbol{\eta}_{\boldsymbol{x}}$ is called the autocorrelation function of $\boldsymbol{x}$.

Note that using Theorem 2, the autocorrelation can be written in terms of the localization operator:

$$\boldsymbol{\eta}_{\boldsymbol{x}}[t-s] = \mathcal{T}_s^G \gamma_{\boldsymbol{x}}[t]. \tag{5.3}$$

For a WSS signal, the autocorrelation function depends only on one parameter, $t-s$, and is linked to the Power Spectral Density (PSD) through the Wiener-Khintchine Theorem [134]. The latter states that the PSD of the stochastic process $\boldsymbol{x}$ denoted $\boldsymbol{\gamma}_{\boldsymbol{x}}[\ell]$ is the Fourier transform of its auto-correlation :

$$\boldsymbol{\gamma}_{\boldsymbol{x}}[\ell] = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} \eta_{\boldsymbol{x}}[t] e^{-j2\pi \frac{\ell t}{N}}, \tag{5.4}$$

where $j = \sqrt{-1}$. As a consequence, when a signal is convolved with a filter $\check{h}$, its PSD is multiplied by the energy of the convolution kernel: for $\boldsymbol{y} = \check{h} * \boldsymbol{x}$, we have

$$\boldsymbol{\gamma}_{\boldsymbol{y}}[\ell] = |h[\ell]|^2 \gamma_{\boldsymbol{x}}[\ell],$$

where h is the Fourier transform of $\check{h}$. For more information about stationarity, we refer the reader to [78].

When generalizing these concepts to graphs, the underlying structure for stationarity will no longer be time, but graph vertices.

5.4 Stationarity of graph signals

We now generalize stationarity to graph signals. While we define stationarity through the localization operator, Girault [45] uses an isometric translation operator instead. That proposition is briefly described in Section 5.4.2, where we also show the equivallence of both definitions.

5.4.1 Stationarity under the localisation operator

Let $\boldsymbol{x} \in \mathbb{R}^N$ be a stochastic graph signal with a finite number of variables indexed by the vertices of a weighted undirected graph. The expected value of each variable is written $\overline{\boldsymbol{x}}[i] = \mathbb{E}[\boldsymbol{x}[i]]$ and the covariance matrix of the stochastic signal is $\boldsymbol{\Sigma}_{\boldsymbol{x}} = \mathbb{E}\left[(\boldsymbol{x} - \overline{\boldsymbol{x}})(\boldsymbol{x} - \overline{\boldsymbol{x}})^*\right]$). We additionally define $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \overline{\boldsymbol{x}}$. For discrete time WSS processes, the covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ is Toeplitz, or circulant for periodic boundary conditions, reflecting translation invariance. In that case, the

covariance is diagonalized by the Fourier transform. We now generalize this property to take into account the intricate graph structure.

As explained in Section 3.1, the localization operator adapts a kernel to the graph structure. As a result, our idea is to use the localization operator to adapt the correlation between the samples to the graph structure. This results in a localised version of the correlation function, whose properties can then be studied via the associated kernel.

Definition 24. *A stochastic graph signal $\boldsymbol{x}$ defined on the vertices of a graph $\mathcal{G}$ is called Graph Wide-Sense (or second order) Stationary (GWSS), if and only if it satisfies the following properties:*

1. *its first moment is constant over the vertex set: $\overline{\boldsymbol{x}}[i] = \mathbb{E}[\boldsymbol{x}[i]] = c \in \mathbb{R}$ and*
2. *its covariance is invariant with respect to the localization operator:*

$$\boldsymbol{\Sigma}_{\boldsymbol{x}}[i,n] = \mathbb{E}\big[(\boldsymbol{x}[i] - \overline{\boldsymbol{x}}[i])(\boldsymbol{x}[n] - \overline{\boldsymbol{x}}[n])\big] = \mathcal{T}_i^G \gamma_{\boldsymbol{x}}[n].$$

The first part of the above definition is equivalent to the first property of time WSS signals. The requirement for the second moment is a natural generalization where we are imposing an invariance with respect to the localization operator instead of the translation. It is a generalization of Definiton 23 using (5.3). In simple words, the covariance is assumed to be driven by a global kernel (filter) $\gamma_{\boldsymbol{x}}$. The localization operator adapts this kernel to the local structure of the graph and provides the correlation between the vertices. Additionally, Definition 24 implies that the spectral components of $\boldsymbol{x}$ are uncorrelated.

Theorem 13. *If a signal is GWSS, if and only if its covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ is jointly diagonalizable with the Laplacian of $\mathcal{G}$ with*[3] *$\gamma_{\boldsymbol{x}}(\lambda_\ell) = \boldsymbol{u}_\ell^* \boldsymbol{\Sigma}_{\boldsymbol{x}} \boldsymbol{u}_\ell$, i.e., $\boldsymbol{\Sigma}_{\boldsymbol{x}} = \boldsymbol{U} \boldsymbol{\Gamma}_{\boldsymbol{x}} \boldsymbol{U}^*$, where $\boldsymbol{\Gamma}_{\boldsymbol{x}}$ is a diagonal matrix.*

Proof. By Definition 15, the covariance localization operator can be written as:

$$\mathcal{T}_i^G \gamma_{\boldsymbol{x}}[n] = \gamma_{\boldsymbol{x}}(\boldsymbol{L})[i,n] = (\boldsymbol{U} \gamma_{\boldsymbol{x}}(\boldsymbol{\Lambda}) \boldsymbol{U}^*)[i,n] \tag{5.5}$$

where $\gamma_{\boldsymbol{x}}(\boldsymbol{\Lambda})$ is a diagonal matrix satisfying $\gamma_{\boldsymbol{x}}(\boldsymbol{\Lambda})[\ell,\ell] = \gamma_{\boldsymbol{x}}(\lambda_\ell)$. To complete the proof set $\boldsymbol{\Gamma}_{\boldsymbol{x}} = \gamma_{\boldsymbol{x}}(\boldsymbol{\Lambda})$. □

The choice of the filter $\gamma_{\boldsymbol{x}}$ in this result is somewhat arbitrary, but we shall soon see that we are interested in localized kernels. In that case, $\gamma_{\boldsymbol{x}}$ will be typically be the lowest degree polynomial satisfying the constraints and can be constructed using Lagrange interpolation for instance.

[3] If the graph Laplacian has an eigenspace of multiplicity greater than one, this condition implies that all eigenvalues of the covariance matrix associated to this eigenspace are equal, i.e., if $\lambda_{\ell_1} = \lambda_{\ell_2}$, then $\boldsymbol{u}_{\ell_1}^* \boldsymbol{\Sigma}_{\boldsymbol{x}} \ell_1 = \boldsymbol{u}_{\ell_2}^* \boldsymbol{\Sigma}_{\boldsymbol{x}} \ell_2$. On a ring graph, it ensures 1) that the Fourier transform of the PSD to be symmetric with respect of the 0 frequency, and 2) that the autocorrelation $\boldsymbol{\eta}_{\boldsymbol{x}}$ is real and symmetric (See Theorem 2.)

Definition 24 provides a fundamental property of the covariance. The size of the correlation (distance over the graph) depends on the support of localized the kernel $\mathcal{T}_i^G \gamma_{\boldsymbol{x}}$. In [117, Theorem 1 and Corollary 2], it has been proved that the concentration of $\mathcal{T}_i^G \gamma_{\boldsymbol{x}}$ around i depends on the regularity of $\gamma_{\boldsymbol{x}}$ (See Theorem 4). For example, if $\gamma_{\boldsymbol{x}}$ is polynomial of degree K, it is exactly localized in a ball of radius K. Hence we will be mostly interested in such low degree polynomial kernels.

The graph spectral covariance matrix of a stochastic graph signal is given by $\boldsymbol{\Gamma}_{\boldsymbol{x}} = \boldsymbol{U}^* \boldsymbol{\Sigma}_{\boldsymbol{x}} \boldsymbol{U}$. For a GWSS signal this matrix is diagonal and the graph power spectral density (PSD) of $\boldsymbol{x}$ becomes:

$$\gamma_{\boldsymbol{x}}(\lambda_\ell) = \left(\boldsymbol{U}^* \boldsymbol{\Sigma}_{\boldsymbol{x}} \boldsymbol{U}\right)_{\ell,\ell}. \tag{5.6}$$

Table 5.1 presents the differences and the similarities between the classical and the graph case. For a regular cyclic graph (ring), the localization operator is equivalent to the traditional translation and we recover the classical cyclic-stationarity results by setting $\boldsymbol{\eta}_{\boldsymbol{x}} = \mathcal{T}_0^G \gamma_{\boldsymbol{x}}$. Our framework is thus a generalization of stationarity to irregular domains.

Example 21 (Gaussian i.i.d. noise)**.** *Normalized Gaussian i.i.d. noise is GWSS for any graph. Indeed, the first moment is* $\mathbb{E}[\boldsymbol{x}[i]] = 0$. *Moreover, the covariance matrix can be written as* $\boldsymbol{I} = \boldsymbol{\Sigma}_{\boldsymbol{x}} = \boldsymbol{U}\boldsymbol{I}\boldsymbol{U}^*$ *with any orthonormal matrix* $\boldsymbol{U}$ *and thus is diagonalizable with any graph Laplacian. We also observe that the PSD is constant, which implies that similar to the classical case, white noise contains all "graph frequencies".*

When $\gamma_{\boldsymbol{x}}$ is a bijective function, the covariance matrix contains an important part of the graph structure: the Laplacian eigenvectors. On the contrary, if $\gamma_{\boldsymbol{x}}$ is not bijective, some of the graph structure is lost as it is not possible to recover all eigenvectors. This is for instance the case when the covariance matrix is low-rank. As another example, let us consider completely uncorrelated centered samples with variance 1. In this case, the covariance matrix becomes $\boldsymbol{\Sigma}_{\boldsymbol{x}} = \boldsymbol{I}$ and loses all graph information, even if by definition the stochastic signal remains stationary on the graph.

One of the crucial benefits of stationarity is that it is preserved by filtering, while the PSD is simply reshaped by the filter. The same property holds on graphs.

Theorem 14. *When a graph filter* g *is applied to a GWSS signal, the result remains GWSS, the mean becomes* $\overline{g(\boldsymbol{L})\boldsymbol{x}} = \overline{\boldsymbol{x}} g(0)$ *and the PSD satisfies:*

$$\gamma_{g(\boldsymbol{L})\boldsymbol{x}}(\lambda_\ell) = |g(\lambda_\ell)|^2 \cdot \gamma_{\boldsymbol{x}}(\lambda_\ell). \tag{5.7}$$

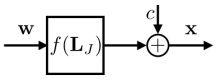
Proof. The output of a filter g can be written as $\boldsymbol{x}' = g(\boldsymbol{L})\tilde{\boldsymbol{x}} + g(\boldsymbol{L})\overline{\boldsymbol{x}}$. If the input signal $\boldsymbol{x}$ is GWSS, we can check easily that the first moment of the filter's output is constant, $\mathbb{E}\left[g(\boldsymbol{L})\boldsymbol{x}\right] =$

$g(\boldsymbol{L})\mathbb{E}[\overline{\boldsymbol{x}}] = g(0)\overline{\boldsymbol{x}}$. The computation of the second moment gives:

$$\begin{aligned}\mathbb{E}\left[g(\boldsymbol{L})\tilde{\boldsymbol{x}}\left(g(\boldsymbol{L})\tilde{\boldsymbol{x}}\right)^*\right] &= g(\boldsymbol{L})\mathbb{E}\left[\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^*\right]g(\boldsymbol{L})^* \\ &= g(\boldsymbol{L})\boldsymbol{\Sigma}_{\boldsymbol{x}}g(\boldsymbol{L})^* \\ &= \boldsymbol{U}g^2(\boldsymbol{\Lambda})\gamma_{\boldsymbol{x}}(\boldsymbol{\Lambda})\boldsymbol{U}^*,\end{aligned}$$

which is equivalent to our claim. □

Theorem 14 provides a simple way to artificially produce stationary signals with a prescribed PSD by simply filtering white noise :



The resulting signal will be stationary with PSD g^2. In the sequel, we assume for simplicity that the signal is centered at 0, i.e., $\overline{\boldsymbol{x}} = 0$. Note that the input white noise could well be non-Gaussian.

	Classical	Graph
Stationary with respect to	Translation	Localization operator
First moment	$\mathbb{E}[\boldsymbol{x}[i]] = \overline{\boldsymbol{x}}[i] = c \in \mathbb{R}$	$\mathbb{E}[\boldsymbol{x}[i]] = \overline{\boldsymbol{x}}[i] = c \in \mathbb{R}$
Second moment (We use $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \overline{\boldsymbol{x}}$)	$\boldsymbol{\Sigma}_{\boldsymbol{x}}[i,n] = \mathbb{E}[\tilde{\boldsymbol{x}}[i])\tilde{\boldsymbol{x}}^*[n]] = \eta_{\boldsymbol{x}}[i-n]$ $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ Toeplitz	$\boldsymbol{\Sigma}_{\boldsymbol{x}}[i,n] = \mathbb{E}[\tilde{\boldsymbol{x}}[i])\tilde{\boldsymbol{x}}^*[n]] = \gamma_{\boldsymbol{x}}(\boldsymbol{L})_{i,n}$ $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ diagonalizable with $\boldsymbol{L}$
Wiener Khintchine	$\gamma_{\boldsymbol{x}}(\lambda_\ell) = \frac{1}{\sqrt{N}}\sum_{i=1}^{N}\eta_{\boldsymbol{x}}[n]e^{-j2\pi\frac{n\ell}{N}}$	$\gamma_{\boldsymbol{x}}(\lambda_\ell) = (\Gamma_{\boldsymbol{x}})_{\ell,\ell} = (U^*\boldsymbol{\Sigma}_{\boldsymbol{x}}U)_{\ell,\ell}$
Result of filtering	$\gamma_{\check{g}*\boldsymbol{x}}(\lambda_\ell) = \lvert g(\lambda_\ell)\rvert^2 \cdot \gamma_{\boldsymbol{x}}(\lambda_\ell)$	$\gamma_{g(\boldsymbol{L})\boldsymbol{x}}[\ell] = \lvert g(\lambda_\ell)\rvert^2 \cdot \gamma_{\boldsymbol{x}}(\lambda_\ell)$

Table 5.1 – Comparison between classical and graph stationarity. In the classical case, we work with a N periodic discrete signal and we use $\check{g}$ to denote the inverse Fourier transform of g.

5.4.2 Comparison with the work of B. Girault

Stationarity for graph signals has been studied recently in [44, 45]. The proposed definition is based on the isometric graph translation operator $\boldsymbol{T}_B$ defined in Appendix B.1. Using this operator, this stationarity definition is a natural extension of the classical case.

Definition 25. [44, Definition 16] *A stochastic signal $\boldsymbol{x}$ on the graph $\mathcal{G}$ is Wide-Sense Stationary (WSS) if and only if*

1. $\mathbb{E}[\boldsymbol{T}_B\boldsymbol{x}] = \mathbb{E}[\boldsymbol{x}]$
2. $\mathbb{E}[\boldsymbol{T}_B\boldsymbol{x}(\boldsymbol{T}_B\boldsymbol{x})^*] = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^*]$

While this definition is based on a fairly different construction, the resulting notion of stationarity is similar. We distinguish two cases: 1) In the case where all eigenvalues are disjoint, they are equivalent. Indeed, [44, Theorem 7] says that if a signal is stationary with Definition 25, then its first moment is constant and the covariance matrix in the spectral domain $\boldsymbol{U}^*\boldsymbol{\Sigma}_{\boldsymbol{x}}\boldsymbol{U}$ has to be diagonal. Using Theorem 13, we therefore recover Definition 24. 2) In the case where the graph has at least an eigenvalue with multiplicity, e.g., a ring graph, our Definition 24 is more restrictive than Definition 25, since we need for every $\lambda_{\ell_1} = \lambda_{\ell_2}$, that $\gamma_{\boldsymbol{x}}(\lambda_{\ell_1}) = \gamma_{\boldsymbol{x}}(\lambda_{\ell_2})$. As a result, there exist signals that are only stationary according to the Definition 25 of Girault, but not according to our Definition 24. As a consequence, for real signals on a ring graph, our definition forces the autocorrelation function to be symmetric, matching exactly the classical stationarity Definition 23, whereas this is not true for Girault's definition.

Let us consider as an example the ring graph with real sine/cosine Fourier basis. Note that a different basis choice leads to the same conclusion. The stochastic signal $\boldsymbol{x}[i] = w\cos(2\pi i\frac{k}{N})$, where $w \sim \mathcal{N}(0,1)$. This signal is made of a single graph Fourier mode, i.e., $\boldsymbol{x} = w\boldsymbol{u}_{2k-1}$. The first moment is given by $\mathbb{E}[\boldsymbol{x}] = \boldsymbol{u}_{2k-1}\mathbb{E}[w] = 0$ and the covariance matrix reads:

$$\boldsymbol{\Sigma}_{\boldsymbol{x}}[n,i] = \mathbb{E}[\boldsymbol{x}[n]\boldsymbol{x}[i]] = \cos\left(2\pi n\frac{k}{N}\right)\cos\left(2\pi i\frac{k}{N}\right).$$

To verify the stationary property of this signal, let us observe this quantity in the spectral domain:

$$\boldsymbol{u}_{\ell_1}^*\Sigma_{\boldsymbol{x}}\boldsymbol{u}_{\ell_2} = \begin{cases} \frac{N}{4} & \text{if } \ell_1 = \ell_2 = 2k-1 \\ 0 & \text{otherwise} \end{cases}.$$

This signal is not stationary according to the classical definition. Indeed it is not invariant with respect to translation. To observe it, just compute $1 = \mathbb{E}[\boldsymbol{x}[N]\boldsymbol{x}[N]] \neq \mathbb{E}\big[\boldsymbol{x}[\frac{N}{4k}]\boldsymbol{x}[\frac{N}{4k}]\big] = 0$. Our definition agrees to this: Applying Theorem 13, even if the covariance matrix in the spectral domain is diagonal, we find that the signal is not stationary. Indeed, we cannot find a kernel satisfying $g(\lambda_\ell) = \boldsymbol{u}_\ell^*\boldsymbol{\Sigma}_{\boldsymbol{x}}\boldsymbol{u}_\ell$ for all ℓ as we have $\lambda_{2k} = \lambda_{2k-1}$ and $0 = u_{2k}^*\Sigma_{\boldsymbol{x}}\boldsymbol{u}_{2k} \neq \boldsymbol{u}_{2k-1}^*\Sigma_{\boldsymbol{x}}\boldsymbol{u}_{2k-1} = \frac{N}{4}$. However, according to the definition by Girault, this signal is stationary [44, Theorem 7].

Another key difference is that our definition allows us to generalize the notion of PSD to the graph setting in a simpler manner. To extend the notion of PSD using Girault's definition, one would have to deal with a block diagonal structure of the covariance matrix in the spectral domain that changes depending on the choice of eigenvectors at eigenvalue multiplicities.[4]

[4] For a subspace associated with an eigenvalue with multiplicity greater than one, there exist multiple possible sets of eigenvectors.

5.4.3 Gaussian random field interpretation

The framework of stationary signals on graphs can be interpreted using Gaussian Marcov Random Field (GMRF). Let us assume that the signal $\boldsymbol{x}$ is drawn from a distribution

$$\mathbb{P}[\boldsymbol{x}] = \frac{1}{Z_{p(Lg)^{\frac{1}{2}}}} e^{-(\boldsymbol{x}-\overline{\boldsymbol{x}})^* p(\boldsymbol{L})(\boldsymbol{x}-\overline{\boldsymbol{x}})}, \tag{5.8}$$

where $Z_{p(Lg)^{\frac{1}{2}}} = \int_{\mathbb{R}^N} e^{-\boldsymbol{x}^* p(\boldsymbol{L})\boldsymbol{x}} \mathrm{d}\boldsymbol{x}$. If we assume that $p(\boldsymbol{L})$ is invertible, drawing from this distribution will generate a stationary $\boldsymbol{x}$ with covariance matrix given by:

$$\boldsymbol{\Sigma}_{\boldsymbol{x}} = \big(p(\boldsymbol{L})\big)^{-1} = p^{-1}(\boldsymbol{L}).$$

In other words, assuming a GRF probabilistic model with inverse covariance matrix $p(\boldsymbol{L})$ leads to a stationary graph signal with a PSD = p^{-1}. However a stationary graph signal is not necessarily a GRF. Indeed, stationarity assumes statistical properties on the signal that are not necessarily based on the Gaussian distribution.

In Section 3 of [42], Gadde and Ortega have presented a GMRF model for graph signals. But they restrict themselves to the case where $p(\boldsymbol{L}) = \boldsymbol{L} + \delta \boldsymbol{I}$. Following a similar approach, Zhang et al. [138] link the inverse covariance matrix of a GMRF with the Laplacian. Our approach is much broader than these two contributions since we do not make any assumption on the function $p(\boldsymbol{L})$. Finally, we exploit properties of stationary signals, such as the characterization of the PSD, to explicitly solve signal processing problems in Section 5.6.

5.5 Estimation of the signal PSD

As the PSD is central in our method, we need a reliable and scalable way to compute it. Equation (5.6) suggests a direct estimation method using the Fourier transform of the covariance matrix. We could thus estimate the covariance $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ empirically from N_s realizations $\{\boldsymbol{x}_m\}_{m=1\ldots,M}$ of the stochastic graph signal $\boldsymbol{x}$, as

$$\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,n] = \frac{1}{M}\sum_{m=1}^{M} (\boldsymbol{x}_m[i] - \dot{\overline{\boldsymbol{x}}}[i]))(\boldsymbol{x}_m[n] - \dot{\overline{\boldsymbol{x}}}[n])^*,$$

where $\dot{\overline{\boldsymbol{x}}}[i] = \sum_{m=1}^{M} \boldsymbol{x}_m[i]$. Then our estimate of the PSD would read

$$\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell) = (\boldsymbol{U}^* \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}} \boldsymbol{U})[\ell,\ell].$$

In fact, one can prove that this is equivalent to use the following estimator:

$$\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell) = \frac{1}{M}\sum_{m=1}^{M} |\hat{\boldsymbol{x}}_m[\ell]|^2. \tag{5.9}$$

The accuracy of this estimator is determined by the number of samples M and the kurtosis of the random graph signal.

Theorem 15. *Given a stationary signal with second moment (PSD)* $\mathbb{E}\left[|\hat{\boldsymbol{x}}[\ell]|^2\right] = \gamma_{\boldsymbol{x}}$ *and fourth order moments (kurtosis)* $\hat{\boldsymbol{m}}_4[\ell] = \frac{\mathbb{E}\left[|\hat{\boldsymbol{x}}[\ell]|^4\right]}{\mathbb{E}\left[|\hat{\boldsymbol{x}}[\ell]|^2\right]^2} = \frac{\mathbb{E}\left[|\hat{\boldsymbol{x}}[\ell]|^4\right]}{\gamma^2_{\boldsymbol{x}}(\lambda_\ell)}$*, the sample PSD estimator* $\dot{\gamma}(\lambda_\ell)$

(a) is unbiased, $\mathbb{E}\left[\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)\right] = \gamma_{\boldsymbol{x}}(\lambda_\ell)$*, and*

(b) has variance $\mathbf{Var}\left[\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)\right] = \gamma^2_{\boldsymbol{x}}(\lambda_\ell)\,\dfrac{\hat{\boldsymbol{m}}_4[\ell]-1}{K}$.

The proof is deferred in Appendix D.6.

Unfortunately, when the number of nodes is considerable, this method requires the diagonalization of the Laplacian, an operation whose complexity in the general case scales as $O\left(N^3\right)$ for the number of operations and $O\left(N^2\right)$ for memory requirements. Additionally, when the number of available realizations K is small, it is not possible to obtain a good estimate of the covariance matrix. To overcome these issues, inspired by Bartlett [5] and Welch [133], we propose to use a graph generalization of the Short Time Fourier transform [117] to construct a scalable estimation method.

Bartlett's method can be summarized as follows. After removing the mean, the signal is first cut into equally sized segments without overlap. Then, the Fourier transform of each segment is computed. Finally, the PSD is obtained by averaging over segments the squared amplitude of the Fourier coefficients. Welch's method is a generalization that works with overlapping segments.

On the other hand, we can see the PSD estimation of both methods as the averaging over time of the squared coefficients of a Short Time Fourier Transform (STFT). Let $\boldsymbol{x}$ be a realization of a stochastic graph signal and $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \overline{\boldsymbol{x}}$ with $\overline{\boldsymbol{x}}[i] = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}[n] = c$, the classical PSD estimator can thus be written as

$$\ddot{\boldsymbol{\gamma}}_{\boldsymbol{x}}[\ell] = \frac{\sum_{n=1}^{N} \left(\mathrm{STFT}\left\{\tilde{\boldsymbol{x}}\right\}[\ell, n]\right)^2}{N \left\| g(\boldsymbol{\lambda}) \right\|_2^2},$$

where g is the window used for the STFT. This is shown in Figure 5.5. Here the factor N in the denominator comes from the fact that the STFT is N times redundant.

Method Our method is based on this idea, using the windowed graph Fourier transform [117]. Instead of a translated rectangular window in time, we use a kernel g shifted by multiples of a step τ in the spectral domain, i.e.,

$$g_k(\lambda_\ell) = g(\lambda_\ell - k\tau), \qquad k = 0 \ldots (K-1), \quad \tau = \frac{\lambda_{\max}}{K-1}.$$

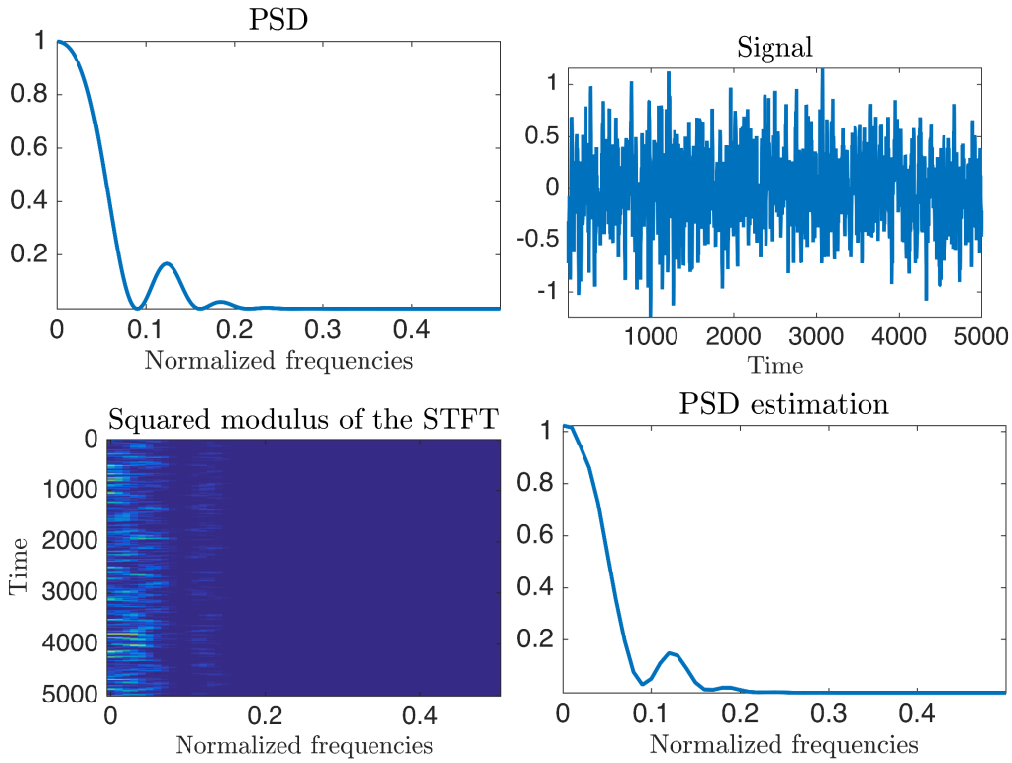


Figure 5.5 – Illustration of the PSD estimation process for a temporal signal. Top right: original PSD. Top left: a stationary signal. Bottom right: The squared modulus of the STFT of the signal. Bottom left: Sum of the STFT squared coefficients over time. We observe that averaging the squared STFT coefficients approximate well the PSD. This method is a version of to the Welch method that is generalizable to graphs.

We then localize each spectral translation at each individual node of the graph. The coefficients of the graph windowed Fourier transform can be seen as a matrix with elements

$$\boldsymbol{C}[i,k] = \langle \boldsymbol{x}, \mathcal{T}_i^G g_k \rangle = \left(g_k(\boldsymbol{L})\boldsymbol{x}\right)[i].$$

Our algorithm consists in averaging the squared coefficients of this transform over the vertex set. Because graphs have an irregular spectrum, we additionally need a normalization factor which is given by the norm of the window g_k: $\left\|g_m(\boldsymbol{\lambda})\right\|_2^2 = \sum_\ell g(\lambda_\ell - k\tau)^2$. Note that this norm will vary for the different k. Our final estimator reads :

$$\ddot{\gamma}_{\boldsymbol{x}}(k\tau) = \frac{\left\|g_k(\boldsymbol{L})\boldsymbol{x}\right\|_2^2}{\left\|g_k(\boldsymbol{\lambda})\right\|_2^2} = \frac{\sum_{i=1}^N \boldsymbol{C}[i,m]^2}{\left\|g_k(\boldsymbol{\lambda})\right\|_2^2}, \tag{5.10}$$

where $\boldsymbol{x}$ is a single realization of the stationary stochastic graph signal. This estimator provides a discrete approximation of the PSD. Interpolation is used to obtain a continuous estimator. This approach avoids the computation of the eigenvectors and the eigenvalues of the

Laplacian.

Our complete estimation procedure is as follows.

1. We design a filterbank by choosing a mother function g (for example a Gaussian $g(\lambda) = e^{-\lambda^2/\sigma^2}$ or an Itersine as presented in (3.25)). A frame is then created by shifting uniformly K times g in the spectral domain: $g_k(\lambda) = g(\lambda - k\tau) = e^{-(\lambda - k\tau)^2/\sigma^2}$.

2. We compute the estimator $\ddot{\gamma}_x(k\tau)$ from the stationary signal $\boldsymbol{x}$. Note that if we have access to M_1 realizations $\{\boldsymbol{x}_m\}_{m=1,\dots,M_1}$ of the stationary signal, we can of course average them to further reduce the variance.

3. We use the following trick to quickly approximate $\left\| g_k \right\|_2^2$. Using M_2 randomly-generated Gaussian normalized zero centered white signals $\boldsymbol{w} \sim \mathcal{D}(\boldsymbol{0}, \boldsymbol{I})$, we use from Lemma 3 the fact that:

$$\mathbb{E}\left[\left\| g_k(\boldsymbol{L})\boldsymbol{w} \right\|_2^2 \right] = \left\| g_k(\boldsymbol{\lambda}) \right\|_2^2.$$

   The techniques is described in details in Section 3.2.2 and the approximation error is characterized by Theorem 5.

4. The last step consists in computing the ratio between the two quantities and interpolating the discrete points $(k\tau, \big(g * \gamma_{\boldsymbol{x}}\big)(k\tau))$.

Variance and bias of the estimator Studying the bias of (5.10) reveals its interest :

$$\frac{\mathbb{E}\left[\left\| g_k(\boldsymbol{L})\tilde{\boldsymbol{x}} \right\|_2^2 \right]}{\left\| g_k \right\|_2^2} = \frac{\sum_{\ell=0}^{N-1} \big(g(\lambda_\ell - k\tau)\big)^2 \gamma_{\boldsymbol{x}}(\lambda_\ell)}{\sum_{\ell=0}^{N-1} \big(g(\lambda_\ell - k\tau)\big)^2}, \tag{5.11}$$

where $\boldsymbol{x}$ is the stationary stochastic graph signal. For a filter g well concentrated at the origin, (5.11) gives a smoothed estimate of $\gamma_{\boldsymbol{x}}(m\tau)$. This smoothing corresponds to the windowing operation in the vertex domain: the less localized the kernel g in the spectral domain, the more pronounced the smoothing effect in (5.11) and the more concentrated the window in the vertex domain. It is very interesting to note that we recover the traditional trade-off between bias and variance in non-parametric spectral estimation. Indeed, if g is very sharply localized on the spectrum, ultimately a Dirac delta, the estimator (5.10) is unbiased. Let us now study the variance. Intuitively, if the signal is correlated only over small regions of the vertex set, we could isolate them with localized windows of a small size and then average those uncorrelated estimates together to reduce the variance. These small size windows on the vertex set correspond to large band-pass kernels g_k and therefore large biases. However, if those correlated regions are large, and this happens when the PSD is localized in low frequencies, we cannot hope to benefit from vertex-domain averaging since the graph is finite. Indeed the corresponding windows g_k on the vertex set are so large that a single window spans the whole graph and there is no averaging effect: the variance increases precisely when we try to suppress the bias.

Theorem 16. *Given a stationary signal $\boldsymbol{x}$ with bounded spectral second moment (the PSD) $\gamma_{\boldsymbol{x}}$ and spectral fourth order moments $\hat{\boldsymbol{m}}_4 = \frac{\mathbb{E}[(\hat{\boldsymbol{x}}-\hat{\boldsymbol{\mu}})^4]}{\mathbb{E}[(\hat{\boldsymbol{x}}-\hat{\boldsymbol{\mu}})^2]^2} = \frac{\mathbb{E}[(\hat{\boldsymbol{x}}-\hat{\boldsymbol{\mu}})^4]}{\gamma_{\boldsymbol{x}}^2}$ (the kurtosis) , the PSD estimator $\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)$ defined in* (5.10)

(a) has bias

$$\left|\mathbb{E}\left[\ddot{\gamma}_{\boldsymbol{x}}(k\tau) - \gamma_{\boldsymbol{x}}(k\tau)\right]\right| \leq \frac{\epsilon}{\left\|g(\boldsymbol{\lambda} - k\tau\mathbf{1})\right\|_2^2} \sum_{\ell=0}^{N-1} g(\lambda_\ell - k\tau)^2 \left|\lambda_\ell - k\tau\right|,$$

where ϵ is the Lipschitz constant of $\gamma_{\boldsymbol{x}}(\lambda_\ell)$, and

(b) has variance

$$\mathbf{Var}\left[\ddot{\gamma}_{\boldsymbol{x}}(k\tau)\right] = \frac{\gamma_{\boldsymbol{x}}(k\tau)^2}{\left\|g(\boldsymbol{\lambda} - k\tau\mathbf{1})\right\|_2^4} \sum_{\ell=0}^{N-1} g(\lambda_\ell - k\tau)^4 \frac{\hat{\boldsymbol{m}}_4[\ell] - 1}{K}.$$

The proof is deferred in Appendix D.7.

Error analysis The difference between the approximation and the exact PSD is caused by three different factors.

1. The inherent bias of the estimator, which is now directly controlled by the spreading of the windows as shown in Theorem 16 a.
2. We estimate the expected value using M_1 realizations of the signal (often $M_1 = 1$). For large graphs $N \gg M_1$ and a few filters $M \ll N$, this error is usually low because the variance of $\left\|g_k(\boldsymbol{L})\tilde{\boldsymbol{x}}\right\|_2^2$ is inversely proportional to the bias. In Theorem 16 b, given that $\hat{\boldsymbol{m}}[\ell]$ is constant, this is encoded by $\frac{\sum_{\ell=0}^{N-1} g(\lambda_\ell - k\tau)^4}{\|g(\boldsymbol{\lambda} - k\tau\mathbf{1})\|_2^4} = \frac{\|g(\boldsymbol{\lambda} - k\tau\mathbf{1})\|_4^4}{\|g(\boldsymbol{\lambda} - k\tau\mathbf{1})\|_2^4}$ that relates to the spreading of the window g. If g is concentrated this factor is close to 1. However if g is close to be constant, it becomes close to $\frac{1}{N}$. We additionally observe that the estimation error improves as $\frac{1}{K}$.
3. We use a fast filtering method based on a polynomial approximation of the filter. For a rough approximation, $\sigma \gg \frac{\lambda_{\max}}{N}$, this error is usually negligible. However, in the other cases, this error may become large.

Experimental assessment of the method Figure 5.6 shows the results of our PSD-estimation algorithm on a 10-nearest neighbors graph of $20'000$ nodes (random geometric graph, weighted with an exponential kernel) and only $M = 1$ realization of the stationary graph signal. We compare the estimation using frames of $K = 10, 30, 100$ Gaussian filters. The parameters σ and τ are adapted to the number of filters such that the shifted windows have an overlap of approximately 2 ($\tau = \sigma^2 = \frac{(K+1)\lambda_{\max}}{K^2}$). For this experiment M_2 is set to 4 and the Chebysheff polynomial order is 30 The estimated curves are smoothed versions of the PSD.

Complexity analysis The approximation scales with the number of edges of the graph $O(E)$, (which is proportional to N in many graphs). Precisely, our PSD estimation method necessi-

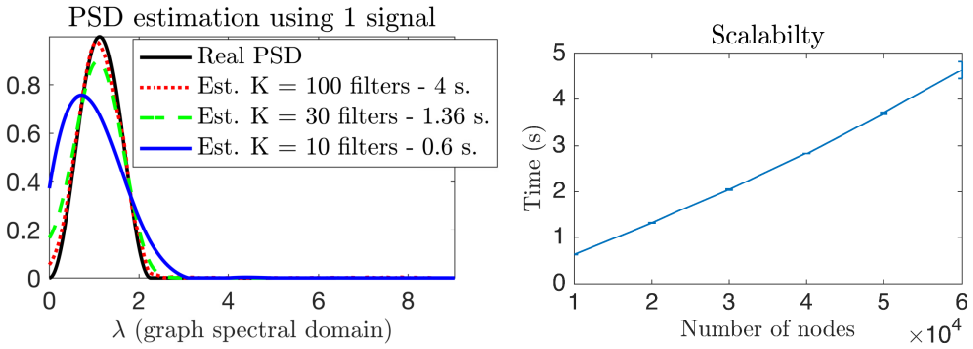


Figure 5.6 – Left: PSD estimation on a graph of 20′000 nodes with $M = 1$ measurements. Our algorithm is able to successively estimate the PSD of a signal. Right: Computation time versus size of the graph (average over 10 runs.). We use $K = 30$ filters. The algorithm scales linearly with the number of edges.

tates $(M + M_2)K$ filtering operations (with M the number of shifts of g). A filtering operation costs approximatively $O_c E$, with O_c the order of the Chebysheff polynomial and E the number of edges as detailled in Section 2.4. The final computational cost of the method is thus $O(O_c(M + M_2)KE)$.

5.6 Graph Wiener filters and optimization framework

Using stationary signals, we can naturally extend the framework of Wiener filters [135] largely used in signal processing for Mean Square Error (MSE) optimal linear prediction. Wiener filters for graphs have already been succinctly proposed in [44, pp 100]. Since the construction of Wiener filters is very similar for non-graph and graph signals, we present only the latter here. The main difference is that the traditional frequencies are replaced by the graph Laplacian eigenvalues[5] λ_ℓ. Figure 5.7 presents the Wiener estimation scheme.

Graph Wiener filtering The Wiener filter can be used to produce a mean-square error optimal estimate of a stationary signal under a linear but noisy observation model. Let us consider the GWSS stochastic signal $\boldsymbol{x}$ with PSD of $s^2(\lambda_\ell)$. For simplicity, we assume $\boldsymbol{\mu} = \overline{\boldsymbol{x}} = 0$. The measurements $\mathbf{y}$ are given by:

$$\boldsymbol{y} = h(\boldsymbol{L})\boldsymbol{x} + \boldsymbol{w}_\sigma, \tag{5.12}$$

where $h(\boldsymbol{L})$ is a graph filter and $\boldsymbol{w}_\sigma$ additive uncorrelated noise of PSD $\sigma^2(\lambda_\ell)$.

[5] The graph eigenvalues are equivalent to classical squared frequencies.

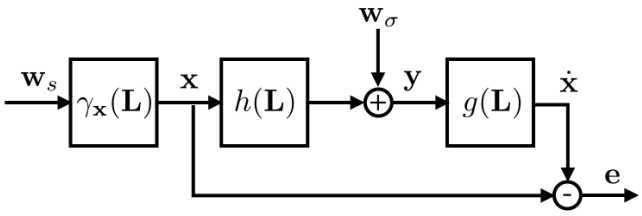


Figure 5.7 – Wiener estimation scheme. $\boldsymbol{w}_\sigma$ is a centered random variable with covariance $\boldsymbol{I}$. $\boldsymbol{w}_s$ generates the stationary stochastic signal $\boldsymbol{x}$ thanks to the filter $s(\boldsymbol{L})$. The random variable $\boldsymbol{y}$ is then generated by filtering $\boldsymbol{x}$ through $h(\boldsymbol{L})$ and adding uncorrelated noise $\boldsymbol{w}_\sigma$ with PSD $n(\lambda)$. The estimator of $\boldsymbol{x}$ given $\boldsymbol{y}$: $\dot{\boldsymbol{x}}|\boldsymbol{y}$ is obtained with the Wiener filter $g(\boldsymbol{L})$. The estimation error is denoted $\boldsymbol{e}$. For clarity, we assume that $\overline{\boldsymbol{x}} = 0$.

To recover $\boldsymbol{x}$, Wiener filters can be extended to the graph case:

$$g(\lambda_\ell) = \frac{h(\lambda_\ell)s^2(\lambda_\ell)}{h^2(\lambda_\ell)s^2(\lambda_\ell) + n(\lambda_\ell)}. \tag{5.13}$$

The expression above can be derived by exactly mimicking the classical case and minimises the expected quadratic error, which can be written as:

$$\boldsymbol{e}[\ell] = \mathbb{E}\left[\left(\hat{\boldsymbol{x}}[\ell] - \hat{\dot{\boldsymbol{x}}}[\ell]\right)^2\right] = \mathbb{E}\left[\hat{\boldsymbol{x}}[\ell] - g(\lambda_\ell)\hat{\boldsymbol{y}}[\ell]\right]^2,$$

where $\dot{\boldsymbol{x}} = g(\boldsymbol{L})\boldsymbol{y}$ is the estimator of $\boldsymbol{x}$ given $\boldsymbol{y}$. Theorem 19 proves the optimality of this filter for the graph case.

Wiener optimization In this contribution, we would like to address a more general problem. Let us suppose that our measurements are generated as:

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}_\sigma, \tag{5.14}$$

where the GWSS stochastic graph signal $\boldsymbol{x}$ has a PSD denoted as $\gamma_{\boldsymbol{x}}(\lambda_\ell) = s^2(\lambda_\ell)$ and the noise $\boldsymbol{w}_\sigma$ a PSD of $\sigma^2(\lambda_\ell)$. We assume $\boldsymbol{x}$ and $\boldsymbol{w}_\sigma$ to be uncorrelated. $\boldsymbol{H}$ is a general linear operator not assumed to be diagonalizable with $\boldsymbol{L}$. As a result, we cannot build a Wiener filter that constructs a direct estimation of the signal $\boldsymbol{x}$. If $\boldsymbol{x}$ varies smoothly on the graph, i.e., is low frequency based, a classic optimization scheme would be the following:

$$\dot{\boldsymbol{x}}|\boldsymbol{y} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\|\boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}\right\|_2^2 + \beta \boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x}. \tag{5.15}$$

This optimization scheme presents two main disadvantages. Firstly, the parameter β must be tuned in order to remove the best amount of noise. Secondly, it does not take into account the data structure characterized by the PSD $s^2(\lambda_\ell)$.

Our solution to overcome these issues is to solve the following optimization problem that we

suggestively call *Wiener optimization*

$$\dot{\boldsymbol{x}}|\boldsymbol{y} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \boldsymbol{H}\boldsymbol{x} - \boldsymbol{y} \right\|_2^2 + \left\| w(\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{\mu}) \right\|_2^2, \tag{5.16}$$

where $w(\lambda_\ell)$ is the Fourier penalization weights. These weights are defined as

$$w(\lambda_\ell) = \left| \frac{\sigma(\lambda_\ell)}{s(\lambda_\ell)} \right| = \frac{1}{\sqrt{\mathrm{SNR}(\lambda_\ell)}}.$$

Notice that compared to (5.15), the parameter β is exchanged with the PSD of the noise. As a result, if the noise parameters are unknown, Wiener optimization does not completely solve the issue of finding the regularization parameter. In the noise-less case, one can alternatively solve the following problem

$$\bar{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| s^{-1}(\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{\mu}) \right\|_2^2, \qquad \text{s. t. } \boldsymbol{H}\boldsymbol{x} = \boldsymbol{y}. \tag{5.17}$$

Problem (5.16) generalizes Problem (5.15) which assumes implicitly a PSD of $\frac{1}{\lambda_\ell}$ and a constant noise level of γ across all frequencies. Note that this framework generalizes two main assumptions done on the data in practice:

1. The signal is smooth on the graph, i.e., the edge derivative has a small ℓ_2-norm. As seen before this is done by setting the PSD as $\frac{1}{\lambda_\ell}$. This case is studied in [46].
2. The signal is band-limited, i.e., it is a linear combination of the k lowest graph Laplacian eigenvectors. This class of signal simply have a null PSD for $\lambda_\ell > \lambda_k$.

Theoretical motivations for the optimization framework The first motivation is intuitive. The weight $w(\lambda_\ell)$ heavily penalizes frequencies associated to low SNR and vice versa.

The second and main motivation is theoretical. If we have a Gaussian Random multivariate signal with i.i.d. Gaussian noise, then Problem (5.16) is a MAP estimator.

Theorem 17. *If $\boldsymbol{x} \sim \mathcal{N}\left(0, s^2(\boldsymbol{L})\right)$ and $\boldsymbol{w}_\sigma \sim \mathcal{N}\left(0, \sigma^2\right)$, i.e., $\boldsymbol{u}$ is GWSS and Gaussian, then problem* (5.16) *is a MAP estimator for $\boldsymbol{x}|\boldsymbol{y}$*

The proof is given in Appendix D.2.

Theorem 18. *If $\boldsymbol{x}$ is GWSS with PSD $s^2(\boldsymbol{L})$ and mean $\boldsymbol{\mu}$, $\boldsymbol{w}_\sigma$ is i.i.d. white noise, then problem* (5.16) *leads to the linear minimum mean square estimator:*

$$\dot{\boldsymbol{x}}|\boldsymbol{y} \quad = \quad \boldsymbol{\Sigma}_{\boldsymbol{x}\boldsymbol{y}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{y} + \left(\boldsymbol{I} - \boldsymbol{\Sigma}_{\boldsymbol{x}\boldsymbol{y}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{H}\right)\boldsymbol{\mu} \tag{5.18}$$

with $\boldsymbol{\Sigma}_{\boldsymbol{x}\boldsymbol{y}} = s^2(\boldsymbol{L})\boldsymbol{H}^*$ and $\boldsymbol{\Sigma}_{\boldsymbol{y}} = \boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^*$

The proof is given in Appendix D.4.

Additionally, when $\boldsymbol{H}$ is jointly diagonalizable with $\boldsymbol{L}$, Problem (5.16) can be solved by a single filtering operation.

Theorem 19. *If the operator $\boldsymbol{H}$ is diagonalizable with $\boldsymbol{L}$, (i.e., $\boldsymbol{H} = h(\boldsymbol{L}) = \boldsymbol{U}a(\boldsymbol{\Lambda})\boldsymbol{U}^*$), then problem* (5.16) *is optimal with respect of the weighting $w(\boldsymbol{L})$ in the sense that its solution minimizes the mean square error:*

$$\mathbb{E}\left[\|\boldsymbol{e}\|_2^2\right] = \mathbb{E}\left[\|\dot{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\right] = \mathbb{E}\left[\sum_{i=1}^{N} (\dot{\boldsymbol{x}}[i] - \boldsymbol{x}[i])^2\right].$$

Additionally, the solution can be computed by the application of the corresponding Wiener filter.

The proof is given in Appendix D.3.

The last motivation is algorithmic and requires the knowledge of proximal splitting methods [25, 58]. Problem (5.16) can be solved by a splitting scheme that minimizes iteratively each of the terms. The minimization of the regularizer, i.e., the proximal operator of $\|w(\boldsymbol{L})\tilde{\boldsymbol{x}}\|_2^2$, becomes a Wiener de-noising operation:

$$\begin{aligned}\operatorname{prox}_{\frac{1}{2}\|w(\boldsymbol{L}))\tilde{\boldsymbol{x}}\|_2^2}(\boldsymbol{y}) &= \boldsymbol{\mu} + \operatorname*{argmin}_{\tilde{\boldsymbol{x}}} \left\|w(\boldsymbol{\Lambda})\boldsymbol{U}^*\tilde{\boldsymbol{x}}\right\|_2^2 + \left\|\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{y}}\right\|_2^2 \\ &= \boldsymbol{\mu} + g(\boldsymbol{L})\tilde{\boldsymbol{y}} = \boldsymbol{\mu} + g(\boldsymbol{L})(\boldsymbol{y} - \boldsymbol{\mu})\end{aligned}$$

with

$$g(\lambda_\ell) = \frac{1}{1 + w^2(\lambda_\ell)} = \frac{s^2(\lambda_\ell)}{s^2(\lambda_\ell) + \sigma^2(\lambda_\ell)}.$$

Advantage of the Wiener optimization framework over a Gaussian MAP estimator Theorem 17 shows that the optimization framework is equivalent to a Gaussian MAP estimator. In practice, when the data is only close to stationary, the true MAP estimator will perform better than Wiener optimization. So one could ask why we bother defining stationarity on graphs. Firstly, assuming stationarity leads us for a more robust estimate of the covariance matrix. This is shown in Figure 5.6, where only one signal is used to estimate the PSD (and thus the covariance matrix). Another example is the USPS experiment presented in the next section. We estimate the PSD by using only 20 digits. The final result is much better than a Gaussian MAP based on the empirical covariance. Secondly, we have a scalable solution for Problem (5.16) (See Algorithm 4 below). On the contrary the classical Gaussian MAP estimator requires the explicit computation of a large part of the covariance matrix and its inverse, which are both not scalable operations.

Solving Problem (5.16) Note that Problem (5.16) can be solved with a simple gradient descent. However, for a large number of nodes N, the matrix $w(\boldsymbol{L})$ requires $O\left(N^3\right)$ operations to be computed and $O\left(N^2\right)$ bits to be stored. This difficulty can be overcome by applying

its corresponding filter operator at each iteration. As already mentioned, the cost of the approximation scale with the number of edges $O(O_c|E|)$ [126].

When $s(\lambda_\ell) \approx 0$ for some λ_ℓ the operator $w(\boldsymbol{L})$ becomes badly conditioned. To overcome this issue, Problem (5.16) can be solved efficiently using a forward-backward splitting scheme [26, 25, 58]. The proximal operator of $\|w(\boldsymbol{L})\tilde{x}\|_2^2$ has been given above and we use the term $\left\|\boldsymbol{H}\boldsymbol{x}-\boldsymbol{y}\right\|_2^2$ as the differentiable function. Algorithm 4 uses an accelerated forward backward scheme [6] to solve Problem (5.16) where β is the step size (we select $\beta = \frac{1}{2\lambda_{\max}(\boldsymbol{H})^2}$), ϵ the stopping tolerance, J the maximum number of iterations and δ is a very small number to avoid a possible division by 0.

Algorithm 4 Fast Wiener optimization to solve (5.16)

INPUT: $\boldsymbol{z}_1 = \boldsymbol{y}$, $\boldsymbol{u}_0 = \boldsymbol{y}$, $t_1 = 1$, $\epsilon > 0$, $\beta \leq \frac{1}{2\lambda_{\max}(\boldsymbol{H})^2}$
SET: $g(\lambda) = \frac{s^2(\lambda)}{s^2(\lambda)+\beta\sigma^2(\lambda)}$ ▷ Wiener filter
for $j = 1, \dots J$ **do**
  $\boldsymbol{v} = \boldsymbol{z}_j - \beta\boldsymbol{H}^*(\boldsymbol{H}\boldsymbol{z}_j - \boldsymbol{y})$ ▷ Gradient step
  $\boldsymbol{u}_{j+1} = g(\boldsymbol{L})\boldsymbol{v}$ ▷ Proximal step
  $t_{j+1} = \frac{1+\sqrt{1+4t_j^2}}{2}$ ▷ FISTA scheme
  $\boldsymbol{z}_{j+1} = \boldsymbol{z}_j + \frac{t_j-1}{t_{j+1}}(\boldsymbol{u}_j - \boldsymbol{u}_{j-1})$ ▷ Update step
  if $\frac{\|\boldsymbol{z}_{j+1}-\boldsymbol{z}_j\|_F^2}{\|\boldsymbol{z}_j\|_F^2+\delta} < \epsilon$ **then** ▷ Stopping criterion
    BREAK
  end if
end for
SOLUTION: $\boldsymbol{z}_J$

5.7 Evidence of graph stationarity: illustration with USPS

Stationarity may not be an obvious hypothesis for a general dataset, since our intuition does not allow us to easily capture the kind of shift invariance that is really implied. In this section we give additional insights on stationarity from a more experimental point of view. To do so, we will show that the well-known USPS dataset is close to stationary on a nearest neighbor graph. We show similar results with a dataset of faces.

Images can be considered as signals on the 2-dimensional euclidean plane and, naturally, when the signal is sampled, a grid graph is used as a discretization of this manifold. The corresponding eigenbasis is the 2 dimensional DCT.[6] Many papers have exploited the fact that natural texture images are stationary 2-dimensional signals [32], i.e., stationary signals on the grid graph. In [104], the authors go one step further and ask the following question: suppose that pixels of images have been permuted, can we recover their relative two-dimensional

[6] This is a natural extension of [125]

Data \Graph	2-dimensional grid	20 nearest neighbors graph
Shifted all digits	0.86	1
All digits	0.66	0.79
Digit 3	0.64	0.83
Digit 7	0.52	0.79
Digit 9	0.52	0.81

Table 5.2 – $s_r(\mathbf{\Gamma}) = \frac{\|\mathrm{diag}(\mathbf{\Gamma})\|_2}{\|\mathbf{\Gamma}\|_F}$: stationarity measures for different graphs and different datasets. The nearest neighbors graph adapts to the data. The individual digits are stationary with the nearest neighbor graph.

location? Amazingly, they answer positively adding that only a few thousand images are enough to approximately recover the relative location of the pixels. The grid graph seems naturally encoded within images.

The observation of [104] motivates the following experiment involving stationarity on graphs. Let us select the USPS data set which contains 9298 digit images of 16×16 pixels. We create 5 classes of data: (a) the circularly shifted digits,[7] (b) the original digits and (c), (d) and (e) the classes of digit 3, 7 and 9. As a pre-processing step, we remove the mean of each pixel, thus forcing the first moment to be 0, and focus on the second moment. For those 5 cases, we compute the covariance matrix $\mathbf{\Sigma}$ and its "Fourier transform",

$$\mathbf{\Gamma} = \boldsymbol{U}^* \mathbf{\Sigma} \boldsymbol{U}, \tag{5.19}$$

for 2 different graphs: (a) the grid and (b) the 20 nearest neighbors graph. In this latter case, each node is a pixel and is associated to a feature vector containing the corresponding pixel value of all images. We use the squared euclidean distance between feature vectors and an exponential kernel to define edge weights (Graph 3 with $k = 20$). We then compute the stationarity level of each class of data with both graphs using the following measure:

$$s_r(\mathbf{\Gamma}) = \left(\frac{\sum_\ell \mathbf{\Gamma}[\ell,\ell]^2}{\sum_{\ell_1} \sum_{\ell_2} \mathbf{\Gamma}[\ell_1,\ell_2][\ell,\ell]^2} \right)^{\frac{1}{2}} = \frac{\|\mathrm{diag}(\mathbf{\Gamma})\|_2}{\|\mathbf{\Gamma}\|_F}. \tag{5.20}$$

The closer $s_r(\mathbf{\Gamma})$ is to 1, the more diagonal the matrix $\mathbf{\Gamma}$ is and the more stationary the signal. Table 5.2 shows the obtained stationarity measures. The less universal the data, the less stationary it is on the grid. Clearly, specificity inside the data requires a finer structure than a grid. This is confirmed by the behavior of the nearest neighbors graph. When only one digit class is selected the nearest neighbors graph still yields very stationary signals.

Let us focus on the digit 3. For this experiment, we build a 20 nearest neighbors graph with only 50 samples. Figure 5.8 shows the eigenvectors of the Laplacian and of the covariance matrix. Because of stationarity, they are very similar. Moreover, they have a 3-like shape. Since

[7] We performed all possible shifts in both directions. Because of this, the covariance matrix becomes Toeplitz

the data is almost stationary, we can use the associated graph and the PSD to generate samples by filtering i.i.d Gaussian noise with the following PSD based kernel: $g(\lambda_\ell) = \sqrt{\mathbf{\Gamma}[\ell,\ell]}$. The resulting digits have a 3-like shape confirming the that the class is stationary on the nearest neighbors graph.

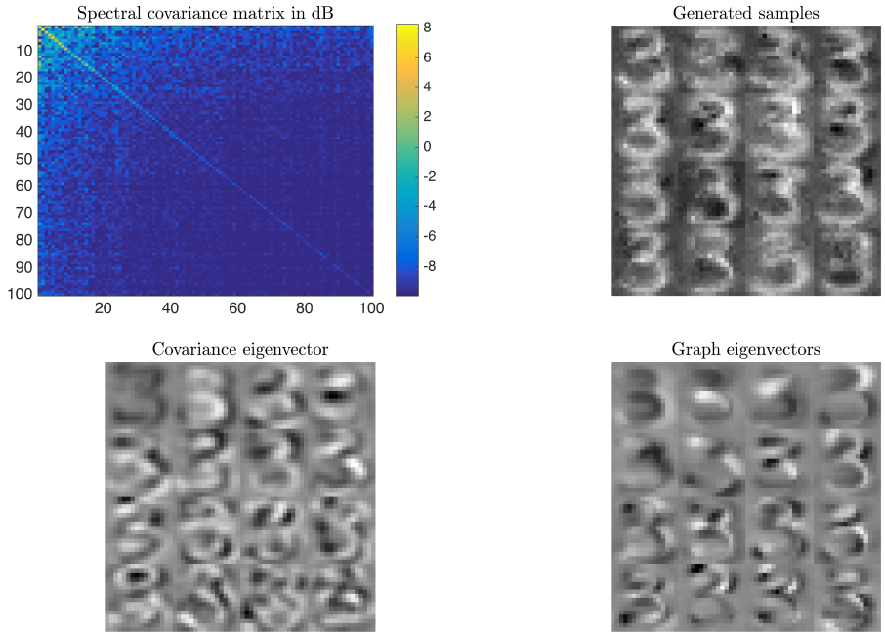


Figure 5.8 – Studying the number 3 of USPS using a 20-neighbors graph. Top left: Spectral covariance matrix of the data (Note the diagonal shape of the matrix). We only display the upper left part for better visibility. Top right: generated samples by filtering Gaussian random noise on the graph. Bottom left: Covariance eigenvectors associated with the 16 highest eigenvalues. Bottom right: Laplacian eigenvectors associated to the 16 smallest non-zero eigenvalues. Because of stationarity, Laplacian eigenvectors are similar to the covariance eigenvectors.

To further illustrate this phenomenon on a different dataset, we use the CMUPIE set of cropped faces. With a nearest neighbor graph we obtained a stationarity level of $s_r = 0.92$. This has already been observed in [53] where the concept of Laplacianfaces is introduced. Finally in [110] the authors succesfully use the graph between features to improve the quality of a low-rank recovery problem. The reason seems to be that the principal components of the data are the lowest eigenvectors of the graph, which is again a stationarity assumption.

To intuitively motivate the effectiveness of nearest neighbors at producing stationary signals, let us define the centering operator $J = I - \mathbf{1}\mathbf{1}^\top / N$. Given M signals x_m, the matrix of average squared distances between the centered features ($\sum_{i=1}^N x_m[i] = 0$) is directly proportional to

the covariance matrix :

$$\bar{\boldsymbol{\Sigma}}_{\boldsymbol{x}} = -\frac{1}{2} JDJ, \tag{5.21}$$

where $D[i,n] = \frac{1}{M}\sum_{m=1}^{M}(x_m[i] - x_m[n])^2$ and $\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,n] = \frac{1}{M}\sum_{m=1}^{M} x_m[i]x_m[n]$. The proof is given in Appendix D.5. The nearest-neighbors graph can be seen as an approximation of the original distance matrix, which pleads for using it as a good proxy destined at leveraging the spectral content of the covariance. Put differently, when using realizations of the signal as features and computing the k-NN graph we are connecting strongly correlated variables via strong edge weights.

5.8 Experiments

All experiments were performed with the GSPBox [86] and the UNLocBoX [87] two open-source softwares. The code to reproduce all figures of the paper can be downloaded at: https://lts2.epfl.ch/rrp/stationarity/. As the stationary signals are random, the reader may obtain slightly different results. However, conclusions shall remain identical. The models used in our comparisons are detailed in the Appendix D.1 for completeness, where we also detail how the tuning of the parameters is done. All experiments are evaluated with respect to the Signal to Noise Ratio (SNR) measure:

$$\mathrm{SNR}(x, \dot{x}) = -10\log\left(\frac{\mathbf{Var}[x - \dot{x}]}{\mathbf{Var}[x]}\right)$$

5.8.1 Synthetic dataset

In order to obtain a first insight into applications using stationarity, we begin with some classical problems solved on a synthetic dataset. Compared to real data, this framework allows us to be sure that the signal is stationary on the graph.

Graph Wiener deconvolution We start with a de-convolution example on a random geometric graph. This can model an array of sensors distributed in space or simply a mesh. The signal is chosen with a low frequency band-limited PSD. To produce the measurements, the signal is convolved with the heat kernel $h(\lambda) = e^{-\tau\lambda}$. Additionally, we add some uncorrelated i.i.d Gaussian noise. The heat kernel is chosen because it simulates a heat diffusion process. Using de-convolution we aim at recovering the original signal before diffusion. For this experiment, we put ourselves in an ideal case and suppose that both the PSD of the input signal and the noise level are known.

Figure 5.9 presents the results. We observe that Wiener filtering is able to de-convolve the measurements. The second plot shows the reconstruction errors for three different methods: Tikhonov presented in problem (D.2), TV in (D.4) and Wiener filtering in (5.13). Wiener filtering performs clearly much better than the other methods because it has a much better

prior assumption.

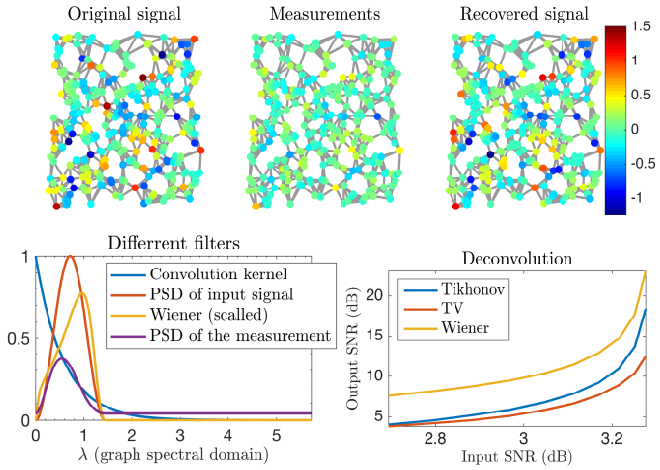


Figure 5.9 – Graph de-convolution on a geometric random graph. The convolution kernel is $e^{-\frac{10x}{\lambda_{\max}}}$. Top: Signal and filters for a noise level of 0.16. Bottom: evolution of the error with respect of the noise.

Graph Wiener in-painting In our second example, we use Wiener optimization to solve an in-painting problem. This time, we suppose that the PSD of the input signal is unknown and we estimate it using 50 signals. Figure 5.10 presents quantitative results for the in-painting. Again, we compare three different optimization methods: Tikhonov (D.1), TV (D.4) and Wiener (5.16). Additionally we compute the classical MAP estimator based on the empirical covariance matrix (See [97] 2.23). Wiener optimization performs clearly much better than the other methods because it has a much better prior assumption. Even with 50 measurements, the MAP estimator performs poorly compared to graphs method. The reason is that the graph contains a lot of the covariance information. Note that the PSD estimated with only one measurement is sufficient to outperform Tikhonov and TV.

5.8.2 Meteorological dataset

We apply our methods to a weather measurements dataset, more precisely to the temperature and the humidity. Since intuitively these two quantities are correlated smoothly across space, it suggests that they are more or less stationary on a nearest neighbors geographical graph.

The French national meteorological service has published in open access a dataset[8] with hourly weather observations collected during the Month of January 2014 in the region of Brest (France). From these data, we wish to ascertain that our method still performs better than the two other models (TV and Tikhonov) on real measurements. The graph is built from the coordinates of the weather stations by connecting all the neighbors in a given radius with a

[8] Access to the raw data is possible directly through our code or through the link https://donneespubliques.meteofrance.fr/donnees_libres/Hackathon/RADOMEH.tar.gz

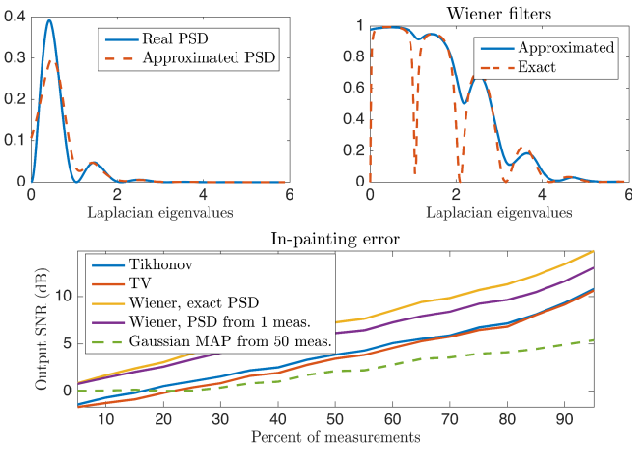


Figure 5.10 – Wiener in-painting on a geometric graph of 400 nodes. Top: true VS approximated PSD and resulting Wiener filters. Bottom: in-painting relative error with respect to number of measurements.

weight function $\boldsymbol{W}[i,n] = e^{-d_{in}^2 \tau}$ where τ is adjusted to obtain a average degree around 3 (τ, however, is not a sensitive parameter). For our experiments, we consider every time step as an independent realization of a GWSS signal. As sole pre-processing, we remove the temperature mean of each station independently. This is equivalent to removing the first moment. Thanks to the 744 time observation, we can estimate the covariance matrix and check whether the signal is stationary on the graph.

Prediction - Temperature The result of the experiment with temperatures is displayed in Figure 5.11. The covariance matrix shows a strong correlation between the different weather stations. Diagonalizing it with the Fourier basis of the graph shows that the meteorological instances are not really stationary within the distance graph as the resulting matrix is not really diagonal. However, even in this case, Wiener optimization still outperforms graph TV and Tikhonov models, showing the robustness of the proposed method. In our experiment, we solve an prediction problem with a mask operator covering 50 per cent of measurements and an initial average SNR of 13.4 dB . We then average the result over 744 experiments (corresponding to the 744 observations) to obtain the curves displayed in Figure 5.11. We observe that Wiener optimization performs always better than the two other methods.

Prediction - Humidity Using the same graph, we have performed another set of experiments on humidity observations. In our experiment, we solve an prediction problem with a mask operator covering 50% of measurements and various amounts of noise. The rest of the testing framework is identical as for the temperature and the conclusions are similar.

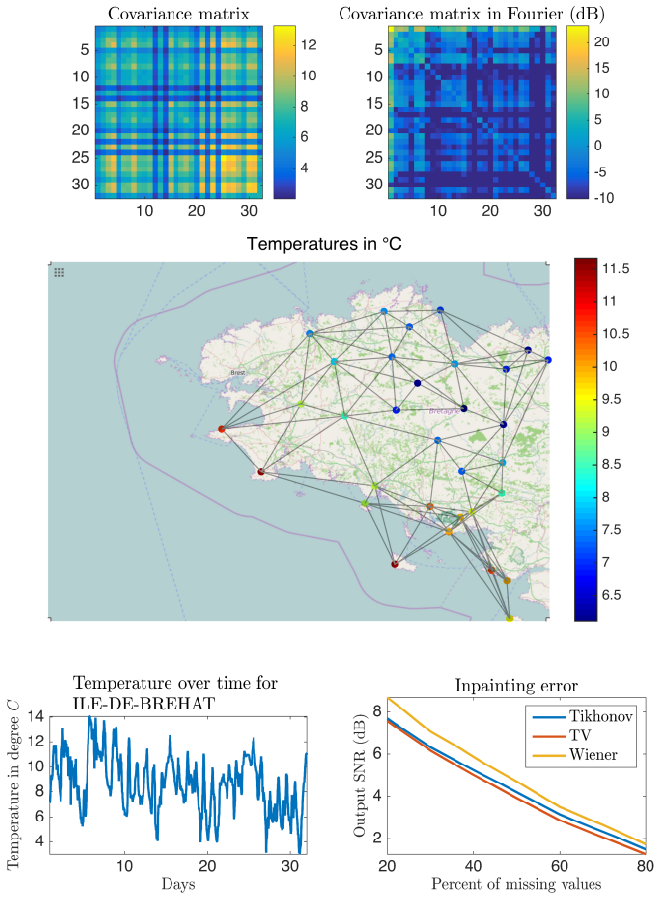


Figure 5.11 – Top: Covariance matrices. Bottom left: A realization of the stochastic graph signal (first measure). Bottom center: the temperature of the Island of Brehat. Bottom right: Recovery errors for different noise levels.

5.8.3 USPS dataset

We perform the same kind of in-painting/de-noising experiment with the USPS dataset. For our experiments, we consider every digit as an independent realization of a GWSS signal. As sole pre-processing, we remove the mean of each pixel separately. This ensures that the first moment is 0. We create the graph[9] and estimate the PSD using only the first 20 digits and we use 500 of the remaining ones to test our algorithm. We use a mask covering 50% of the pixel and various amount of noise. We then average the result over 500 experiments (corresponding to the 500 digits) to obtain the curves displayed in Figure 5.13.[10] For this

[9] The graph is created using patches of pixels of size 5×5. The pixels' patches help because we have only a few digits available. When the size of the data increases, a nearest neighbor graph performs even better.

[10] All parameters have been tuned optimally in a probabilistic way. This is possible since the noise is added artificially. The models presented in Appendix D.1 have only one parameter to be tuned: ϵ which is set to $\epsilon = \sigma\sqrt{\#\boldsymbol{y}}$, where σ is the variance of the noise and $\#y$ the number of elements of the vector y. In order to be fair with the MAP estimator, we construct the graph with the only 20 digits used in the PSD estimation.

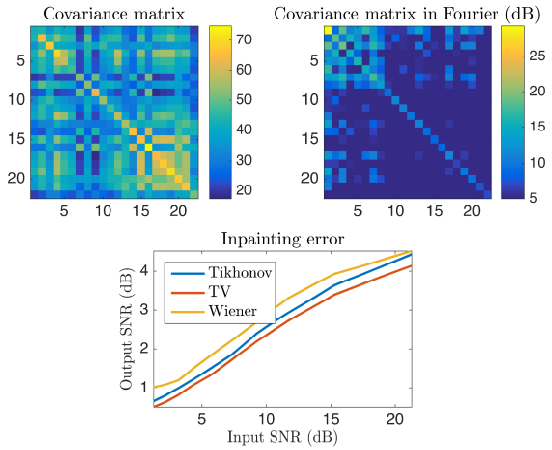


Figure 5.12 – Top: Covariance matrices. Bottom: Recovery errors for different noise levels.

experiment, we also compare to traditional TV de-noising [17] and Tikhonov de-noising. The optimization problems used are similar to (D.1). Additionally we compute the classical MAP estimator based on the empirical covariance matrix for the solution see ([97] 2.23). The results presented in Figure 5.13 show that graph optimization is outperforming classical techniques meaning that the grid is not the optimal graph for the USPS dataset. Wiener once again outperforms the other graph-based models. Moreover, this experiment shows that our PSD estimation is robust when the number of signals is small. In other words, using the graph allows us for a much better covariance estimation than a simple empirical average. When the number of measurements increases, the MAP estimator improves in performance and eventually outperforms Wiener because the data is close to stationary on the graph.

5.8.4 ORL dataset

For this last experiment, we use the ORL faces dataset. We have a good indication that this dataset is close to stationary since CMUPIE (a smaller faces dataset) is also close to stationary. Each image has $112 \times 92 = 10304$ pixels making it complicated to estimate the covariance matrix and to use a Gaussian MAP estimator. Wiener optimization on the other hand does not necessitate an explicit computation of the covariance matrix. Instead, we estimate the PSD using the algorithm presented in Section 5.5. A detailed experiment is performed in Figure 5.14. After adding Gaussian noise to the image, we remove randomly a percentage of the pixels. We consider the obtained image as the measurement and we reconstruct the original image using TV, Tikhonov and Wiener priors. In Figure 5.15, we display the reconstruction results for various noise levels. We create the graph with 300 faces[11] and estimate the PSD with 100 faces. We test the different algorithms on the 100 remaining faces.

[11] We build a nearest neighbor graph based on the pixels values.

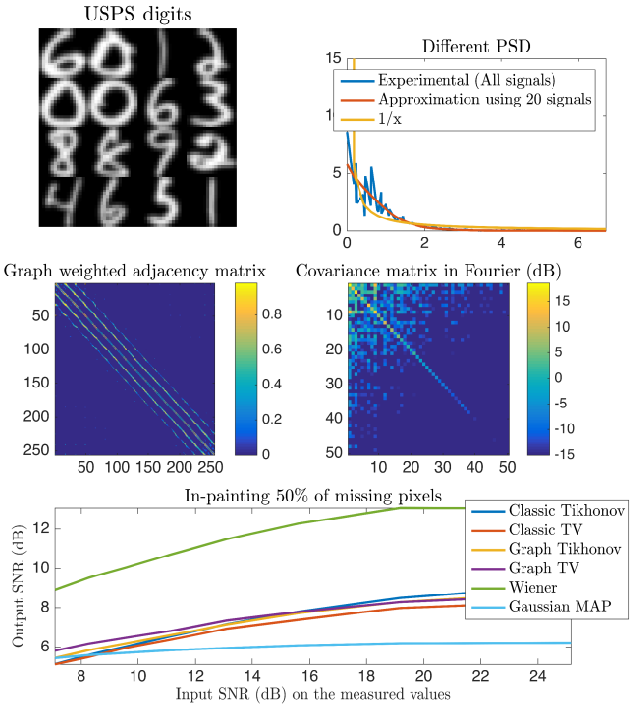


Figure 5.13 – Top left: Some digits of the USPS dataset. Top right: Different PSDs. Compared to $\frac{1}{\lambda}$, the approximation is a smoothed version of the experimental PSD. Middle left: Weights matrix of the 10 nearest neighbors (patch) graph (The diagonal shape indicates the grid base topology of the graph). Middle right: spectral covariance matrix for the first 50 graph frequencies. Since we only use 20 digits for the graph construction, the stationarity level is low. Nevertheless, Wiener optimization outperforms other methods. Bottom: Recovery errors for different noise levels. Methods using the graph perform better. Even if the data is not stationary on the graph, the stationarity assumption helps a lot in the recovery.

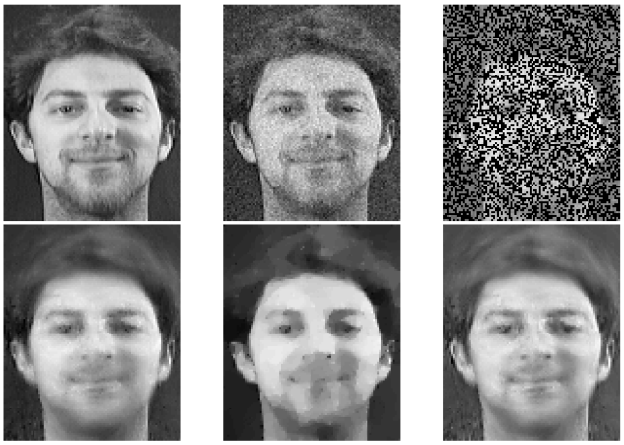


Figure 5.14 – ORL dataset, single in-painting experiment. Top left: Original image. Top center: Noisy image (SNR 12.43 dB). Top right: Measurements 50% of the noisy image. Bottom left: Reconstruction using Tikhonov prior (SNR 12.12 dB). Bottom center: Reconstruction using classic TV prior (SNR 13.53 dB). Bottom right: Reconstruction using Wiener optimization (SNR 14.42 dB).

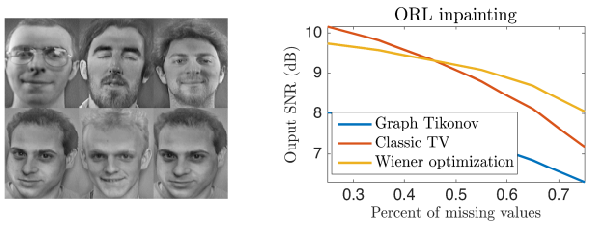


Figure 5.15 – Inpainting experiment on ORL dataset. Left: some images of the dataset. Right: reconstruction error.

6 Manifold regularization via graph total variation

6.1 Introduction

Up to this point, we have been exclusively working on the graph vertices and we have not developed any technique able to handle new samples without having to recompute a new solution. This limits the practical use of many GSP techniques as they become unsuitable for online applications where most of the computations have to be performed in advance and only a small amount of computational power is available for each new coming sample. One solution to overcome this issue is the pre-computation of a global solution that covers not only the graph vertices, but the entire space.

It can be done using Reproducing Kernel Hilbert Spaces (RKHS) as they provide a convenient way to find a function characterizing the entire space [81]. Furthermore, thanks to the representer Theorem, it is possible to transform an infinite dimensional problem into another one that is not only finite but usually also tractable without any tradeoff or relaxation.

6.1.1 Learning with Reproducing Kernel Hilbert Spaces

Given a Mercer kernel $\mathcal{K} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ (self-adjoint positive definite), we know (Moore RKHS-Aronszajn theorem) that there exists a unique Hilbert Space $\mathcal{H}_\mathcal{K}$ of functions on $\mathcal{M} \to \mathbb{R}$ for which $\mathcal{K}$ is a reproducing kernel. Let us denote $\|\cdot\|_\mathcal{K}$ the corresponding norm. Given a set of M labeled point $(\boldsymbol{x}_i, y_i)$, the standard framework estimates the unknown label function by solving:

$$\dot{f} = \underset{f \in \mathcal{H}_\mathcal{K}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \gamma \left\| f \right\|_\mathcal{K}^2, \tag{6.1}$$

where V is a loss function. Traditional choices include the squared loss function $(f(\boldsymbol{x}_i) - y_i)^2$ and Hinge loss function $\max[0, 1 - y_i f(\boldsymbol{x}_i)]$ used by support vector machine (SVM). The

representer theorem [109] states that the solution of (6.1) can be written as

$$\dot{f}(\boldsymbol{x}) = \sum_{i=1}^{M} \boldsymbol{\alpha}[i]\,\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}). \tag{6.2}$$

Therefore, we only need to search for the M-dimensional vector $\boldsymbol{\alpha}$ to find $\dot{f}$.

6.1.2 Semi-supervised learning

As illustrated in Figure 5.1, graphs are a predilection tool to include unlabeled samples in the problem. Also, the unlabeled samples do not carry direct information on the label function f, they inform on the manifold shape and can substantially improve the learning process (See Figure 5.1). This idea has been developed by Belkin and Niogi in [7], where they propose to solve the following problem:

$$\dot{f} = \underset{f \in \mathcal{H}_{\mathcal{K}}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \gamma_{\mathcal{K}} \left\| f \right\|_{\mathcal{K}}^2 + \gamma_{\mathcal{M}} \int_{x \in \mathcal{M}} \left\| \nabla_{\mathcal{M}} f(\boldsymbol{x}) \right\|_2^2 \mathrm{d}\mu(\boldsymbol{x}), \tag{6.3}$$

where μ is the Lebesgue measure associated with the manifold $\mathcal{M}$. In general the manifold is unknown and we cannot compute the integral $\int_{x \in \mathcal{M}} \left\| \nabla_{\mathcal{M}} f(\boldsymbol{x}) \right\|_2^2 \mathrm{d}\mu(\boldsymbol{x})$. Nevertheless, it can be approximated using a graph built from both the labeled and unlabeled points. Given N points $\boldsymbol{x}_i$ where only the M first ones are labeled, the optimization problem can be formulated as

$$\dot{f} = \underset{f \in \mathcal{H}_{\mathcal{K}}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \gamma_{\mathcal{K}} \left\| f \right\|_{\mathcal{K}}^2 + \frac{\gamma_{\mathcal{M}}}{N} \left\| \nabla_{\mathcal{G}} \boldsymbol{f} \right\|_2^2, \tag{6.4}$$

where $\nabla_{\mathcal{G}}$ is a combinatorial graph gradient created from all the N points and $\boldsymbol{f}[i] = f(\boldsymbol{x}[i])$ ($\left\| \nabla_{\mathcal{G}} \boldsymbol{f} \right\|_2^2 = \boldsymbol{f}^* \boldsymbol{L} \boldsymbol{f}$). Problem (6.4) approximates problem (5.15) because the graph Laplacian $\boldsymbol{L} = \nabla_{\mathcal{G}}^* \nabla_{\mathcal{G}}$ converges toward the Laplace-Beltrami operator $\Delta_{\mathcal{M}} = \nabla_{\mathcal{M}}^* \nabla_{\mathcal{M}}$ of the manifold $\mathcal{M}$ [8, 9]. According to the representer theorem [7], the problem (6.4) admits a solution of the form

$$\dot{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \boldsymbol{\alpha}[i]\,\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i).$$

It is important to notice that the vector $\boldsymbol{\alpha}$ has N elements, compared to (6.2), where $\boldsymbol{\alpha}$ has only M elements, i.e., unlabeled samples add degrees of freedom to the solution.

6.1.3 Total variation and Tikhonov regularization

Problem (6.3) is well suited for functions with slow variations. However, it is not adapted to piecewise constant functions with high localized derivatives. To solve this problem, we propose to replace the Tikhonov regularizer $\int_{\boldsymbol{x} \in \mathcal{M}} \left\| \nabla_{\mathcal{M}} f(\boldsymbol{x}) \right\|_2^2 \mathrm{d}\mu(\boldsymbol{x})$ by the Total Variation (TV)

norm on the manifold

$$\|f\|_{\mathcal{M}-TV} = \int_{\boldsymbol{x}\in\mathcal{M}} \|\nabla_{\mathcal{M}} f(\boldsymbol{x})\|_2 \, \mathrm{d}\mu(\boldsymbol{x}). \tag{6.5}$$

While the Tikhonov regularizer does not penalize small derivative changes favoring "functions with slow variations", the total variation norm is less affected by large derivative changes, thus favoring "piecewise flat functions". An illustration of the differences is shown in Figure 6.1 where both regularizers are compared.

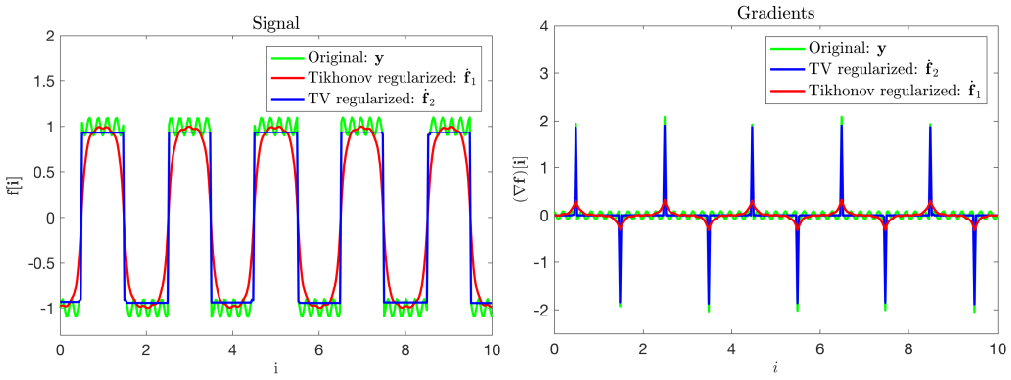


Figure 6.1 – Difference between the Tikhonov and the TV regularizers. Given the function $\boldsymbol{y}$ (in green), we compare the two solutions $\dot{\boldsymbol{f}}_1 = \operatorname{argmin}_{\boldsymbol{f}} \|\boldsymbol{f}-\boldsymbol{y}\|_2^2 + \|\nabla \boldsymbol{f}\|_2^2$ in red and $\dot{\boldsymbol{f}}_2 = \operatorname{argmin}_{\boldsymbol{f}} \|\boldsymbol{f}-\boldsymbol{y}\|_2^2 + \|\nabla \boldsymbol{f}\|_1$ in blue. The TV regularizer favors sparse gradient leading to a piecewise constant function.

Interestingly the same concept applies to graphs. In Figure 6.2, we show the difference between the TV and the Tikhonov regularization for a graph signal. Let us illustrate the difference with a piecewise constant graph signal. Given the measurements $\boldsymbol{y} = \boldsymbol{M}\boldsymbol{x}$, where $\boldsymbol{M}$ is a linear masking operator, we use the two following optimization problems:

$$\dot{\boldsymbol{x}}_1 = \operatorname*{argmin}_{\boldsymbol{z}} \|\nabla_{\mathcal{G}} \boldsymbol{z}\|_2^2 \quad \text{such that} \quad \boldsymbol{M}\boldsymbol{z} = \boldsymbol{y} \tag{6.6}$$

and,

$$\dot{\boldsymbol{x}}_2 = \operatorname*{argmin}_{\boldsymbol{z}} \|\nabla_{\mathcal{G}} \boldsymbol{z}\|_1 \quad \text{such that} \quad \boldsymbol{M}\boldsymbol{z} = \boldsymbol{y}, \tag{6.7}$$

to recover $\boldsymbol{x}$. In the solution of problem (6.6), the missing values are an averaging of their neighbors leading to a smooth transition between the positive and negative parts of the graph signal. On the contrary, in the solution of problem (6.7), the signal is piecewise constant because the total variation norm favors sparse gradient.

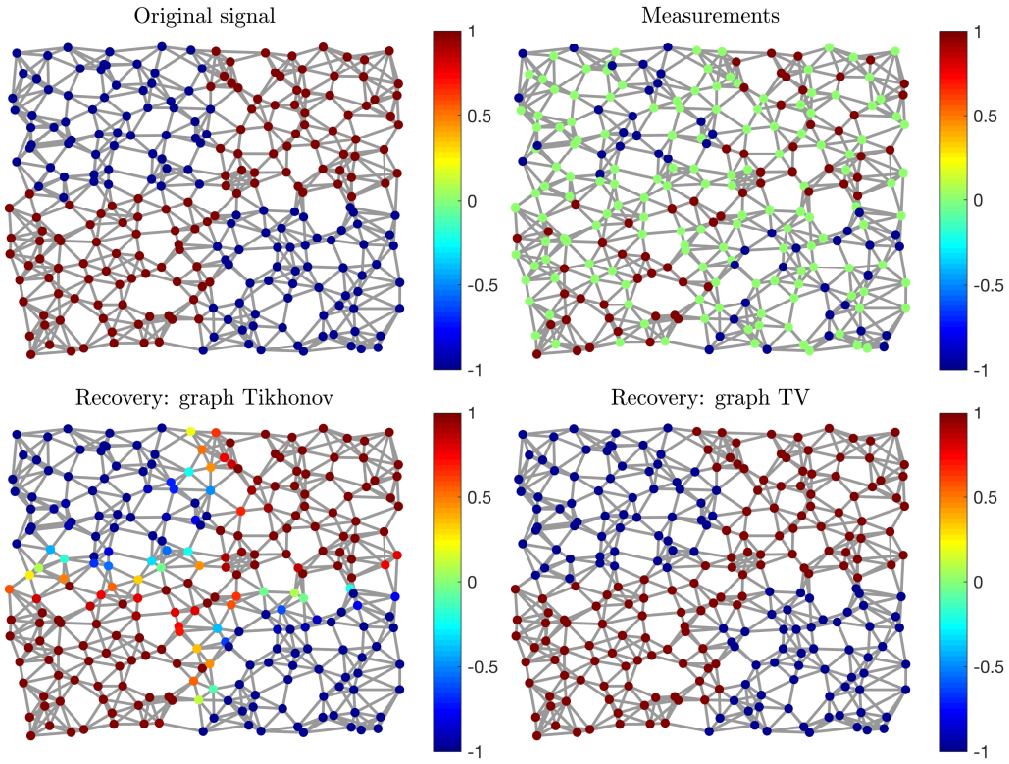


Figure 6.2 – Difference between the Tikhonov and the TV regularizers on graphs. Given the measurement $\boldsymbol{y}$, we compare the two solutions of problems (6.6) and 6.7 (respectively bottom left and right). The TV regularizer favors piecewise constant signals and thus provides a more accurate solution.

6.1.4 Problem formulation

Let us suppose that the function f is piecewise constant on the manifold $\mathcal{M}$. As illustrated in Figure 6.1, it will have a small total variation norm. Using this prior, we formulate the following new optimization problem

$$\hat{f} = \underset{f \in \mathcal{H}_{\mathcal{K}}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \gamma_{\mathcal{K}} \|f\|_{\mathcal{K}}^2 + \gamma_{\mathcal{M}} \|f\|_{TV-\mathcal{M}}, \tag{6.8}$$

where

$$\|f\|_{TV-\mathcal{M}} = \int_{x \in \mathcal{M}} \|\nabla_{\mathcal{M}} f(\boldsymbol{x})\|_2 \, \mathrm{d}\mu(\boldsymbol{x}). \tag{6.9}$$

Problematically, in most of the cases, we cannot compute $\|f\|_{TV-\mathcal{M}}$ because the manifold is not known. Fortunately, similarly to the Tikhonov case, the TV norm on the manifold can be

approximated by the TV norm on a graph. As a result, we propose to solve:

$$\dot{f} = \underset{f \in \mathcal{H}_{\mathcal{K}}}{\operatorname{argmin}} \frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \gamma_{\mathcal{K}} \left\| f \right\|_{\mathcal{K}}^2 + \frac{\gamma_{\mathcal{M}}}{N} \left\| \boldsymbol{f} \right\|_{TV-\mathcal{G}}, \tag{6.10}$$

where

$$\left\| \boldsymbol{f} \right\|_{TV-\mathcal{G}} = \left\| \nabla_{\mathcal{G}} \boldsymbol{f} \right\|_1 = \sum_{i=1,n=1}^{N} \sqrt{\boldsymbol{W}[i,n]} \left| \boldsymbol{f}[n] - \boldsymbol{f}[i] \right|. \tag{6.11}$$

The transformation from problem (6.8) to problem (6.10) is intuitive considering the work of Belkin and Niogi. However, it is motivated only if the two following requirements are satisfied:

1. the representer Theorem applies to (6.10), and
2. the TV norm on graphs converges toward the TV norm on the manifold as the number of samples increases.

These results are proved in Section 6.3.

6.2 Practical use of manifold regularization

Before presenting the theoretical results, let us explain how problem (6.10) is solved in practice. Let us select the squared ℓ_2 norm as a cost function, i.e., $V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) = \left(f(\boldsymbol{x}_i) - y_i\right)^2$. Let $\boldsymbol{K}$ be the positive definite $N \times N$ matrix such that $\boldsymbol{K}[i,n] = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_n)$ and $\boldsymbol{M}$ be the masking operator selecting the M labeled samples. Then, thanks to Theorem 20, the solution of problem (6.10) is given by $\dot{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \dot{\boldsymbol{\alpha}}[i] \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i)$ where

$$\dot{\alpha} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \frac{N}{M} \left\| \boldsymbol{M}\boldsymbol{K}\boldsymbol{\alpha} - \boldsymbol{y} \right\|_2^2 + \gamma_{\mathcal{K}} \boldsymbol{\alpha}^* \boldsymbol{K}\boldsymbol{\alpha} + \gamma_{\mathcal{M}} \left\| \nabla_{\mathcal{G}} \boldsymbol{K}\boldsymbol{\alpha} \right\|_1. \tag{6.12}$$

Problem 6.12 is convex and can be solved using proximal splitting methods [25]. We cannot use a gradient descent since the objective function is not differentiable. Numerically, primal-dual approaches [58] provide the most suitable algorithms to solve (6.12).

Example 22 (One dimensional manifold)**.** *Let us consider the one-dimensional manifold contained in the segment between* 0 *and* 1*. The label function f is defined as the step*

$$f(x) = \begin{cases} 1 & \text{if } x > 0.5 \\ -1 & \text{otherwise.} \end{cases}$$

Given $M = 50$ labeled and $N - M = 450$ unlabeled points, using the same cost function $V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) = \frac{N}{M}\left(y_i - f(\boldsymbol{x}_i)\right)^2$, we compare three different regularizations:

1. *the RKHS norm $\gamma_{\mathcal{K}} \left\| f \right\|_{\mathcal{K}}$,*
2. *the RKHS norm and the graph Tikhonov $\gamma_{\mathcal{K}} \left\| f \right\|_{\mathcal{K}} + \gamma_{\mathcal{M}} \left\| \nabla_{\mathcal{G}} \boldsymbol{f} \right\|_2^2$, and*

3. *the RKHS and the graph TV norms* $\gamma_{\mathcal{K}} \|f\|_{\mathcal{K}} + \gamma_{\mathcal{M}} \|\nabla_{\mathcal{G}} \boldsymbol{f}\|_1$.

The results are displayed in Figure 6.3. First, we observe that, if the unlabeled samples are not included in the learning process, the algorithm does not recover the function where labeled points are missing. Second, the Tikhonov graph regularizer fixes this issue but leads to a function with slow variations. Finally, the TV regularization is clearly the most suitable, as it is the only one to promote a piecewise constant function.

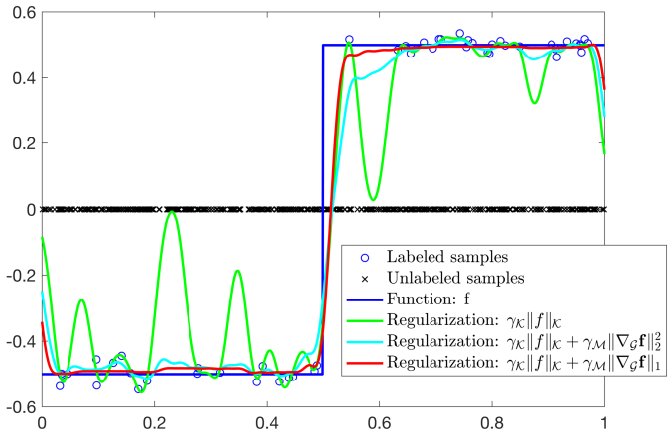


Figure 6.3 – Comparison between 3 different regularization schemes. The TV regularizer is required to recover a piecewise constant function.

6.3 Main theoretical results

Let us now establish the theoretical results that are the core of this chapter. We start with the representer theorem that permits us to compute the solution of problem (6.10).

Theorem 20 (Representer Theorem for problem 6.10)**.** *Let* $\mathcal{M}$ *be a compact manifold and* $\mathcal{K} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ *a positive semi-definite kernel with a corresponding Hilbert space* $\mathcal{H}_{\mathcal{K}}$*. The minimizer of optimization problem* (6.10) *admits an expansion*

$$\dot{f}(\boldsymbol{x}) = \sum_{i=1}^{N} \boldsymbol{\alpha}[i] \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i)$$

in terms of labeled and unlabeled examples.

Proof. The last term of the optimization problem depends on $\boldsymbol{x}_i$ and $f(\boldsymbol{x}_i)$. As a result, optimization problem (6.10) we can simply apply [109, Theorem 1 (Nonparametric Representer Theorem)] with $g(x) = x$ and $c((\boldsymbol{x}_1, y_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_N, y_N, f(\boldsymbol{x}_N))) = \frac{1}{M}\sum_{i=1}^{M} V(\boldsymbol{x}_i, y_i, f(\boldsymbol{x}_i)) + \frac{\gamma_{\mathcal{M}}}{N} \|\nabla_G \boldsymbol{f}\|_1$. □

Let us now focus on the convergence proofs. We consider a k-dimensional compact smooth

manifold $\mathcal{M}$ embedded in $\mathbb{R}^T$. The gradient of a $\mathcal{C}^1(\mathcal{M})$ function f evaluated in $\boldsymbol{x}$ is given by $\nabla_{\mathcal{M}} f(\boldsymbol{x})$. It is a vector pointing in the direction of the fastest ascent of f. Note that the gradient evaluated in $\boldsymbol{x}$ lives in the tangent space of the point $\boldsymbol{x}$: $T_{\mathcal{M}}(\boldsymbol{x}) \in \mathbb{R}^k$. For convenience, we define the linear operator $\nabla_{\mathcal{G},v_i}$ to be the gradient vector at node v_i:

$$\left(\nabla_{\mathcal{G},v_i}\boldsymbol{x}\right)[n] = \frac{\partial x}{\partial e_{in}} = \sqrt{\boldsymbol{W}[i,n]}\left(\boldsymbol{x}[n] - \boldsymbol{x}[i]\right). \tag{6.13}$$

Given a set of N points $\{\boldsymbol{x}_i\}_{i=1\dots N}$, we associate a complete weighted graph where each $\boldsymbol{x}_i$ is associated to the vertex v_i and we define the edges' weights to be:

$$\mathcal{W}(v_i, v_n) = \boldsymbol{W}[i,n] = e^{\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_n\|_2^2}{4t}}, \tag{6.14}$$

where t is a positive constant depending on N. Using this special weight function, Belkin and Niogi have shown that the graph Laplacian converges to the Laplace-Beltrami operator of the manifold $\mathcal{M}$.

Theorem 21 (Theorem 3.1 of [8], Laplacian convergence)**.** *Let data points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ be sampled from a uniform distribution on a manifold $\mathcal{M} \subset \mathbb{R}^T$. Using the weight function* (6.14)*, define a sequence $t_N = N^{-\frac{1}{k+2+\alpha}}$, where $\alpha > 0$ and let $f \in \mathcal{C}^\infty(\mathcal{M})$, then the following holds:*

$$\lim_{N\to\infty} \frac{1}{t_N (4\pi t_N)^{\frac{k}{2}}} (\boldsymbol{L}\boldsymbol{f})[i] = \frac{1}{\mathrm{vol}(\mathcal{M})} (\Delta_{\mathcal{M}} f)(\boldsymbol{x}_i) \tag{6.15}$$

where the limit is taken in probability and $\mathrm{vol}(\mathcal{M})$ *is the volume of the manifold with respect to the canonical measure.*

Generalizing this result for the gradient is particularly complicated as we are comparing objects with different dimensions. At node v_i, the gradient on the graph $\nabla_{\mathcal{G},v_i}\boldsymbol{f}$ is a vector of dimension N and the gradient on the manifold evaluated in $\boldsymbol{x}_i$ is a vector of dimension T.

Our solution to overcome this issue is to focus on the norm of the gradient that is sufficient to prove convergence of the total variation norm (our final goal).

Theorem 22. *Let the data points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ be sampled from a uniform distribution on a manifold $\mathcal{M} \subset \mathbb{R}^T$. Using the weight function* (6.14)*, define a sequence $t_N = \left(\frac{1}{N}\right)^{\frac{1}{k+1+\alpha}}$, where $\alpha > 0$ and let $f \in \mathcal{C}^1(\mathcal{M})$ with finite gradients; then the following holds:*

$$\lim_{N\to\infty} \frac{1}{t_N^{\frac{k+1}{2}} 2^{\frac{k+1}{2}} \pi^{\frac{k-1}{2}}} \frac{1}{N} \left\|\nabla_{\mathcal{G},v_i}\boldsymbol{f}\right\|_1 = \frac{1}{\mathrm{vol}(\mathcal{M})} \left\|\nabla_{\mathcal{M}} f(\boldsymbol{x}_i)\right\|_2 \tag{6.16}$$

where the limit is taken in probability.

The proof is deferred in Section 6.4. While the previous result characterizes only a single point, we can integrate it over the manifold in order to get a global result proving the convergence of the total variation norm.

Theorem 23. *Let the data points $\boldsymbol{x}_1,\dots,\boldsymbol{x}_N$ be sampled from a uniform distribution on a manifold $\mathcal{M}\subset\mathbb{R}^T$. Using the weight function* (6.14)*, define a sequence $t_N=\left(\frac{1}{N}\right)^{\frac{2}{k+1+\alpha}}$, where $\alpha>0$ and let $f\in\mathcal{C}^1(\mathcal{M})$ with finite gradients; then the following holds:*

$$\lim_{N\to\infty}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\frac{1}{N^2}\left\|\boldsymbol{f}\right\|_{TV-\mathcal{G}}=\frac{1}{\mathrm{vol}(\mathcal{M})}\left\|f\right\|_{TV-\mathcal{M}}, \tag{6.17}$$

where the limit is taken in probability.

The proof is deferred in Section 6.5. Note that we have assumed that the function f to be derivable with continuous finite gradient. In consequence our results do not apply to discontinuous functions.

6.4 Proof of Theorem 22

Proof. The proof is done by the application of a succession of lemmas that are proved later in this Section.

$$\begin{aligned}
&\lim_{N\to\infty}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\frac{1}{N}\left\|\nabla_{\mathcal{G},v_i}\boldsymbol{f}\right\|_1\\
=&\lim_{N\to\infty}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\frac{1}{N}\sum_{n=1}^{N}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t_N}}\left|f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\right| &(6.18)\\
=&\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}\in\mathcal{M}}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}}\left|f(\boldsymbol{x})-f(\boldsymbol{x}_i)\right|\mathrm{d}\mu(\boldsymbol{x}) &(6.19)\\
=&\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}\in\mathcal{B}_{\boldsymbol{x}_i}}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}}\left|f(\boldsymbol{x})-f(\boldsymbol{x}_i)\right|\mathrm{d}\mu(\boldsymbol{x}) &(6.20)\\
=&\frac{1}{\mathrm{vol}(\mathcal{M})}\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{x}_i}}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\boldsymbol{0})\right|\mathrm{d}\boldsymbol{x} &(6.21)\\
=&\frac{1}{\mathrm{vol}(\mathcal{M})}\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\boldsymbol{0})\right|\mathrm{d}\boldsymbol{x} &(6.22)\\
=&\frac{1}{\mathrm{vol}(\mathcal{M})}\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k}e^{-\frac{\|\boldsymbol{x}\|_2^2}{2t}}\left|\langle\nabla_{\mathcal{M}}\tilde{f}(\boldsymbol{0}),\boldsymbol{x}\rangle\right|\mathrm{d}\boldsymbol{x} &(6.23)\\
=&\frac{1}{\mathrm{vol}(\mathcal{M})}\left\|\nabla_{\mathbb{R}^k}\tilde{f}(\boldsymbol{0})\right\|_2 &(6.24)\\
=&\frac{1}{\mathrm{vol}(\mathcal{M})}\left\|\nabla_{\mathcal{M}}f(\boldsymbol{x}_i)\right\|_2. &(6.25)
\end{aligned}$$

The first part of the proof (Hoeffding, reduction, exponential map) follows similar steps as the Laplacian convergence used in [8].

- Line (6.18) follows from the definition.

- Line (6.19) is obtained using Hoeffding inequality as detailed in Lemma 5.
- Line (6.20) consists in a reduction to a ball $\mathcal{B}_{\boldsymbol{x}_i}$ around $\boldsymbol{x}_i$ that is done thanks to Lemma 6. It allows us to use the exponential map.
- Line (6.21) is obtained using the exponential map transformation as detailed in Section 6.4.3 and Lemma 7. We use $\exp(\tilde{\mathcal{B}}_{\boldsymbol{x}_i}) = \mathcal{B}_{\boldsymbol{x}_i}$.
- Line (6.22) follows from Lemma 9. We are back in full Euclidean space.
- Line (6.23) is a first order approximation of (6.22) detailed in Lemma 10.
- Line (6.24) is obtained by computing the integral (Lemma 17).
- Line (6.25) follows from the fact that exponential map spans the tangent space of the point $\boldsymbol{x}_i$.

□

6.4.1 From the graph to the manifold

Now we proceed with the different important lemmas used in the proof of Theorem 22. The first step consists in increasing the number of points N towards infinity in order for the sum to become an integral.

Lemma 5. *Define $t_N = \left(\frac{1}{N}\right)^{\frac{1}{k+1+\alpha}}$ with $\alpha > 0$. Given that the samples $\boldsymbol{x}_i$ are uniformly distributed on the manifold $\mathcal{M}$ we have:*

$$\lim_{N\to\infty} \frac{1}{t_N^{\frac{k+1}{2}}} \frac{1}{N} \sum_n e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t_N}} \left|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_i)\right| \quad = \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\mathcal{M}} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{x}) - f(\boldsymbol{x}_i)\right| d\mu(\boldsymbol{x})$$

Proof. We start by computing the expected value of the sum:

$$\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t}} \left|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_i)\right|\right] = \frac{1}{t^{\frac{k+1}{2}}} \int_{\mathcal{M}} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{x}) - f(\boldsymbol{x}_i)\right| \mathrm{d}\mu(\boldsymbol{x}).$$

Then we apply the Hoeffding's inequality to find:

$$\mathbb{P}\left[\frac{1}{t^{\frac{k+1}{2}}}\left|\frac{1}{N}\sum_{n=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t}} \left|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_i)\right| - \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t}} \left|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_i)\right|\right]\right| > \epsilon\right] \leq 2e^{-\frac{1}{2}\epsilon^2 t^{k+1} N}$$

Let us choose $t_N = \left(\frac{1}{N}\right)^{\frac{1}{k+1+\alpha}}$ with $\alpha > 0$. We observe that the right term goes to 0 as $N \to \infty$ which implies

$$\lim_{N\to\infty} \frac{1}{t_N^{\frac{k+1}{2}}} \frac{1}{N} \sum_n e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t_N}} \left|f(\boldsymbol{x}_n) - f(\boldsymbol{x}_i)\right| \quad = \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\mathcal{M}} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{x}) - f(\boldsymbol{x}_i)\right| \mathrm{d}\mu(\boldsymbol{x}).$$

□

6.4.2 Reduction of the integral from the manifold to a ball

The weights are decreasing exponentially with the distance. As a result, the value of the integral is governed by the local behavior of the function, i.e., in an open Ball around the point of interest $\boldsymbol{p}$. We express this through the following lemma that was expressed in a different form in [8, Lemma 4.1].

Lemma 6. *For any point $\boldsymbol{p} \in \mathcal{M}$ and any open set $\mathcal{B}_{\boldsymbol{p}} \subset \mathcal{M}$ with $\boldsymbol{x} \in \mathcal{B}_{\boldsymbol{p}}$ we have the following:*

$$\lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\mathcal{M}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{p}) - f(\boldsymbol{x})\right| d\mu(\boldsymbol{x}) = \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\mathcal{B}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}-\boldsymbol{p}\|_2^2}{2t}} \left|f(\boldsymbol{p}) - f(\boldsymbol{x})\right| d\mu(\boldsymbol{x})$$

Proof. Let $d = \inf_{\boldsymbol{x}\in\mathcal{B}_{\boldsymbol{p}}} \left\|\boldsymbol{x}-\boldsymbol{p}\right\|_2^2$ and let $C = \int_{\boldsymbol{x}\in(\mathcal{M}-\mathcal{B}_{\boldsymbol{p}})} \mathrm{d}\mu(\boldsymbol{x})$ be the volume of the complement of $\mathcal{B}_{\boldsymbol{p}}$. Since the set $\mathcal{B}_{\boldsymbol{p}}$ is open and $\boldsymbol{p} \in \mathcal{B}_{\boldsymbol{p}}$, we have $d > 0$. We can bound the approximation error by:

$$\begin{aligned}\frac{1}{t^{\frac{k+1}{2}}} \left| \int_{\boldsymbol{x}\in\mathcal{M}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{x}) - f(\boldsymbol{p})\right| \mathrm{d}\mu(\boldsymbol{x}) - \int_{\boldsymbol{x}\in\mathcal{B}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{x}\|_2^2}{2t}} \left|f(\boldsymbol{x}) - f(\boldsymbol{p})\right| \mathrm{d}\mu(\boldsymbol{x}) \right| \\ \leq \frac{2}{t^{\frac{k+1}{2}}} M \sup_{\boldsymbol{x}\in\mathcal{M}} \left(|f(\boldsymbol{x})|\right) e^{-\frac{-d^2}{2t}}\end{aligned}$$

Finally, when $t \to 0$, we have:

$$\lim_{t\to 0} \frac{2}{t^{\frac{k+1}{2}}} M \sup_{x\in\mathcal{M}} \left(|f(x)|\right) e^{-\frac{-d^2}{2t}} = 0.$$

□

6.4.3 The exponential map

Since the manifold $\mathcal{M}$ is assumed to be smooth, we know that in every point, there is a local system of coordinates that is locally Euclidean. It is called the exponential map. Using this system of coordinates would allow us to work in $\mathbb{R}^k$, where k is the dimension of the manifold $\mathcal{M}$.

The exponential map at a point $\boldsymbol{p}$ is a map from the tangent space $T_{\mathcal{M}}(\boldsymbol{p}) \in \mathbb{R}^k$ to the manifold such that

- $\exp_{\boldsymbol{p}}(\boldsymbol{0}) = \boldsymbol{p}$,
- $\exp_{\boldsymbol{p}}$ is a local isomorphism, and
- an image of a straight line through the origin is a geodesic in $\mathcal{M}$.

As a consequence, we can express a function $f : \mathcal{M} \to \mathbb{R}$, in geodesic coordinates around $\boldsymbol{p}$:

$$\tilde{f}(\boldsymbol{x}) = f(\exp_{\boldsymbol{p}}(\boldsymbol{x}))$$

Since we have a smooth manifold, the exponential map is locally invertible and we can choose a ball $\tilde{\mathcal{B}}_{\boldsymbol{p}} \in \mathbb{R}^k$ of radius $\epsilon > 0$ (where ϵ has to be chosen appropriately) such that in this ball, the exponential map is a diffeomorphism. Let us define $\mathcal{B}_{\boldsymbol{p}}$ to be:

$$\mathcal{B}_{\boldsymbol{p}} = \exp_{\boldsymbol{p}} \tilde{\mathcal{B}}_{\boldsymbol{p}} \subset \mathcal{M} \tag{6.26}$$

Using this change of variable, we can construct the following Lemma.

Lemma 7. *Let the points $\boldsymbol{p}, \boldsymbol{y}$ belong to a k-dimensional smooth compact manifold $\mathcal{M}$, and, define the exponential map $\boldsymbol{y} = \exp_{\boldsymbol{p}}(\boldsymbol{x})$. Given that $\tilde{f}(\boldsymbol{x}) = f(\exp_{\boldsymbol{p}}(\boldsymbol{x}))$ and $\mathcal{B}_{\boldsymbol{p}} = \exp_{\boldsymbol{p}}(\tilde{\mathcal{B}}_{\boldsymbol{p}})$, then we have*

$$\lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{y}\in\mathcal{B}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{y}\|_2^2}{2t}} \left|\left(f(\boldsymbol{y}) - f(\boldsymbol{p})\right)\right| d\mu(\boldsymbol{y}) = \frac{1}{\mathrm{vol}(\mathcal{M})} \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{x}\|_2^2}{2t}} \left|\left(\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right)\right| d\mu(\boldsymbol{x})$$

The proof of Lemma 7 uses ingredients from [8]. The first one is the following lemma that bounds the difference between the geodesic and the Euclidean distance.

Lemma 8. *[8, Lemma 4.3] For any two points $\boldsymbol{p}, \boldsymbol{y} \in \mathcal{M}$, $\boldsymbol{y} = \exp_{\boldsymbol{p}}(\boldsymbol{x})$, the relation between the Euclidean and the geodesic distance*

$$g_{\boldsymbol{p}}(\boldsymbol{x}) = \|\boldsymbol{x}\|_{\mathbb{R}^k}^2 - \left\|\boldsymbol{y}-\boldsymbol{p}\right\|_{\mathbb{R}^T}^2 \tag{6.27}$$

is $\mathcal{O}\left(\|\boldsymbol{x}\|_{\mathbb{R}^k}^4\right)$. In other words, there exists a finite constant C such that

$$0 \le \|\boldsymbol{x}\|_{\mathbb{R}^k}^2 - \left\|\boldsymbol{y}-\boldsymbol{p}\right\|_{\mathbb{R}^T}^2 = g_{\boldsymbol{p}}(\boldsymbol{x}) \le C\,\|\boldsymbol{x}\|_{\mathbb{R}^k}^4$$

for all $\boldsymbol{p} \in \mathcal{M}$. The constant C depends upon the embedding of the manifold and bounds on the third derivatives of the embedding coordinates.

Proof of Lemma 7. Let us denote $D(f, \boldsymbol{p})$ the left-hand side of the equality and perform the exponential map change of coordinates around $\boldsymbol{p}$:

$$\begin{aligned} D(f,\boldsymbol{p}) &= \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{y}\in\mathcal{B}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{p}-\boldsymbol{y}\|_2^2}{2t}} \left|f(\boldsymbol{y}) - f(\boldsymbol{p})\right| \mathrm{d}\mu(\boldsymbol{y}) \\ &= \frac{1}{\mathrm{vol}(\mathcal{M})} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{p}-\exp_{\boldsymbol{p}}(\boldsymbol{x})\|_2^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| \sqrt{\det\left(\boldsymbol{G}(\boldsymbol{x})\right)} \mathrm{d}\boldsymbol{x} \end{aligned}$$

where $\boldsymbol{G}(\boldsymbol{x})$ is the metric tensor in exponential coordinates. Now it has been shown in [123, 124] that

$$\det\left(\boldsymbol{G}(\boldsymbol{x})\right) = 1 - \frac{1}{6}\boldsymbol{x}^*\boldsymbol{R}\boldsymbol{x} + \mathcal{O}\left(\|\boldsymbol{x}\|^3\right)$$

where $\boldsymbol{R}$ is the Ricci curvature tensor. On a smooth compact manifold $\mathcal{M}$, the elements of the

tensor $\boldsymbol{R}$ are bounded and therefore we have: $\sqrt{\det(\boldsymbol{G}(\boldsymbol{x}))} = 1 + O(\|\boldsymbol{x}\|^2)$

Now note that the function $e^\alpha = 1 + O(\alpha e^\alpha)$ for $\alpha > 0$. Thus we have

$$e^{-\frac{\left\|\boldsymbol{p}-\exp_{\boldsymbol{p}}(\boldsymbol{x})\right\|^2}{2t}} = e^{-\frac{\|\boldsymbol{x}\|^2 - g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}} = e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} e^{\frac{g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}} = e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left(1 + O\left(\frac{1}{2t} g_{\boldsymbol{p}}(\boldsymbol{x}) e^{\frac{g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}}\right)\right)$$

where $g_{\boldsymbol{p}}(\boldsymbol{x})$ is $O\left(\|x\|^4\right)$ from Lemma 8. Finally, our expression becomes

$$\begin{aligned} D(f,\boldsymbol{p}) &= \frac{1}{\mathrm{vol}(\mathcal{M})} \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left(1 + O\left(\frac{1}{2t} g_{\boldsymbol{p}}(\boldsymbol{x}) e^{\frac{g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}}\right)\right) \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| \left(1 + O\left(\|\boldsymbol{x}\|^2\right)\right) \mathrm{d}\boldsymbol{x} \\ &= A(f,\boldsymbol{p}) + B(f,\boldsymbol{p}) + C(f,\boldsymbol{p}) \\ &= A(f,\boldsymbol{p}) \end{aligned}$$

where

$$\begin{aligned} A(f,\boldsymbol{p}) &= \frac{1}{\mathrm{vol}(\mathcal{M})} \lim_{t\to 0} \frac{1}{\mathrm{vol}(\mathcal{M})} \frac{1}{t^{\frac{k+1}{2}}} \int_{\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| \mathrm{d}\boldsymbol{x} \\ B(f,\boldsymbol{p}) &= \frac{1}{\mathrm{vol}(\mathcal{M})} \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| O\left(\|\boldsymbol{x}\|^2\right) \mathrm{d}\boldsymbol{x} \\ C(f,\boldsymbol{p}) &= \frac{1}{\mathrm{vol}(\mathcal{M})} \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} O\left(\frac{1}{2t} g_{\boldsymbol{p}}(\boldsymbol{x}) e^{\frac{g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}}\right) \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| \left(1 + O\left(\|\boldsymbol{x}\|^2\right)\right) \mathrm{d}\boldsymbol{x}. \end{aligned}$$

To conclude the proof, we simply need to show that both $B(f,\boldsymbol{p})$ and $C(f,\boldsymbol{p})$ are equal to 0. Let us start with $B(f,\boldsymbol{p})$.

For every $\boldsymbol{x}$, the function to be integrated is positive and we know that there are constants K_1, K_2 and K_3 such that

$$\begin{aligned} B(f,\boldsymbol{p}) &\leq \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\boldsymbol{0})\right| K_1 \|\boldsymbol{x}\|^2 \mathrm{d}\boldsymbol{x} \\ &\leq \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\tilde{\mathcal{B}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} K_2 \|\boldsymbol{x}\|^2 \mathrm{d}\boldsymbol{x} \\ &\leq \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} K_2 \|\boldsymbol{x}\|^2 \mathrm{d}\boldsymbol{x} \\ &= \lim_{t\to 0} K_2 \frac{1}{t^{\frac{k+1}{2}}} \frac{k}{2} (2t)^{\frac{k+2}{2}} \pi^{\frac{k}{2}} = \lim_{t\to 0} K_3 t^{\frac{1}{2}} = 0. \end{aligned}$$

The key ingredient for the bound is the fact that $\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{x}\|^2}{t}} \|\boldsymbol{x}\|_2^2 \mathrm{d}\boldsymbol{x} = \frac{k}{2} t^{\frac{k+2}{2}} \pi^{\frac{k}{2}}$, which is proved in Lemma 15.

For the term $C(f,\boldsymbol{p})$, we use similar argument. Thanks to Lemma 8, we know that there exist

some constants K_1, K_2 such that

$$\begin{aligned} C(f,\boldsymbol{p}) &= \frac{1}{\mathrm{vol}(\mathcal{M})}\lim_{t\to 0}\frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} O\left(\frac{1}{2t} g_{\boldsymbol{p}}(\boldsymbol{x}) e^{\frac{g_{\boldsymbol{p}}(\boldsymbol{x})}{2t}}\right)\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\mathbf{0})\right|\left(1+O\left(\|\boldsymbol{x}\|^2\right)\right)\mathrm{d}\boldsymbol{x} \\ &\leq \lim_{t\to 0}\frac{1}{t^{\frac{k+3}{2}}}\int_{\boldsymbol{x}\in\tilde{\mathcal{B}}_{\boldsymbol{p}}} e^{-\frac{\|\boldsymbol{x}\|^2}{4t}}\|\boldsymbol{x}\|^4 K_1\mathrm{d}\boldsymbol{x} \\ &\leq \lim_{t\to 0}\frac{1}{t^{\frac{k+3}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|^2}{4t}}\|\boldsymbol{x}\|^4 K_1\mathrm{d}\boldsymbol{x} \\ &\leq K_2\lim_{t\to 0} t^{\frac{1}{2}}, \end{aligned}$$

where we use the fact that $\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{x}\|^2}{t}}\|\boldsymbol{x}\|_2^4\,\mathrm{d}\boldsymbol{x} = \frac{k^2+2k}{4}t^{\frac{k+4}{2}}\pi^{\frac{k}{2}}$ (Lemma 16).

□

6.4.4 Analysis in $\mathbb{R}^k$

In this section, we prove the two final lemmas which are basically the explicit computation of the integral in the Euclidean space.

Lemma 9. *For any open set $\tilde{\mathcal{B}}\subset\mathbb{R}^k$ such that $\mathbf{0}\in\tilde{\mathcal{B}}$, we have the following.*

$$\lim_{t\to 0}\frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\tilde{\mathcal{B}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\mathbf{0})\right| dx = \lim_{t\to 0}\frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\mathbf{0})\right| d\boldsymbol{x}$$

Proof. Let us bound the error and show that it converges towards 0 with $t\to 0$. We define

$$e(t) = \frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\tilde{\mathcal{B}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\mathbf{0})\right|\mathrm{d}\boldsymbol{x} - \frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}}\left|\tilde{f}(\boldsymbol{x})-\tilde{f}(\mathbf{0})\right|\mathrm{d}\boldsymbol{x}$$

Since $\tilde{\mathcal{B}}$ is open and contains $\mathbf{0}$, we can find a ball centered in $\mathbf{0}$ of radius $r>0$ such that

$\mathcal{B}_{\mathbf{0},r} \in \tilde{\mathcal{B}}$.

$$\begin{aligned}
|e(t)| &= \frac{1}{t^{\frac{k+1}{2}}} \left| \int_{\boldsymbol{x}\in\tilde{\mathcal{B}}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\mathbf{0})\right| \mathrm{d}\boldsymbol{x} - \int_{\mathbb{R}^k} e^{-\frac{\|x\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\mathbf{0})\right| \mathrm{d}\boldsymbol{x} \right| \\
&= \frac{1}{t^{\frac{k+1}{2}}} \left| \int_{\boldsymbol{x}\in\mathbb{R}^k\setminus\tilde{\mathcal{B}}} e^{-\frac{\|x\|^2}{2t}} \left|\tilde{f}(\boldsymbol{x}) - \tilde{f}(\mathbf{0})\right| \mathrm{d}\boldsymbol{x} \right| \\
&\leq \frac{1}{t^{\frac{k+1}{2}}} \left| \int_{\mathbb{R}^k\setminus\mathcal{B}_{0,r}} e^{-\frac{\|\boldsymbol{x}\|^2}{2t}} C \mathrm{d}\boldsymbol{x} \right| \\
&= \frac{2^k C}{t^{\frac{k+1}{2}}} \left| \int_{x=r}^{\infty} e^{-\frac{x^2}{2t}} \mathrm{d}x \right|^k \\
&= \frac{2^{k+\frac{1}{2}} C}{t^{\frac{1}{2}}} \left| \int_{x=\frac{r}{\sqrt{2t}}}^{\infty} e^{-x^2} \mathrm{d}x \right|^k \\
&= \frac{2^{k+\frac{1}{2}} C}{t^{\frac{1}{2}}} \left(\frac{\sqrt{\pi}}{2} \mathrm{erfc}\left(\frac{r}{\sqrt{2t}} \right) \right)^k,
\end{aligned}$$

where $C = 2\max_{\boldsymbol{x}} \tilde{f}(\boldsymbol{x})$. Furthermore

$$\lim_{t\to 0} \frac{C_2}{t^{\frac{1}{2}}} \left(\mathrm{erfc}\left(\frac{r}{\sqrt{2t}} \right) \right)^k = 0,$$

which implies that the error is 0 and that the equality holds. This final equality is found using a change of variable and L'Hospital's rule

$$\lim_{t\to 0} \frac{1}{t} \mathrm{erfc}\left(\frac{r}{\sqrt{2t}} \right) = \lim_{t\to\infty} \frac{\mathrm{erfc}\left(\frac{r\sqrt{t}}{\sqrt{2}} \right)}{\frac{1}{t}} = \lim_{t\to\infty} \frac{\frac{2}{\sqrt{\pi}} e^{-\frac{r^2 t}{2}}}{\frac{1}{t^2}} = 0.$$

□

Lemma 10. *Given a bounded function $f \in \mathcal{C}^1(\mathbb{R}^k)$ with bounded derivatives, we have the following equality*

$$\lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|_2^2}{t}} \left| f(\boldsymbol{x}) - f(\mathbf{0}) \right| d\boldsymbol{x} = \lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|_2^2}{t}} \left| \langle \nabla_{\mathcal{M}} f(\mathbf{0}), \boldsymbol{x} \rangle \right| d\boldsymbol{x}$$

Proof. The proof consists in a) showing the following equality

$$\lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{\|\boldsymbol{x}\|_2^2}{t}} \left| \left| f(\boldsymbol{x}) - f(\mathbf{0}) \right| - \left| \langle \nabla_{\mathbb{R}^k} f(\mathbf{0}), \boldsymbol{x} \rangle \right| \right| \mathrm{d}\boldsymbol{x} = 0$$

and b) using Lemma 14 to obtain the desired result.

Let us consider the following expansion of the function f:

$$f(\boldsymbol{x}) = f(\mathbf{0}) + \langle \nabla_{\mathbb{R}} f(\mathbf{0}), \boldsymbol{x} \rangle + O\left(\|\boldsymbol{x}\|^2\right)$$

The gradient approximates the function up to the first order. As a consequence, the error varies with the second order. Moreover since f is $\mathcal{C}^1(\mathbb{R}^k)$ with bounded derivatives, there exists a constant C such that

$$\Big|\big|f(\boldsymbol{x})-f(\mathbf{0})\big|-\big|\langle\nabla_{\mathbb{R}^k}f(\mathbf{0}),\boldsymbol{x}\rangle\big|\Big|\leq\big|\big(f(\boldsymbol{x})-f(\mathbf{0})\big)-\langle\nabla_{\mathbb{R}^k}f(\mathbf{0}),\boldsymbol{x}\rangle\big|\leq C\left\|\boldsymbol{x}\right\|_2^2,\quad\forall\boldsymbol{x}\in\mathbb{R}^k.$$

The first inequality follows from the triangular inequality, i.e., we have $\big||a|-|b|\big|\leq\big|a-b\big|$.

We can now compute an upper bound on $E(t)$:

$$\begin{aligned}
&\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k}e^{-\frac{\|\boldsymbol{x}\|_2^2}{t}}\Big|\big|f(\boldsymbol{x})-f(\mathbf{0})\big|-\big|\langle\nabla_{\mathbb{R}^k}f(\mathbf{0}),\boldsymbol{x}\rangle\big|\Big|\mathrm{d}\boldsymbol{x}\\
\leq\;&\lim_{t\to0}\frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}\in\mathbb{R}^k}e^{-\frac{\|\boldsymbol{x}\|_2^2}{t}}C\left\|\boldsymbol{x}\right\|_2^2\mathrm{d}\boldsymbol{x}\\
=\;&\lim_{t\to0}C\frac{1}{t^{\frac{k+1}{2}}}\frac{k}{2}t^{\frac{k+2}{2}}\pi^{\frac{k}{2}}=\lim_{t\to0}C\frac{k}{2}t^{\frac{1}{2}}\pi^{\frac{k}{2}}=0
\end{aligned}$$

The application of Lemma 14 concludes the proof. □

6.5 Proof of Theorem 23

Proof. The proof uses the ingredient of Theorem 22. We have

$$\begin{aligned}
&\lim_{N\to\infty}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\frac{1}{N^2}\left\|\boldsymbol{f}\right\|_{TV-\mathcal{G}}\\
=\;&\lim_{N\to\infty}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\frac{1}{N^2}\sum_{n=1}^{N}\sum_{i=1}^{N}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t_N}}\big|\big(f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\big)\big| &(6.28)\\
=\;&\lim_{t\to0}\frac{1}{t_N^{\frac{k+1}{2}}2^{\frac{k+1}{2}}\pi^{\frac{k-1}{2}}}\int_{\boldsymbol{x}_i\in\mathcal{M}}\int_{\boldsymbol{x}_n\in\mathcal{M}}e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}}\big|\big(f(\boldsymbol{x}_i)-f(\boldsymbol{x}_n)\big)\big|\,\mathrm{d}\mu(\boldsymbol{x}_n)\mathrm{d}\mu(\boldsymbol{x}_i) &(6.29)\\
=\;&\lim_{t\to0}\int_{\boldsymbol{x}_i\in\mathcal{M}}\left\|\nabla_{\mathcal{M}}f(\boldsymbol{x}_i)\right\|_2\mathrm{d}\mu(\boldsymbol{x}_i) &(6.30)\\
=\;&\frac{1}{\mathrm{vol}(\mathcal{M})}\left\|f\right\|_{TV-\mathcal{M}}, &(6.31)
\end{aligned}$$

where

- (6.28) and (6.31) are obtained by definition,
- (6.29) follows from Lemma 11,
- and (6.30) using the equality between (6.19) and (6.25) proved in Theorem 22.

□

Lemma 11. *Define* $t_N = \left(\frac{1}{N}\right)^{\frac{2}{k+1+\alpha}}$ *with* $\alpha > 0$. *Given that the samples* $\boldsymbol{x}_i, \boldsymbol{x}_n$ *are uniformly distributed on the manifold* $\mathcal{M}$ *we have:*

$$\begin{aligned}
&\lim_{N\to\infty} \frac{1}{t_N^{\frac{k+1}{2}}} \frac{1}{N^2} \sum_{n=1}^{N}\sum_{i=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t_N}} \left|\left(f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\right)\right| \\
=&\lim_{t\to 0} \frac{1}{t^{\frac{k+1}{2}}} \int_{\boldsymbol{x}_1\in\mathcal{M}}\int_{\boldsymbol{x}_2\in\mathcal{M}} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}\|_2^2}{2t}} \left|\left(f(\boldsymbol{x}_1)-f(\boldsymbol{x}_2)\right)\right| d\mu(\boldsymbol{x}_1)d\mu(\boldsymbol{x}_2)
\end{aligned}$$

We start by computing the expected value of the sum:

$$\mathbb{E}\left[\frac{1}{N^2}\sum_{n=1}^{N}\sum_{i=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t}} \left|f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\right|\right] = \frac{1}{t^{\frac{k+1}{2}}}\int_{\boldsymbol{x}_1\in\mathcal{M}}\int_{\boldsymbol{x}_2\in\mathcal{M}} e^{-\frac{\|\boldsymbol{x}_1-\boldsymbol{x}_2\|_2^2}{2t}} \left|\left(f(\boldsymbol{x})-f(\boldsymbol{x}_i)\right)\right| d\mu(\boldsymbol{x}_1)d\mu(\boldsymbol{x}_2).$$

Then we apply the Hoeffding's inequality to find:

$$\mathbb{P}\left[\frac{1}{t^{\frac{k+1}{2}}}\left|\frac{1}{N^2}\sum_{n=1}^{N}\sum_{i=1}^{N} e^{-\frac{\|\boldsymbol{x}_n-\boldsymbol{x}_i\|_2^2}{2t}} \left|f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\right| - \mathbb{E}\left[\frac{1}{N^2}\sum_{n=1}^{N}\sum_{i=1}^{N} e^{-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_n\|_2^2}{2t}} \left|f(\boldsymbol{x}_n)-f(\boldsymbol{x}_i)\right|\right]\right| > \epsilon\right] \leq 2e^{-\frac{1}{2}\epsilon^2 t^{k+1}N^2}$$

Let us choose $t_N = \left(\frac{1}{N}\right)^{\frac{2}{k+1+\alpha}}$ *with* $\alpha > 0$. *We observe that the right term goes to* 0 *as* $N \to \infty$ *which implies the desired result.*

7 Discussion

7.1 Future directions

This dissertation does not escape the universal rule stating that *a thesis is never finished.* If the days were lasting longer, we would extend this work in the following directions.

Sampling graph signals: Developing sampling techniques for graph signals is one consistent direction to extend the results of this thesis. On the one hand, the probabilistic framework of stationarity developed in Chapter 5 is particularly suitable to define optimal sampling algorithms as 1) it fully characterizes the intrinsic signal correlations and 2) it proposes an optimal estimator for general learning problems. On the other hand, the random sampling scheme presented in Example 20, where samples are drawn randomly with probability proportional the norm of localized kernels, shows promiseful results. Assembling these two facts, the logical direction is to show that given a stationary signal with PSD g^2, the optimal random sampling weights are $p_i = \frac{\|\mathcal{T}_i^G g\|_2^2}{\|\boldsymbol{\lambda}\|_2^2}$. Some preliminary results are already available in [79, Theorem 1] where this sampling scheme is proved to embed the data given a sufficient number of measurements. To obtain a fully coherent sampling theory, we still need to bound the reconstruction error of an estimator such as (5.16).

Given a stationary signal, active sampling methods can also benefit directly from the localization operator as it provides a direct access to the correlation between the vertices. One way to do it is to generalize the results of [137] to the graph settings. It would result in an accurate but inefficient methods. Schemes to improve the efficiency of this approach are given in [79], making active sampling on graphs an exciting topic too.

Graph out of sample extension: In order to transfer GSP techniques in a semi-supervised learning setting, it is important to focus on the case, where the graph is constructed from a point cloud and where we are searching for a global solution. In Chapter 6 for example, we have used the total variation norm on graphs to regularize our problem. Similarly, we could use the concept of stationarity for manifold regularization. Nevertheless, it is cumbersome to rely

on a predefined RKHS to compute the solution at any points and it would be very convenient to directly extend the solution to the new samples without using any predefined RKHS kernel. One solution is to perform as follows. First, instead of the continuous RKHS kernel used classically, we use a discrete RKHS graph kernel adapted to the graph signal (similarly to [103]). Second, we compute the solution on the graph. Finally, we extend the RKHS graph kernel outside of the graph to recover the global solution. Utilizing the results [8, 9, 132], we have already[1] constructed a method that extends a graph kernel to new samples using a spectral approach. The results are similar to those of the Nyström interpolation method [10]. At this point, the major task resides mainly in proving convergence results similarly to [121, 119, 120], i.e., given a number of labeled and unlabeled samples, we want to bound the estimation error of an estimator.

Using graph stationarity in anomaly detection: In anomaly detection, one tries to detect if a sample statistically differs from a testing group. In general, the number of wrong samples is insufficient to model the abnormal class. As a result, many techniques try to estimate the distribution of valid samples and test if a new sample statistically belongs to it. Motivated by the results of Chapter 5 and Section 5.7, we can build a covariance estimator and thus parametrize a Gaussian distribution with a small number of samples. First, using the valid samples, a nearest neighbors graph is constructed by connecting similar features. Then, the graph PSD is estimated using the same dataset. Eventually, using the PSD and the graph, a method has to be constructed to check if a new sample belongs the distribution. The proposed method remains over-constraint for many problems as most of the data will only be close to stationary on a nearest neighbor graph. However, if the number of features is much larger than the size of the training set, this technique may be substantially more accurate than other ones.

7.2 Considerations on Graph Signal Processing

After a few years of existence, hundreds of contributions, its own yearly workshop, many conferences special sessions and an almost dedicated journal, GSP is not really emerging anymore and it occupies some space in the Data Science community. Nevertheless, there is still no application where GSP provides the most elegant and prevalent solution yet. To get closer, our feeling is that we should 1) work on Machine Learning applications that are not directly related to graphs, and 2) find new solutions to learn graphs as suggested in [54].

The reproducible research : During the 4 years of our thesis, we have seriously tried to promote reproducible research. Beside distributing the code to reproduce the results of our publications, we have created, maintained and promoted two open-source software projects: the UNLocBoX [87] and GSPBOX [86]. While this effort has somehow paid off, we have not met our goal, i.e., to make most of the GSP research reproducible. Today we believe that a

[1] Unfortunately, there is no publication available yet.

different approach is needed to fulfill this objective. In order to encourage the community to publish its code and reward the scientists doing it, we should create a standard database of GSP problems with some baseline results.

We thank the reader for staying this far with us.

A A gentle introduction to graph signal processing

A.1 Computation of the divergence operator

Proof. Let us develop our hypothesis $\langle \nabla_{\mathcal{G}} \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{E}|}} = \langle \boldsymbol{x}, \mathrm{div}_{\mathcal{G}} \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{V}|}}$:

$$
\begin{aligned}
& & \langle \nabla_{\mathcal{G}} \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{E}|}} &= \langle \boldsymbol{x}, \mathrm{div}_{\mathcal{G}} \boldsymbol{y} \rangle_{\mathbb{R}^{|\mathcal{V}|}} \\
\Rightarrow & & \tfrac{1}{2} \textstyle\sum_n \sum_i \sqrt{\boldsymbol{W}[i,j]} \left(\boldsymbol{x}[n] - \boldsymbol{x}[i] \right) \boldsymbol{y}[i,n] &= \displaystyle\sum_n \boldsymbol{x}[n] \mathrm{div}_{\mathcal{G}} \boldsymbol{y}[n] \\
\Leftrightarrow & & \tfrac{1}{2} \left(\textstyle\sum_n \sum_i \sqrt{\boldsymbol{W}[i,n]} \boldsymbol{x}[n] \boldsymbol{y}[i,n] - \sum_n \sum_i \sqrt{\boldsymbol{W}[i,n]} \boldsymbol{x}[i] \boldsymbol{y}[i,n] \right) &= \displaystyle\sum_n \boldsymbol{x}[n] \mathrm{div}_{\mathcal{G}} \boldsymbol{y}[n] \\
\Leftrightarrow & & \tfrac{1}{2} \left(\textstyle\sum_n \boldsymbol{x}[n] \sum_i \sqrt{\boldsymbol{W}[i,n]} \boldsymbol{y}[i,n] - \sum_n \sum_i \boldsymbol{x}[n] \sqrt{\boldsymbol{W}[n,i]} \boldsymbol{y}[n,i] \right) &= \displaystyle\sum_n \boldsymbol{x}[n] \mathrm{div}_{\mathcal{G}} \boldsymbol{y}[n] \\
\Leftrightarrow & & \tfrac{1}{2} \textstyle\sum_n \boldsymbol{x}[i] \left(\sum_i \sqrt{\boldsymbol{W}[i,n]} \boldsymbol{y}[i,n] - \sqrt{\boldsymbol{W}[n,i]} \boldsymbol{y}[n,i] \right) &= \displaystyle\sum_n \boldsymbol{x}[n] \mathrm{div}_{\mathcal{G}} \boldsymbol{y}[n]
\end{aligned}
$$

where the factor $\frac{1}{2}$ follows from the fact that we count each edges twice. As a result, one valid way to define the divergence operator is

$$\mathrm{div}_{\mathcal{G}} \boldsymbol{y}[n] = \frac{1}{2} \sum_i \sqrt{\boldsymbol{W}[i,n]} \boldsymbol{y}[i,n] - \sqrt{\boldsymbol{W}[n,i]} \boldsymbol{y}[n,i]$$

□

A.2 Laplacian

Proof. We assume the graph to be undirected, i.e., $\boldsymbol{W}[i,n] = \boldsymbol{W}[n,i]$.

$$\begin{aligned}
(\boldsymbol{L}\boldsymbol{x})[n] &= \left(\mathrm{div}_{\mathcal{G}} \nabla_{\mathcal{G}} \boldsymbol{x}\right)[n] \\
&= \frac{1}{2}\left(\sum_i \sqrt{\boldsymbol{W}[i,n]}\left((\nabla_{\mathcal{G}}\boldsymbol{x})[i,n] - (\nabla_{\mathcal{G}}\boldsymbol{x})[n,i]\right)\right) \\
&= \sum_i \boldsymbol{W}[i,n]\left(\boldsymbol{x}[n] - \boldsymbol{x}[i]\right) \\
&= (\boldsymbol{D} - \boldsymbol{W})\,\boldsymbol{x}[n].
\end{aligned}$$

□

B Structural clustering via the graph localization operator

B.1 Translation for graphs

Intuitively, translating a signal is equivalent to moving it in one direction. Let $\boldsymbol{x} \in \mathbb{R}^N$ be a vector containing one period of a discrete N-periodical signal. Translating $\boldsymbol{x}$ by i is simply given by:

$$T_i \boldsymbol{x}[n] = \boldsymbol{x}[\lfloor n-i \rfloor_N], \tag{B.1}$$

where $\lfloor n \rfloor_N = n - N \text{ floor } \left(\frac{n-1}{N}\right)$ is an operation that maps any index n to the range $1 \dots N$. This operation does not affect the shape of the signal $\boldsymbol{x}$ and can be seen as a multiplication by an eigenvector in the spectral domain

$$\widehat{T_i \boldsymbol{x}}[\ell] = e^{2\pi j \frac{i\ell}{N}} \hat{\boldsymbol{x}}[\ell]. \tag{B.2}$$

B.1.1 Generalizations of translation for graphs

Based on (B.1) and (B.2), multiple generalizations of translations have been proposed. Unfortunately, none of them satisfies the two most essential properties that one would naturally expect from a translation: 1) Isometric (the translation conserves the energy of the signal), and 2) Localization (localized signals remain concentrated after translation).

Previous works use two different paths to generalize translation for graphs. In the first, one tries to generalize a unit of translation i.e., T_1. This approach allows for the composition $T_i T_n \boldsymbol{x} = T_{i+n} \boldsymbol{x}$, but hinders localization properties (here i, n are not node indices but amounts of translation). In the second way, one tries to generalize the translation as the shift to a specified node, which does not allow for composition because of the irregular structure of graphs. However, in that case, the resulting signal can (under some hypotheses) be localized.

The graph shift operator

Based on the so-called *Algebraic Signal Processing framework* [95], the authors of [108] proposed to use the weight (adjacency) matrix as a translation operator, i.e., a unit of translation is defined as:

$$\boldsymbol{T}_1\boldsymbol{x} = \boldsymbol{W}\boldsymbol{x} \tag{B.3}$$

In case the graph is a directed "ring" with weight matrix

$$\boldsymbol{W}[i,n] = \begin{cases} 1 & \text{if } \lfloor i-n \rfloor_N = 1 \\ 0 & \text{otherwise,} \end{cases}$$

the graph shift operator becomes equivalent to the traditional translation.

A careful analysis shows that the shift operator diffuses, for most of the graphs, the energy along the edges. In fact, for a symmetric graph, it is related to the graph Laplacian as $\boldsymbol{W} = \boldsymbol{D} - \boldsymbol{L}$. For a d-regular graph,[1] the shift operator is a low pass graph filter with function $h_l(x) = d - x$. In general the graph shift is not isometric and (as a diffusion operator) does not preserve localization.

An isometric graph translation operator

Another generalization of the translation operator is proposed in [44, 47]. It has the particularity to conserve energy.

Definition 26. *For a graph signal $\boldsymbol{x}$, a unit of translation is defined as:*

$$\boldsymbol{T}_B s := \exp\left(j2\pi\sqrt{\frac{\boldsymbol{L}}{\rho_{\mathcal{G}}}}\right)\boldsymbol{x} = b(\boldsymbol{L})\boldsymbol{x}, \tag{B.4}$$

where $b(x) = \exp\left(j2\pi\sqrt{\frac{x}{\rho_{\mathcal{G}}}}\right)$. The constant $\rho_{\mathcal{G}}$ is an upper bound on the maximum eigenvalue of the graph Laplacian $\lambda_{\max}$ defined as

$$\rho_{\mathcal{G}} := \max_{i\in\mathcal{V}} \sqrt{2d[i](d[i] + \bar{d}[i])},$$

where $\bar{d}[i] = \frac{\sum_{n=1}^{N} W[i,n]d[n]}{d[i]}$.

While this operator conserves the energy of the signal ($\|T_B\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$), it does not have localization properties.

[1] A graph with constant degree d

The generalized graph translation operator

From a different perspective, leveraging the well-defined graph Fourier transform, Shuman et al. define the generalized translation for graph signals as the convolution with a Kronecker delta [117, Equation 26]. The graph convolution $*$ being an element-wise multiplication in the spectral domain, they obtain the following.

Definition 27. *For a graph signal $\boldsymbol{x}$ and a vertex i, the generalized graph translation operator reads:*

$$\boldsymbol{T}_i \boldsymbol{x}[n] := (\boldsymbol{x} * \delta_i)[n] = \sum_{\ell=0}^{N-1} \hat{\boldsymbol{x}}[\ell] \boldsymbol{u}_\ell^*[i] \boldsymbol{u}_\ell[n]. \tag{B.5}$$

Unfortunately, the generalized translation operator does not perform what we would intuitively expect from it, i.e., it does not translate a signal $\boldsymbol{x}$ from node n to node i. Instead when $\hat{\boldsymbol{x}}$ changes smoothly across the frequencies (Theorem 3), then $\boldsymbol{T}_i \boldsymbol{x}$ is localized around node i, while $\boldsymbol{x}$ is in general not localized at a particular node or set of nodes.

Problem with translation

While all three definitions generalize classical translation, none of them is making what one would naturally expect from translation, i.e., move a signal localized around a node i toward another node n. We believe that translation is not necessary to build GSP and we suggest dropping it in favor of the localization operator which is inspired by the generalized graph translation operator.

B.2 An alternative generalized translation operator

Interestingly, one could define a generalized translation operator as inversing a localization operation and re-localizing the mass around another node. One way to do this is to define:

$$\boldsymbol{T}_{i \to n} \boldsymbol{x}[m] := \boldsymbol{T}_n \boldsymbol{T}_i^{-1} \boldsymbol{x}[m] = \sum_{\ell=0}^{N-1} \hat{\boldsymbol{x}}[\ell] \left(\boldsymbol{u}_\ell^*[i] \right)^{-1} \boldsymbol{u}_\ell^*[n] \boldsymbol{u}_\ell[m]. \tag{B.6}$$

This operator moves some mass from vertex i to vertex n and satisfies

$$\boldsymbol{T}_{i \to n} \mathcal{T}_i^G g = \mathcal{T}_n^G g. \tag{B.7}$$

This equation leads to

$$\boldsymbol{T}_i^{-1} \mathcal{T}_i^G g = \boldsymbol{T}_n^{-1} \mathcal{T}_n^G g = \sum_{\ell=0}^{N-1} g(\lambda_\ell) \boldsymbol{u}_\ell, \quad \forall v_i \in \mathcal{V} \tag{B.8}$$

Unfortunately, this operator is not well defined as $\boldsymbol{u}_\ell[i]$ can be equal to 0. Furthermore, in practice one does not need this operator the localization operator is sufficient.

B.3 Proof of Theorem 1

Proof. Since $|g(\lambda_\ell)| < \infty$, there exist a polynomial p of order N such that $g(\lambda_\ell) = p(\lambda_\ell)$. By definition,

$$\mathcal{T}_i^G g[n] = \mathcal{T}_i^G p[n] = p(\boldsymbol{L})[i,n]$$

We now show that $p(\boldsymbol{L})$ is independent of the eigenvector bases. Any valid basis $\boldsymbol{U}$ satisfies $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^*$. We derive

$$p(\boldsymbol{L}) = \boldsymbol{U}p(\boldsymbol{\Lambda})\boldsymbol{U}^* = \boldsymbol{U}\left(\sum_{m=0}^{N}\alpha_m\boldsymbol{\Lambda}^m\right)\boldsymbol{U}^* = \sum_{m=0}^{N}\alpha_m\boldsymbol{U}\boldsymbol{\Lambda}^m\boldsymbol{U}^* = \sum_{m=0}^{N}\alpha_m\boldsymbol{L}^m. \tag{B.9}$$

Independently from the choice of $\boldsymbol{U}$, we find that $g(\boldsymbol{L}) = \sum_{m=0}^{N}\alpha_m\boldsymbol{L}^m$, which concludes the proof. □

B.4 Proof of Theorem 2

Proof.

1. Using the definition and (2.18), we find:

$$\mathcal{T}_i^G g[n] = \frac{1}{N}\sum_{k=1}^{N} g(\lambda_k)e^{-j2\pi\frac{ki}{N}}e^{j2\pi\frac{kn}{N}} = \frac{1}{N}\sum_{k=1}^{N} g(\lambda_k)e^{-j2\pi\frac{kN}{N}}e^{j2\pi\frac{k(n-i)}{N}} = \mathcal{T}_N^G g[n-i].$$

2. From Theorem 1, we know that the localization operator is independent of the eigenvector basis. Let us choose a real basis. We know that such a basis exists since the Laplacian $\boldsymbol{L}$ is a real symmetric matrix. From Definition 15, if the graph Fourier basis $\boldsymbol{U}$ is real, then $\mathcal{T}_g^G i$ will also be real for $g : \mathbb{R}_+ \to \mathbb{R}$. Instead of this argument, one can also use the next part of the proof (B.10) to show that the localization operator provides a real result.

3. From the first part, we know that $\mathcal{T}_i^G g[n] = \mathcal{T}_N^G g[n-i]$. As a result, we need to show that $\mathcal{T}_N^G g$ is symmetric around the node N or 0. The trick is to group the factors corresponding to the same eigenvalue together, i.e., from (2.19) $\lambda_k = \lambda_{N-k}$. If N is odd, we

obtain:

$$\begin{aligned}
\mathcal{T}_N^G g[n] &= \sum_{\ell=0}^{N-1} g(\lambda_\ell) u_\ell[n] \\
&= \frac{1}{\sqrt{N}}\left(g(0) + \sum_{\ell=1}^{(N-1)/2} g(\lambda_\ell)\exp\left(2\pi j\frac{\ell n}{N}\right) + \sum_{\ell=1}^{(N-1)/2} g(\lambda_\ell)\exp\left(2\pi j\frac{(N-\ell)n}{N}\right)\right) \\
&= \frac{1}{\sqrt{N}}\left(g(0) + \sum_{\ell=1}^{(N-1)/2} g(\lambda_\ell)\left(\exp\left(2\pi j\frac{\ell n}{N}\right) + \exp\left(2\pi j\frac{(N-\ell)n}{N}\right)\right)\right) \\
&= \frac{1}{\sqrt{N}}\left(g(0) + \sum_{\ell=1}^{(N-1)/2} g(\lambda_\ell)\left(\exp\left(2\pi j\frac{\ell n}{N}\right) + \exp\left(-2\pi j\frac{\ell n}{N}\right)\right)\right) \\
&= \frac{1}{\sqrt{N}}\left(g(0) + 2\sum_{\ell=1}^{(N-1)/2} g(\lambda_\ell)\cos\left(2\pi j\frac{\ell n}{N}\right)\right) \qquad \text{(B.10)}
\end{aligned}$$

Since all cosines are symmetric, the result will also be symmetric. When N is even, the same principle can be used with an extra term $(-1)^n g(1)$.

4. From (B.10), we observe that $\mathcal{T}_N^G g = g(\boldsymbol{\lambda})\boldsymbol{B}$ where $\boldsymbol{B}$ is a matrix containing cosines with all possible frequencies. These cosines span the space of the symmetric functions. As a result, if $\boldsymbol{x}$ is symmetric, we can find the coefficients $g(\boldsymbol{\lambda})$ such that $g(\boldsymbol{\lambda})\boldsymbol{B} = \boldsymbol{x} = \mathcal{T}_N^G g$.

□

B.5 Proof of Theorem 5

Proof. Let us first prove that the estimator is unbiased. We have

$$\mathbb{E}\left[\dot{\boldsymbol{S}_{\mathcal{G}}}^2[i,k]\right] = \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}\left[(\boldsymbol{x}_{k,m}[i])^2\right] = \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}\left[\left|\langle \boldsymbol{w}_m, {\mathcal{T}_i}^G g\rangle\right|^2\right] = \frac{1}{M}\sum_{m=1}^{M}\left\|{\mathcal{T}_i}^G g_k\right\|_2^2 = \left\|{\mathcal{T}_i}^G g_k\right\|_2^2,$$

where the third equality follows from Lemma 3. For the variance, let us first focus on the case where $M = 1$.

$$\begin{aligned}
\mathbf{Var}\left[\boldsymbol{S}_{\mathcal{G}}^2[\dot{i},k]\right] &= \mathbb{E}\left[\left(\dot{\boldsymbol{S}_{\mathcal{G}}}[i,k]\right)^2\right] - \left(\mathbb{E}\left[\dot{\boldsymbol{S}_{\mathcal{G}}}[i,k]\right]\right)^2 \\
&= \mathbb{E}\left[\left|\langle \boldsymbol{w}_m, {\mathcal{T}_i}^G g\rangle\right|^4\right] - \left\|{\mathcal{T}_i}^G g_k\right\|_2^4 \\
&= \left\|{\mathcal{T}_i}^G g_k\right\|_4^4 (m_4 - 3) + 2\left\|{\mathcal{T}_i}^G g\right\|_2^4
\end{aligned}$$

When M samples are chosen, then the variance is reduced by a factor of M.

The computation of $\mathbb{E}\left[\left|\langle \boldsymbol{w}_m, {\mathcal{T}_i}^G g\rangle\right|^4\right]$ is detailed below. It depends both on the second mo-

ment and on the first moment of the distribution.

$$\begin{aligned}
\mathbb{E}\left[\left|\sum_n w_n T_i g[n]\right|^4\right] &= \mathbb{E}\left[\sum_{n=1}^{N}\sum_{m=1}^{N}\sum_{o=1}^{N}\sum_{p=1}^{N} \boldsymbol{w}[n]\boldsymbol{w}[m]\boldsymbol{w}[o]\boldsymbol{w}[p]\mathcal{T}_i{}^G g[n]\mathcal{T}_i{}^G g[m]\mathcal{T}_i{}^G g[o]\mathcal{T}_i{}^G g[p]\right] \\
&= \sum_{n=1}^{N}\left(\mathcal{T}_i{}^G g[n]\right)^4 \mathbb{E}\left[\boldsymbol{w}^4[n]\right] + 3\sum_{n=1}^{N}\sum_{m\neq n}^{N}\left(\mathcal{T}_i{}^G g[n]\right)^2\left(\mathcal{T}_i{}^G g[m]\right)^2 \mathbb{E}\left[\boldsymbol{w}^2[n]\right]\mathbb{E}\left[\boldsymbol{w}^2[m]\right] \\
&= m_4\left\|\mathcal{T}_i{}^G g[n]\right\|_4^4 + 3\sum_{n=1}^{N}\sum_{m\neq n}^{N}\left(\mathcal{T}_i{}^G g[n]\right)^2\left(\mathcal{T}_i{}^G g[m]\right)^2 \\
&= m_4\left\|\mathcal{T}_i{}^G g[n]\right\|_4^4 + 3\sum_{n=1}^{N}\sum_{m=1}^{N}\left(\mathcal{T}_i{}^G g[n]\right)^2\left(\mathcal{T}_i{}^G g[m]\right)^2 - 3\sum_{n=1}^{N}\left(\mathcal{T}_i{}^G g[n]\right)^4 \\
&= (m_4-3)\left\|\mathcal{T}_i{}^G g[n]\right\|_4^4 + 3\left\|\mathcal{T}_i{}^G g\right\|_2^4
\end{aligned}$$

The second line is obtained from the fact that $\mathbb{E}\left[\boldsymbol{w}[n]\boldsymbol{w}[m]\boldsymbol{w}[o]\boldsymbol{w}[p]\right] \neq 0$ only if $(n,m) = (o,p)$, $(n,m) = (p,o)$ or $(n,p) = (m,o)$. □

C Global and local uncertainty principles for graph signals

C.1 Hausdorff-Young inequalities for graph signals

To prove the Hausdorff-Young inequalities for graph signals, we start by restating the Riesz-Thorin interpolation theorem, which can be found in [98, Section IX.4]. **This theorem is valid for any measure spaces with σ-finite measures, and hence in the discrete finite dimensional case.**

Theorem 24 (Riesz-Thorin)**.** *Assume $\boldsymbol{S}$ is a bounded linear operator from ℓ^{p_1} to ℓ^{p_2} and from ℓ^{q_1} to ℓ^{q_2}; i.e., there exist constants B_p and B_q such that*

$$\|\boldsymbol{S}\boldsymbol{x}\|_{p_2} \leq B_p \|\boldsymbol{x}\|_{p_1} \quad \text{and} \quad \|\boldsymbol{S}\boldsymbol{x}\|_{q_2} \leq B_q \|\boldsymbol{x}\|_{q_1}.$$

Then for any t between 0 and 1, $\boldsymbol{S}$ is also a bounded operator from ℓ^{r_1} to ℓ^{r_2} :

$$\|\boldsymbol{S}\boldsymbol{x}\|_{r_2} \leq B_r \|\boldsymbol{x}\|_{r_1},$$

with

$$\frac{1}{r_1} = \frac{t}{p_1} + \frac{1-t}{q_1}, \qquad \frac{1}{r_2} = \frac{t}{p_2} + \frac{1-t}{q_2},$$

and

$$B_r = B_p^t B_q^{1-t}.$$

We shall also need the following reverse form of the result:

Corollary 4. *Assume $\boldsymbol{S}$ is a bounded invertible linear operator from ℓ^{p_1} to ℓ^{p_2} and from ℓ^{q_1} to ℓ^{q_2}, with bounded left-inverse from ℓ^{p_2} to ℓ^{p_1} and from ℓ^{q_2} to ℓ^{q_1}; i.e., there exist constants B_p and B_q such that*

$$\left\|\boldsymbol{S}^{-1}\boldsymbol{y}\right\|_{p_1} \leq B_p \left\|\boldsymbol{y}\right\|_{p_2} \quad \text{and} \quad \left\|\boldsymbol{S}^{-1}\boldsymbol{y}\right\|_{q_1} \leq B_q \left\|\boldsymbol{y}\right\|_{q_2}, \tag{C.1}$$

or, equivalently, there exist constants A_p and A_q such that

$$\|\boldsymbol{S}\boldsymbol{x}\|_{p_2} \geq A_p \|\boldsymbol{x}\|_{p_1} \quad \textit{and} \quad \|\boldsymbol{S}\boldsymbol{x}\|_{q_2} \geq A_q \|\boldsymbol{x}\|_{q_1}. \tag{C.2}$$

Then for any t between 0 and 1,

$$\|\boldsymbol{S}\boldsymbol{x}\|_{r_2} \geq A_r \|\boldsymbol{x}\|_{r_1}, \tag{C.3}$$

with

$$\frac{1}{r_1} = \frac{t}{p_1} + \frac{1-t}{q_1}, \qquad \frac{1}{r_2} = \frac{t}{p_2} + \frac{1-t}{q_2},$$

and

$$A_r = A_p^t A_q^{1-t}.$$

Proof. If $\boldsymbol{S}$ is invertible and has a left-inverse $\boldsymbol{S}^{-1}$ that satisfies $\boldsymbol{S}^{-1}\boldsymbol{S}\boldsymbol{x} = \boldsymbol{x}$ for all $\boldsymbol{x}$, then the equivalence of (C.1) and (C.2) follows from taking $\boldsymbol{y} = \boldsymbol{S}\boldsymbol{x}$, $\boldsymbol{x} = \boldsymbol{S}^{-1}\boldsymbol{y}$, $A_p = B_p^{-1}$, and $A_q = B_q^{-1}$. The proof of (C.3) follows from the application of Theorem 24, with $\boldsymbol{S}$ replaced by $\boldsymbol{S}^{-1}$ and $\boldsymbol{x}$ by $\boldsymbol{S}\boldsymbol{x}$. □

Proof of Theorem 7 (Hausdorff-Young inequalities for graph signals). First, we have the Parseval equality $\|\boldsymbol{x}\|_2^2 = \|\hat{\boldsymbol{x}}\|_2^2$. Second, we have

$$\|\hat{\boldsymbol{x}}\|_\infty = \max_\ell \left| \sum_{n=1}^N \boldsymbol{u}_\ell^*[n]\boldsymbol{x}[n] \right| \leq \max_\ell \sum_{n=1}^N \left| \boldsymbol{u}_\ell^*[n]\boldsymbol{x}[n] \right| \leq \mu_{\mathcal{G}} \sum_{n=1}^N |f(n)| = \mu_{\mathcal{G}} \|\boldsymbol{x}\|_1.$$

Applying the Riesz-Thorin theorem with $p_1 = 2$, $p_2 = 2$, $B_p = 1$, $q_1 = 1$, $q_2 = \infty$, $B_q = \mu_{\mathcal{G}}$, $t = \frac{2}{q}$, $r_1 = p$, and $r_2 = q$ leads to the first inequality (4.9). The proof of the converse is similar, as we have

$$\|\boldsymbol{x}\|_\infty = \max_i \left| \sum_{\ell=0}^{N-1} \boldsymbol{u}_\ell[i]\hat{\boldsymbol{x}}[\ell] \right| \leq \max_i \sum_{\ell=0}^{N-1} |\boldsymbol{u}_\ell[i]\hat{\boldsymbol{x}}[\ell]| \leq \mu_{\mathcal{G}} \sum_{\ell=0}^{N-1} |\hat{\boldsymbol{x}}[\ell]| = \mu_{\mathcal{G}} \|\hat{\boldsymbol{x}}\|_1.$$

The graph Fourier transform is invertible, so (4.10) then follows from Corollary 4, with $p_1 = \infty$, $p_2 = 1$, $A_p = \mu_{\mathcal{G}}^{-1}$, $q_1 = 2$, $q_2 = 2$, $A_q = 1$, $t = \frac{2}{q} - 1$, $r_1 = p$, and $r_2 = q$. □

C.2 Variations of Lieb's uncertainty principle

C.2.1 Generalization of Lieb's uncertainty principle to frames

Proof of Theorem 9. Let $\mathcal{D} = \{\boldsymbol{g}_{i,k}\}$ be a frame of atoms in $\mathbb{C}^N$, with lower and upper frame bounds A and B, respectively. We show the following two inequalities, which together yield (4.18). First, for any signal $\boldsymbol{x} \in \mathbb{C}^N$ and any $p \geq 2$,

$$s_p(\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}) = \frac{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_p}{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_2} \leq \frac{B^{\frac{1}{p}}}{A^{\frac{1}{2}}} \left(\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2\right)^{1-\frac{2}{p}} \tag{C.4}$$

Second, for any signal $\boldsymbol{x} \in \mathbb{C}^N$ and any $1 \leq p \leq 2$,

$$\frac{1}{s_p(\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x})} = \frac{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_p}{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_2} \geq \frac{A^{\frac{1}{p}}}{B^{\frac{1}{2}}} \left(\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2\right)^{1-\frac{2}{p}} \tag{C.5}$$

For any $\boldsymbol{x}$, the frame $\mathcal{D}$ satisfies

$$\sqrt{A}\,\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_2 \leq \sqrt{B}\,\|\boldsymbol{x}\|_2\,. \tag{C.6}$$

The computation of the sup-norm gives

$$\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_\infty = \max_{i,k} \left|\langle \boldsymbol{x}, \boldsymbol{g}_{i,k}\rangle\right| \leq \|\boldsymbol{x}\|_2 \max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2\,. \tag{C.7}$$

From (C.6), $\boldsymbol{A}_{\mathcal{D}}$ is a linear bounded operator form ℓ_2 to ℓ_2 by $\sqrt{B}$. Similarly, from (C.7), this operator is also bounded from ℓ_2 to ℓ_∞ by $\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2$. Interpolating between ℓ_2 and ℓ_∞ with the Riesz-Thorin theorem leads to

$$\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_p \leq B^{\frac{1}{p}} \left(\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2\right)^{1-\frac{2}{p}} \|\boldsymbol{x}\|_2\,. \tag{C.8}$$

We combine (C.6) and (C.8) to obtain (C.4). The second inequality (C.5) is obtained using the following instance of Hölder's inequality:

$$\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_\infty \|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_1\,,$$

which implies that

$$\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_1 \geq \frac{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_2^2}{\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_\infty} \geq \frac{A\,\|\boldsymbol{x}\|_2}{\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2}. \tag{C.9}$$

We then use Corollary 4, the converse of Riesz-Thorin, to interpolate between (C.9) and (C.6), and we find for $p \in [1,2]$:

$$\|\boldsymbol{A}_{\mathcal{D}}\boldsymbol{x}\|_p \geq A^{\frac{1}{p}} \left(\max_{i,k} \left\|\boldsymbol{g}_{i,k}\right\|_2\right)^{1-\frac{2}{p}} \|\boldsymbol{x}\|_2\,. \tag{C.10}$$

Combining (C.10) with the second inequality in (C.6) yields (C.5). □

C.2.2 Discrete version of Lieb's uncertainty principle

Proof of Theorem 8. Theorem 8 is actually a particular case of Theorem 9. To see why, we need to understand the transformation between the graph framework used in this contribution and the classical discrete periodic case. The DFT basis vectors $\left\{\boldsymbol{u}_k(n) = \frac{1}{\sqrt{N}} \exp\left(\frac{j2\pi kn}{N}\right)\right\}_{k=0,1,\ldots,N-1}$ can also be chosen as the eigenvectors of the graph Laplacian for a ring graph with N vertices [125]. The frequencies of the DFT, which correspond up to a sign to the inverse of the period of the eigenvectors, are not the same as the graph Laplacian eigenvalues on the ring graph, which are all positive. We can, however, form a bijection between the set of graph Laplacian eigenvalues and the set of N frequencies of the DFT, by associating one member from each set sharing the same eigenvector. At this point, instead of considering graph filters as continuous functions evaluated on the Laplacian eigenvalues, we can define a graph filter as a mapping from each individual eigenvalue to a complex number. Note that an eigenvalue with multiplicity 2 can have two different outputs (e.g., $\lambda_3 = \lambda_4 = 1$, but the filter has different values at λ_3 and λ_4). With this bijection and view of the graph spectral domain, we can recover the classical discrete periodic setting by forming a ring graph with N vertices. In order to match the graph case, we normalize the atoms by a factor $\sqrt{N}$ (the modulation is a multiplication by a normalized atoms). The discrete windowed Fourier atoms are given by:

$$\boldsymbol{g}_{u,k}[n] = \boldsymbol{g}[n-u] \frac{1}{\sqrt{N}} \exp\left(\frac{j2\pi kn}{N}\right)$$

all have the same norm $N^{-\frac{1}{2}} \left\|\boldsymbol{g}\right\|_2$. Together these N^2 atoms comprise a tight frame on the ring graph with frame bounds $A = B = \left\|\boldsymbol{g}\right\|_2^2$. Inserting these values into (4.16) and (4.17) yields (4.14) and (4.15). □

For the case of $p \geq 2$, we also provide an alternative direct proof following similar ideas to those used in Lieb's proof for the continuous case [64]. The arguments below follow the sketch of the proof of Proposition 2 in [102] and supporting personal communication from Bruno Torrésani. We need two lemmas. The first one is a direct application of Theorem 7, where here $\mu_{\mathcal{G}} = 1/\sqrt{N}$.

Lemma 12. *Let $\boldsymbol{x} \in \mathbb{C}^N$ and p be the Hölder conjugate of p' ($\frac{1}{p} + \frac{1}{p'} = 1$). Then for $1 \leq p \leq 2$, we have*

$$\|\hat{\boldsymbol{x}}\|_{p'} \leq N^{\frac{1}{p'} - \frac{1}{2}} \|\boldsymbol{x}\|_p .$$

Conversely, for $2 \leq p \leq \infty$, we have

$$\|\hat{\boldsymbol{x}}\|_{p'} \geq N^{\frac{1}{p'} - \frac{1}{2}} \|\boldsymbol{x}\|_p .$$

The second lemma is an equivalent of Young's inequality in the discrete case. We denote the circular convolution between two discrete signals $\boldsymbol{x}, \boldsymbol{y}$ by $\boldsymbol{x} * \boldsymbol{y}$. The circular convolution satisfies $\widehat{\boldsymbol{x} * \boldsymbol{y}}[\ell] = \hat{\boldsymbol{x}} \cdot \hat{\boldsymbol{y}}[\ell]$.

Lemma 13. *Let $\boldsymbol{x}, \boldsymbol{y}$ be two discrete signal and $1 \leq p, q, r \leq \infty$ satisfy $1 + \frac{1}{r} = \frac{1}{p} + \frac{1}{q}$. Then*

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_r \leq \|\boldsymbol{x}\|_p \left\| \boldsymbol{y} \right\|_q .$$

Proof. The proof is based on the following inequalities [94, p. 174]

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_1 \leq \|\boldsymbol{x}\|_1 \left\| \boldsymbol{y} \right\|_1 \tag{C.11}$$

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_\infty \leq \|\boldsymbol{x}\|_\infty \left\| \boldsymbol{y} \right\|_1 \tag{C.12}$$

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_\infty \leq \|\boldsymbol{x}\|_p \left\| \boldsymbol{y} \right\|_{p'} , \tag{C.13}$$

where $\frac{1}{p} + \frac{1}{p'} = 1$. For a fixed vector $\boldsymbol{y}$, we define an operator $\boldsymbol{S}_{\boldsymbol{y}}$ by $(\boldsymbol{S}_{\boldsymbol{y}} \boldsymbol{x})[n] = (\boldsymbol{x} * \boldsymbol{y})[n]$. Using (C.11) and (C.12), we observe that this operator is bounded from ℓ^1 to ℓ^1 by $\left\| \boldsymbol{y} \right\|_1$ and from ℓ^∞ to ℓ^∞ by $\left\| \boldsymbol{y} \right\|_1$. Thus, we can apply the Riesz-Thorin theorem to this operator to get

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_p \leq \|\boldsymbol{x}\|_p \left\| \boldsymbol{y} \right\|_1 . \tag{C.14}$$

Similarly, for a fixed vector $\boldsymbol{x}$, we define another operator $T_{\boldsymbol{x}}$ by $(T_{\boldsymbol{x}} \boldsymbol{y})[n] = (\boldsymbol{x} * \boldsymbol{y})[n]$. From (C.14) and (C.13), we observe that this new operator is bounded from ℓ^1 to ℓ^p by $\|\boldsymbol{x}\|_p$ and from $\ell^{p'}$ to ℓ^∞ by $\|\boldsymbol{x}\|_p$. One more application of the Riesz-Thorin theorem leads to the desired result:

$$\left\| \boldsymbol{x} * \boldsymbol{y} \right\|_r \leq \|\boldsymbol{x}\|_p \left\| \boldsymbol{y} \right\|_q ,$$

where $1 + \frac{1}{r} = \frac{1}{p} + \frac{1}{q}$. □

Alternative proof of Theorem 8 for the case $p \geq 2$. Suppose $p > 2$ and let $\frac{1}{p} + \frac{1}{p'} = 1$. We denote

the DFT by $\mathbf{F}$. Noting that $\frac{p}{p'} > 1$, we have

$$\begin{aligned}
\left\| \boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x} \right\|_p^p &= \sum_{u=1}^{N} \sum_{k=0}^{N-1} |\boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x}[u,k]|^p \\
&= \sum_{u=1}^{N} \sum_{k=0}^{N-1} \left| \mathbf{F}(\boldsymbol{x}[\cdot]\boldsymbol{g}[u-\cdot])[k] \right|^p \\
&= \sum_{u=1}^{N} \left\| \mathbf{F}(\boldsymbol{x}[\cdot]\boldsymbol{g}[u-\cdot]) \right\|_p^p \\
&\leq N^{\frac{p}{2}} \sum_{u=1}^{N} N^{\frac{p}{p'}-\frac{p}{2}} \left\| \boldsymbol{x}[\cdot]\boldsymbol{g}[u-\cdot] \right\|_{p'}^p \qquad \text{(C.15)} \\
&= N^{\frac{p}{2}-\frac{p}{p'}} \sum_{u=1}^{N} \left(\sum_{n=1}^{N} |\boldsymbol{x}[n]\boldsymbol{g}[u-n]|^{p'} \right)^{\frac{p}{p'}} \\
&= N^{\frac{p}{2}-\frac{p}{p'}} \sum_{u=1}^{N} \left(\sum_{n=1}^{N} |\boldsymbol{x}^{p'}[n]||\boldsymbol{g}^{p'}[u-n]| \right)^{\frac{p}{p'}} \\
&= N^{\frac{p}{2}-\frac{p}{p'}} \sum_{u=1}^{N} \left((|\boldsymbol{x}^{p'}| * |\boldsymbol{g}^{p'}|)[u] \right)^{\frac{p}{p'}} \\
&= N^{\frac{p}{2}-\frac{p}{p'}} \left\| |\boldsymbol{x}^{p'}| * |\boldsymbol{g}^{p'}| \right\|_{\frac{p}{p'}}^{\frac{p}{p'}} \\
&\leq N^{\frac{p}{2}-\frac{p}{p'}} \left\| \boldsymbol{x}^{p'} \right\|_{\alpha}^{\frac{p}{p'}} \left\| \boldsymbol{g}^{p'} \right\|_{\beta}^{\frac{p}{p'}} \qquad \text{(C.16)} \\
&= N^{1-\frac{p}{2}} \left\| \boldsymbol{x}^{p'} \right\|_{\alpha}^{\frac{p}{p'}} \left\| \boldsymbol{g}^{p'} \right\|_{\beta}^{\frac{p}{p'}},
\end{aligned}$$

for any $1 \leq \alpha, \beta \leq \infty$ satisfying $\frac{1}{\alpha} + \frac{1}{\beta} = p'$. Equation (C.15) follows from the Hausdorff-Young inequality given in Lemma 12 and (C.16) follows from the Young inequality given in Lemma 13 with $r = \frac{p}{p'}$. Now we can perform a change variable $a = \alpha p'$ and $b = \beta p'$ so that $\frac{1}{a} + \frac{1}{b} = 1$, and (C.16) becomes

$$\left\| \boldsymbol{A}_{\mathcal{D}_{DWFT}} \boldsymbol{x} \right\|_p^p \leq N^{1-\frac{p}{2}} \left\| \boldsymbol{x}^{p'} \right\|_{\alpha}^{\frac{p}{p'}} \left\| \boldsymbol{g}^{p'} \right\|_{\beta}^{\frac{p}{p'}} = N^{1-\frac{p}{2}} \left\| \boldsymbol{x} \right\|_a^p \left\| \boldsymbol{g} \right\|_b^p. \qquad \text{(C.17)}$$

Finally, we take $a = b = 2$ and take the p^{th} root of (C.17) to show the first half of Theorem 8. Note that we cannot follow the same line of logic for the case $1 \leq p \leq 2$ without a converse of the Young's inequality in Lemma 13.

□

C.3 Local uncertainty proofs

C.3.1 Proof of Lemma 4

Proof.

$$\begin{aligned}\langle \mathcal{T}_i^G g, \mathcal{T}_n^G h\rangle &= \langle \widehat{\mathcal{T}_i^G g}, \widehat{\mathcal{T}_n^G h}\rangle = \sum_{\ell=0}^{N-1} g(\lambda_\ell)\boldsymbol{u}_\ell[i]h^*(\lambda_\ell)\bar{u}_\ell[n] \\ &= \sum_{\ell=0}^{N-1} \big(g\cdot h\big)(\lambda_\ell)\boldsymbol{u}_\ell[i]\bar{u}_\ell[n] = \mathcal{T}_i^G(g\cdot h)[n].\end{aligned}$$

Moreover, a direct computation shows

$$\left(\sum_i \left|\langle \mathcal{T}_i^G g, \mathcal{T}_n^G h\rangle\right|^p\right)^{\frac{1}{p}} = \left(\sum_i \left|\sqrt{N}\mathcal{T}_n^G(g\cdot h)[i]\right|^p\right)^{\frac{1}{p}} = \left\|\mathcal{T}_b^G(g\cdot h)\right\|_p.$$

□

C.3.2 Proof of Theorem 11

Proof. For notational brevity in this proof, we omit the indices i_0, k_0 for the quantities $\tilde{i}$ and $\tilde{k}$. First, note that

$$\left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty = \max_k \left\|\mathcal{T}_{i_0}^G(g_{k_0}\cdot g_k)\right\|_\infty \leq \left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}}\right\|_2 \left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2,$$

where $\tilde{k}_{i_0,k_0} = \operatorname{argmax}_k \left\|\mathcal{T}_{i_0}^G(g_{k_0}\cdot g_k)\right\|_\infty$ and $\tilde{i}_{i_0,k_0} = \operatorname{argmin}_i \left|\mathcal{T}_{i_0}^G(g_{k_0}\cdot g_{\tilde{k}})[i]\right|$. Let us then interpolate the two following expressions:

$$\left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_2 \leq B^{\frac{1}{2}} \left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2 \tag{C.18}$$

$$\text{and} \quad \left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_\infty \leq \left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}}\right\|_2 \left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2. \tag{C.19}$$

We use the Riesz-Thorin Theorem (Theorem 24) with $p_1 = q_1 = p_2 = 2$, $q_2 = \infty$, $M_p = B^{\frac{1}{2}}$ and $M_q = \left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}}\right\|_2$. Note that $\boldsymbol{A}_g$ is a bounded operator from the Hilbert space spanned by $\mathcal{T}_{i_0}^G g_{k_0}$ (isomorphic to a one-dimensional Hilbert space) to the one spanned by $\{\mathcal{T}_{i_0}^G g_{k_0}\}_{i,k}$. We take $t = \frac{2}{r_2}$ and find $r_1 = 2$, leading to

$$\left\|\boldsymbol{A}_g \mathcal{T}_{i_0}^G g_{k_0}\right\|_{r_2} \leq B^{\frac{1}{r_2}} \left\|\mathcal{T}_{\tilde{i}}^G g_{\tilde{k}}\right\|_2^{1-\frac{2}{r_2}} \left\|\mathcal{T}_{i_0}^G g_{k_0}\right\|_2.$$

Since $\boldsymbol{A}_{\mathrm{g}}$ is a frame, we also have $\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2 \geq A^{\frac{1}{2}}\left\|\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2$, which yields:

$$\frac{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2}{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_p} \geq \frac{A^{\frac{1}{2}}}{B^{\frac{1}{p}}}\left\|\mathcal{T}_{\tilde{i}}^{G}g_{\tilde{k}}\right\|_2^{1-\frac{2}{p}}.$$

Finally, thanks to Hölder's inequality, we have for $p \leq 2$ and $\frac{1}{p}+\frac{1}{q}=1$

$$\begin{aligned}
\frac{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2}{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_p} &\leq \frac{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_q}{\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2} \\
&\leq \frac{B^{\frac{1}{q}}\left\|\mathcal{T}_{\tilde{i}}^{G}g_{\tilde{k}}\right\|_2^{1-\frac{2}{p}}}{A^{\frac{1}{2}}} \\
&\leq \frac{B^{1-\frac{1}{p}}\left\|\mathcal{T}_{\tilde{i}}^{G}g_{\tilde{k}}\right\|_2^{\frac{2}{p}-1}}{A^{\frac{1}{2}}} \\
&\leq \frac{B^{1-\frac{1}{p}}\left(\nu_{\tilde{i}}\left\|g_{\tilde{k}}\right\|_2\right)^{\frac{2}{p}-1}}{A^{\frac{1}{2}}}.
\end{aligned}$$

□

C.3.3 Proof of Corollary 2

Proof. The proof follows directly from the two following equalities. For the denominators, since the frame is tight, we have:

$$\left\|\boldsymbol{A}_{\mathrm{g}}\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2 = A^{\frac{1}{2}}\left\|\mathcal{T}_{i_0}^{G}g_{k_0}\right\|_2.$$

For the numerators, we have

$$
\begin{aligned}
\left\| \boldsymbol{A}_{\mathrm{g}} \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_{\infty} &= \max_{i,k} |\langle \mathcal{T}_i^{G} g_k, \mathcal{T}_{i_0}^{G} g_{k_0} \rangle| \\
&= \max_{i,k} |\mathcal{T}_{i_0}^{G} (g_k \cdot g_{k_0})[i]| && \text{(C.20)} \\
&= \max_{k} \left\| \mathcal{T}_{i_0}^{G} (g_k \cdot g_{k_0}) \right\|_{\infty} \\
&= \left\| \mathcal{T}_{i_0}^{G} g_{k_0}^2 \right\|_{\infty} && \text{(C.21)} \\
&= |\mathcal{T}_{i_0}^{G} g_{k_0}^2 (i_0)| && \text{(C.22)} \\
&= \langle \mathcal{T}_{i_0}^{G} g_{k_0}, \mathcal{T}_{i_0}^{G} g_{k_0} \rangle && \text{(C.23)} \\
&= \left\| \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_2^2,
\end{aligned}
$$

where (C.20) and (C.23) follow from (4.21), (C.21) follows from the second hypothesis, and (C.22) follows from the third hypothesis. □

C.3.4 Proof of Corollary 3

Proof. We have

$$
\left\| \boldsymbol{A}_{\mathrm{g}} \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_{\infty} = \max_{i,k} |\langle \mathcal{T}_i^{G} g_k, \mathcal{T}_{i_0}^{G} g_{k_0} \rangle| \geq \left| \langle \mathcal{T}_{i_0}^{G} g_{k_0}, \mathcal{T}_{i_0}^{G} g_{k_0} \rangle \right| = \left\| \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_2^2. \tag{C.24}
$$

Additionally, because $\{\mathcal{T}_i^{G} g_k\}_{i=1,2,\ldots,N;k=0,1,\ldots,M-1}$ is a frame, we have

$$
\left\| \boldsymbol{A}_{\mathrm{g}} \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_2 \leq B^{\frac{1}{2}} \left\| \mathcal{T}_{i_0}^{G} g_{k_0} \right\|_2. \tag{C.25}
$$

Combining (C.24) and (C.25) yields the desired inequality in (4.26). □

D Stationary signal processing on graphs

D.1 Convex models

Convex optimization has recently become a standard tool for problems such as de-noising, de-convolution or in-painting. Graph priors have been used in this field for more than a decade [122, 139, 92]. The general assumption is that the signal varies smoothly along the edges, which is equivalent to saying that the signal is low-frequency-based. Using this assumption, one way to express mathematically an in-painting problem is the following:

$$\dot{x} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} \qquad \text{s.t.} \qquad \left\| \boldsymbol{M}\boldsymbol{x} - \boldsymbol{y} \right\|_2 \leq \epsilon \tag{D.1}$$

where M is a masking operator and ϵ a constant computed thanks to the noise level. We could also rewrite the objective function as $\boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} + \gamma \left\| \boldsymbol{M}\boldsymbol{x} - \boldsymbol{y} \right\|_2^2$, but this implies a greedy search of the regularization parameter γ even when the noise level is known. For our simulations, we use Gaussian i.i.d. noise of standard deviation σ. It allows us to optimally set the regularization parameter $\epsilon = \sigma\sqrt{\#\boldsymbol{y}}$, where $\#\boldsymbol{y}$ is the number of elements of the measurement vector.

Graph de-convolution can also be addressed with the same prior assumption leading to

$$\dot{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} \qquad \text{s.t.} \qquad \left\| h(\boldsymbol{L})\boldsymbol{x} - \boldsymbol{y} \right\|_2 \leq \epsilon \tag{D.2}$$

where h is the convolution kernel. To be as generic as possible, we combine problems (D.1) and (D.2) together leading to a model capable of performing de-convolution, in-painting and de-noising at the same time:

$$\dot{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} \qquad \text{s.t.} \qquad \left\| \boldsymbol{M}h(\boldsymbol{L})\boldsymbol{x} - \boldsymbol{y} \right\|_2 \leq \epsilon. \tag{D.3}$$

When the signal is piecewise smooth on the graph, another regularization term can be used instead of $\boldsymbol{x}^* \boldsymbol{L}\boldsymbol{x} = \left\| \nabla_{\mathcal{G}} \boldsymbol{x} \right\|_2^2$, which is the ℓ_2-norm of the gradient on the graph. Using the ℓ_1-norm of the gradient favors a small number of major changes in signal and thus is better

for piecewise smooth signals. The resulting model is:

$$\dot{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \nabla_{\mathcal{G}} \boldsymbol{x} \right\|_1 \qquad \text{s.t.} \qquad \left\| \boldsymbol{M} h(\boldsymbol{L}) \boldsymbol{x} - \boldsymbol{y} \right\|_2 \leq \epsilon \tag{D.4}$$

In order to solve these problems, we use a subset of convex optimization tools called proximal splitting methods. Since we are not going to summarize them here, we encourage a novice reader to consult [25, 58] and the references therein for an introduction to the field.

D.2 Proof of Theorem 17

Proof. The proof is a classic development used in Bayesian machine learning. By assumption $\boldsymbol{x}$ is a sample of a Gaussian random multivariate signal $\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{\mu}, s^2(\boldsymbol{L})\right)$. The measurements are given by

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}_\sigma,$$

where $\boldsymbol{w}_\sigma \sim \mathcal{N}\left(\boldsymbol{0}, \sigma^2 \boldsymbol{I}\right)$ and thus have the following first and second moments: $\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{H}\boldsymbol{x}, \sigma^2 \boldsymbol{I}\right)$. For simplicity, we assume $s^2(\boldsymbol{L})$ to be invertible. However this assumption is not necessary. We can write the probabilities of $\boldsymbol{x}$ and $\boldsymbol{y}|\boldsymbol{x}$ as:

$$\mathbb{P}\left[\boldsymbol{x}\right] = \frac{1}{Z_{s^{-1}(\boldsymbol{L})}} e^{-\left\| s^{-1}(\boldsymbol{L})(\boldsymbol{x}-\boldsymbol{\mu}) \right\|_2^2} = \frac{1}{Z_{s^{-1}(\boldsymbol{L})}} e^{-\left\| s^{-1}(\boldsymbol{L})\tilde{\boldsymbol{x}} \right\|_2^2},$$

$$\mathbb{P}\left[\boldsymbol{y}|\boldsymbol{x}\right] = \frac{1}{Z_{\boldsymbol{I}}} e^{-\sigma^2 \left\| (\boldsymbol{H}\boldsymbol{x}-\boldsymbol{y}) \right\|_2^2}.$$

Using Bayes law we find

$$\mathbb{P}(\boldsymbol{x}|\boldsymbol{y}) = \frac{\mathbb{P}(\boldsymbol{y}|\boldsymbol{x})\mathbb{P}(\boldsymbol{x})}{\mathbb{P}(\boldsymbol{y})}.$$

The MAP estimator is

$$\begin{aligned}
\tilde{\boldsymbol{x}}|\boldsymbol{y} &= \underset{\boldsymbol{x}}{\operatorname{argmax}} \mathbb{P}(\boldsymbol{x}|\boldsymbol{y}) \\
&= \underset{\boldsymbol{x}}{\operatorname{argmax}} \log\left(\mathbb{P}(\boldsymbol{x}|\boldsymbol{y})\right) \\
&= \underset{\boldsymbol{x}}{\operatorname{argmin}} -\log\left(\mathbb{P}(\boldsymbol{y}|\boldsymbol{x})\right) - \log\left(\mathbb{P}(\boldsymbol{x})\right) + \log\left(\mathbb{P}(\boldsymbol{y})\right) \\
&= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| s^{-1}(\boldsymbol{L})\tilde{\boldsymbol{x}} \right\|_2^2 + \sigma^{-2} \left\| (\boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}) \right\|_2^2 \\
&= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| w(\boldsymbol{L})\tilde{\boldsymbol{x}} \right\|_2^2 + \left\| (\boldsymbol{H}\boldsymbol{x} - \boldsymbol{y}) \right\|_2^2,
\end{aligned}$$

where $w(\boldsymbol{L}) = \sigma s^{-1}(\boldsymbol{L})$. □

D.3 Proof of Theorem 19

The following is a generalization of the classical proof. For simplicity, we assume that $\overline{\boldsymbol{x}} = 0$, i.e $\tilde{\boldsymbol{x}} = \boldsymbol{x}$.

Proof. Because, by hypothesis $\boldsymbol{H} = h(\boldsymbol{L}) = \boldsymbol{U}h(\boldsymbol{\Lambda})\boldsymbol{U}^*$, we can rewrite the optimization problem (5.16) in the graph Fourier domain using the Parseval identity $\|\boldsymbol{x}\|_2 = \|\boldsymbol{U}\boldsymbol{x}\|_2 = \|\hat{\boldsymbol{x}}\|_2$:

$$\hat{\hat{\boldsymbol{x}}}|\hat{\boldsymbol{y}} = \underset{\hat{\boldsymbol{x}}}{\operatorname{argmin}} \|w(\boldsymbol{\Lambda})\hat{\boldsymbol{x}}\|_2^2 + \left\|h(\boldsymbol{\Lambda})\hat{\boldsymbol{x}} - \hat{\boldsymbol{y}}\right\|_2^2.$$

Since the matrix $h(\boldsymbol{\Lambda})$ is diagonal, the solution of this problem satisfies for all graph eigenvalue λ_ℓ

$$w^2(\lambda_\ell)\hat{\boldsymbol{x}}[\ell] + h^2(\lambda_\ell)\hat{\boldsymbol{x}}[\ell] - h(\lambda_\ell)\hat{\boldsymbol{y}}[\ell] = 0. \tag{D.5}$$

For simplicity, we drop the notation (λ_ℓ) and $[\ell]$. The previous equation is transformed in

$$\dot{\boldsymbol{x}} = \frac{h}{w^2 + h^2}\hat{\boldsymbol{y}}.$$

As a next step, we use the fact that $\hat{\boldsymbol{y}} = h\hat{\boldsymbol{x}} + \hat{\boldsymbol{w}}_\sigma$ to find:

$$\hat{\hat{\boldsymbol{x}}} = \frac{h^2\hat{\boldsymbol{x}} + h\hat{\boldsymbol{w}}_\sigma}{w^2 + h^2}.$$

The error performed by the algorithm becomes

$$\hat{\boldsymbol{e}} = \hat{\hat{\boldsymbol{x}}} - \hat{\boldsymbol{x}} = \frac{-w^2\hat{\boldsymbol{x}}}{w^2 + h^2} + \frac{h\hat{\boldsymbol{w}}_\sigma}{w^2 + h^2}.$$

The expectation of the error can thus be computed:

$$\begin{aligned}\mathbb{E}\left[\hat{\boldsymbol{e}}^2\right] &= \frac{w^4\mathbb{E}\left[\hat{\boldsymbol{x}}^2\right]}{\left(w^2 + h^2\right)^2} + \frac{h^2\mathbb{E}\left[\hat{\boldsymbol{w}}_\sigma^2\right]}{\left(w^2 + h^2\right)^2} - \frac{hw^2\mathbb{E}[\hat{\boldsymbol{x}}\hat{\boldsymbol{w}}_\sigma]}{\left(w^2 + h^2\right)^2} \\ &= \frac{w^4 s^2 + h^2\sigma^2}{\left(w^2 + h^2\right)^2},\end{aligned}$$

with s^2 the PSD of $\boldsymbol{x}$ and σ^2 the PSD of the noise $\boldsymbol{w}_\sigma$. Note that $\mathbb{E}[\hat{\boldsymbol{x}}\hat{\boldsymbol{w}}_\sigma] = 0$ because $\boldsymbol{x}$ and $\boldsymbol{w}$ are uncorrelated. Let us now substitute w^2 by z and minimize the expected error (for each λ_ℓ) with respect to z:

$$\begin{aligned}\frac{\partial}{\partial z}\mathbb{E}\left[\hat{\boldsymbol{e}}^2\right] &= \frac{\partial}{\partial z}\frac{z^2 s^2 + h^2\sigma^2}{\left(z + h^2\right)^2} \\ &= \frac{2zs^2\left(z + h^2\right) - 2\left(z^2 s^2 + h^2\sigma^2\right)}{\left(z + h^2\right)^3} = 0.\end{aligned}$$

From the numerator, we get:

$$2zs^2h^2 - 2h^2\sigma^2 = 0$$

The three possible solutions for z are $z_1 = \frac{\sigma^2}{s^2}$, $z_2 = \infty$ and $z_3 = -\infty$. z_3 is not possible because z is required to be positive. z_2 leads to $\dot{x} = 0$ which is optimal only if $s^2 = 0$. The optimal solution is therefore $z(\lambda_\ell) = \frac{\sigma^2(\lambda_\ell)}{s^2(\lambda_\ell)}$, resulting in

$$w(\lambda_\ell) = \sqrt{\frac{\sigma^2(\lambda_\ell)}{s^2(\lambda_\ell)}}.$$

This finishes the first part of the proof. To show that the solution to (5.16) is a Wiener filtering operation, we replace $w^2(\lambda_\ell)$ by $\frac{\sigma^2(\lambda_\ell)}{s^2(\lambda_\ell)}$ in (D.5) and find

$$\hat{\boldsymbol{x}}[\ell] = \frac{s^2(\lambda_\ell)h(\lambda_\ell)}{h^2(\lambda_\ell)s^2(\lambda_\ell) + \sigma^2(\lambda_\ell)}\hat{\boldsymbol{y}}[\ell],$$

which is the Wiener filter associated to the convolution $h(\boldsymbol{L}) = \boldsymbol{H}$. □

D.4 Proof of Theorem 18

Proof. Let $\boldsymbol{x}$ be GWSS with covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}} = s^2(\boldsymbol{L})$ and mean $\overline{\boldsymbol{x}} = \boldsymbol{\mu}$. The measurements satisfy

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}_\sigma,$$

where $\boldsymbol{w}_\sigma$ is i.i.d noise with PSD σ^2. The variable $\boldsymbol{y}$ has a covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{y}} = \boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^* + \sigma^2\boldsymbol{I}$ and a mean $\overline{\boldsymbol{y}} = \boldsymbol{H}\boldsymbol{\mu}$. The covariance between $\boldsymbol{x}$ and $\boldsymbol{y}$ is $\boldsymbol{\Sigma}_{\boldsymbol{xy}} = \boldsymbol{\Sigma}^*_{\boldsymbol{yx}} = s^2(\boldsymbol{L})\boldsymbol{H}^*$. For simplicity, we assume $s^2(\boldsymbol{L})$ and $\boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^* + \sigma^2\boldsymbol{I}$ to be invertible, however this assumption is not necessary. The Wiener optimization framework reads:

$$\dot{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\| \boldsymbol{H}\boldsymbol{x} - \boldsymbol{y} \right\|_2^2 + \sigma^2 \left\| s^{-1}(\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{\mu}) \right\|_2^2.$$

We perform the following change of variable $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{\mu}$, $\tilde{\boldsymbol{y}} = \boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu}$ and we obtain:

$$\tilde{\dot{\boldsymbol{x}}} = \underset{\tilde{\boldsymbol{x}}}{\operatorname{argmin}} \left\| H\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{y}} \right\|_2^2 + \sigma^2 \left\| s^{-1}(\boldsymbol{L})\tilde{\boldsymbol{x}}) \right\|_2^2.$$

The solution of the problem satisfies

$$\boldsymbol{H}^*\boldsymbol{H}\tilde{\boldsymbol{x}} - \boldsymbol{H}^*\tilde{\boldsymbol{y}} + \sigma^2 s^{-2}(\boldsymbol{L})\tilde{\dot{\boldsymbol{x}}} = 0.$$

From this equation we get $\tilde{\boldsymbol{x}}$ and transform it as:

$$\begin{aligned}
\tilde{\boldsymbol{x}} &= \left(\boldsymbol{H}^*\boldsymbol{H} + \sigma^2 s^{-2}(\boldsymbol{L})\right)^{-1}\boldsymbol{H}^*\tilde{\boldsymbol{y}} \\
&= s(\boldsymbol{L})\left(\sigma^2\boldsymbol{I} + s(\boldsymbol{L})\boldsymbol{H}^*\boldsymbol{H}s(\boldsymbol{L})\right)^{-1} s(\boldsymbol{L})\boldsymbol{H}^*\tilde{\boldsymbol{y}} \\
&= \left(\frac{1}{\sigma^2}s^2(\boldsymbol{L})\boldsymbol{H}^* - \frac{1}{\sigma^2}s^2(\boldsymbol{L})\boldsymbol{H}^*\left(\sigma^2\boldsymbol{I} + \boldsymbol{H}^*s^2(\boldsymbol{L})\boldsymbol{H}\right)^{-1}\boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^*\right)\tilde{\boldsymbol{y}} \qquad \text{(D.6)} \\
&= \frac{1}{\sigma^2}s^2(\boldsymbol{L})\boldsymbol{H}^*\left(\boldsymbol{I} - \left(\sigma^2\boldsymbol{I} + \boldsymbol{H}^*s^2(\boldsymbol{L})\boldsymbol{H}\right)^{-1}\boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^*\right)\tilde{\boldsymbol{y}} \\
&= s^2(\boldsymbol{L})\boldsymbol{H}^*\left(\sigma^2\boldsymbol{I} + \boldsymbol{H}^*s^2(\boldsymbol{L})\boldsymbol{H}\right)^{-1}\tilde{\boldsymbol{y}} \\
&= \boldsymbol{\Sigma}_{\boldsymbol{xy}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\tilde{\boldsymbol{y}}
\end{aligned}$$

where (D.6) follows from the Woodbury, Sherman and Morrison formula. The linear estimator of $\boldsymbol{x}$ corresponding to Wiener optimization is thus:

$$\begin{aligned}
\dot{\boldsymbol{x}} &= \tilde{\tilde{\boldsymbol{x}}} + \boldsymbol{\mu} \\
&= \boldsymbol{\Sigma}_{\boldsymbol{xy}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu}) + \boldsymbol{\mu} \\
&= \boldsymbol{\Sigma}_{\boldsymbol{xy}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{y} + \left(I - \boldsymbol{\Sigma}_{\boldsymbol{xy}}\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{H}\right)\boldsymbol{\mu} \\
&= \boldsymbol{Q}\boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{H})\,\boldsymbol{\mu}
\end{aligned}$$

We observe that it is equivalent to the solution of the linear minimum mean square error estimator:

$$\underset{\boldsymbol{Q},\boldsymbol{b}}{\operatorname{argmin}}\,\mathbb{E}\left[\left\|\boldsymbol{Q}\boldsymbol{y} + \boldsymbol{b} - \dot{\boldsymbol{x}}\right\|_2^2\right]$$

with $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}_\sigma$. See [56, Equation 12.6]. □

Using similar arguments, we can prove that

$$\dot{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\left\|s^{-1}(\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{\mu})\right\|_2^2 \quad \text{s.t. } \boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$$

leads to

$$\dot{\boldsymbol{x}} = s^2(\boldsymbol{L})\left(\boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^*\right)^{-1}\boldsymbol{y} + \left(\boldsymbol{I} - s^2(\boldsymbol{L})\left(\boldsymbol{H}s^2(\boldsymbol{L})\boldsymbol{H}^*\right)^{-1}\boldsymbol{H}\right)\boldsymbol{\mu}$$

and is thus a linear minimum mean square estimator too.

D.5 Development of equation 5.21

Proof. Let us denote the matrix of squared distances $\boldsymbol{D}_{\boldsymbol{x}}[i,j] = \frac{1}{K}\sum_k |\boldsymbol{x}_k[i] - \boldsymbol{x}_k[j]|^2$ for the samples $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_K\}$ of the random multivariate variable $\boldsymbol{x}$ on a N vertices graph. Let us

assume further that $\boldsymbol{\mu}[k] = \sum_{n=1}^{N} \boldsymbol{x}_k[n] = 0$. We show then that $\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}} = -\frac{1}{2}\boldsymbol{J}\boldsymbol{D}_{\boldsymbol{x}}\boldsymbol{J}$ where $\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}$ is the covariance (Gram) matrix defined as $\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,j] = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{x}_k[i]\boldsymbol{x}_k[j]$ and $\boldsymbol{J}$ is centering matrix $\boldsymbol{J}[k,l] = \boldsymbol{\delta}_k[l] - \frac{1}{N}$.

We have

$$(\boldsymbol{J}\boldsymbol{D}_{\boldsymbol{x}}\boldsymbol{J})[i,j] = \boldsymbol{D}_{\boldsymbol{x}}[i,j] + N^{-2}\sum_{k,l=1}^{N} \boldsymbol{D}_{\boldsymbol{x}}[k,l] - N^{-1}\sum_{k=1}^{N} (\boldsymbol{D}_{\boldsymbol{x}}[i,k] + \boldsymbol{D}_{\boldsymbol{x}}[k,j])$$

Let us substitute $\boldsymbol{D}_{\boldsymbol{x}}[i,j] = \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,i] + \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[j,j] - 2\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,j]$, then we find

$$\begin{aligned}
(\boldsymbol{J}\boldsymbol{D}_{\boldsymbol{x}}\boldsymbol{J})[i,j] &= \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,i] + \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[j,j] - 2\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,j] + N^{-2}\left(2N\sum_{n=1}^{N} \bar{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[n,n] - 2\boldsymbol{\mu}^*\boldsymbol{\mu}\right) \\
&- N^{-1}\left(N\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,i] + N\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[j,j] + 2\sum_{n=1}^{N} \dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[n,n] - 2\boldsymbol{\mu}^*(x[j] + x[i])\right) \\
&= -2\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}}[i,j] - 2N^{-2}\boldsymbol{\mu}^*\boldsymbol{\mu} + 2N^{-1}\boldsymbol{\mu}^*(\boldsymbol{x}[j] + \boldsymbol{x}[i]).
\end{aligned}$$

Under the assumption $\boldsymbol{\mu}[k] = \sum_{n=1}^{N} \boldsymbol{x}_k[n] = 0$, we recover the desired result $\dot{\boldsymbol{\Sigma}}_{\boldsymbol{x}} = -\frac{1}{2}\boldsymbol{J}\boldsymbol{D}_{\boldsymbol{x}}\boldsymbol{J}$. □

D.6 Proof of Theorem 15

Proof. The estimator is unbiased as

$$\begin{aligned}
\mathbb{E}\big[\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)\big] &= \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}\big[\boldsymbol{x}_m^*\boldsymbol{u}_\ell\boldsymbol{u}_\ell^*\boldsymbol{x}_m\big] \\
&= \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}\big[\hat{\boldsymbol{x}}_m^*[\ell]\hat{\boldsymbol{x}}_m[\ell]\big] = \gamma_{\boldsymbol{x}}(\lambda_\ell).
\end{aligned}$$

Furthermore, the bias is given by

$$\begin{aligned}
\mathbf{Var}\big[\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)\big] &= \mathbb{E}\Big[\big|\dot{\gamma}_{\boldsymbol{x}}(\lambda_\ell)\big|^2\Big] = \frac{1}{M}\mathbb{E}\Big[\big|\boldsymbol{x}^*\boldsymbol{u}_\ell\boldsymbol{u}_\ell^*\boldsymbol{x}\big|^2\Big] \\
&= \frac{1}{M}\mathbb{E}\Big[\big|\hat{\boldsymbol{x}}^*[\ell]\hat{\boldsymbol{x}}[\ell]\big|^2\Big] = \frac{1}{M}\mathbb{E}\big[|\hat{\boldsymbol{x}}[\ell]|^4\big] = \gamma_{\boldsymbol{x}}(\lambda_\ell)\frac{\hat{\boldsymbol{m}}_4[\ell]}{K}.
\end{aligned}$$

This concludes the proof. □

D.7 Proof of Theorem 16

Proof. Let us start with the bias. Using the Lipschitz property of $\gamma_{\boldsymbol{x}}(\lambda_\ell)$, we write

$$\begin{aligned}\left|\mathbb{E}\left[\ddot{\gamma}_{\boldsymbol{x}}(k\tau)-\gamma_{\boldsymbol{x}}(k\tau)\right]\right| &= \left|\sum_{\ell=0}^{N-1} g^2(\lambda_\ell - k\tau)\frac{\gamma_{\boldsymbol{x}}(k\tau)}{\left\|g(\boldsymbol{\lambda}-k\tau\mathbf{1})\right\|_2^2}-\gamma_{\boldsymbol{x}}(k\tau)\right| \\ &\leq \left|A\gamma_{\boldsymbol{x}}(k\tau)\right| + \frac{\epsilon}{\left\|g(\boldsymbol{\lambda}-k\tau\mathbf{1})\right\|_2^2}\sum_{\ell=1}^{N} g(\lambda_\ell-k\tau)^2\left|\lambda_\ell-k\tau\right|\end{aligned}$$

where by definition $A=\sum_{k=1}^{K}\frac{g^2(\lambda_\ell-k\tau)}{\|g(\boldsymbol{\lambda}-k\tau\mathbf{1})\|_2^2}-1=0$, and the claim follows.

We now need to prove the second part of the theorem concerning the variance. Set $\boldsymbol{w}_{n,\tau}=\frac{g(\vartheta-\vartheta_{n,\tau})^2}{c_g(\vartheta)}$. The centered random variable

$$\ddot{\gamma}_{\boldsymbol{x}}(\lambda)-\gamma_{\boldsymbol{x}}(\lambda)=\sum_{n,\tau}^{N,T} w_{n,\tau}(\dot{h}(\vartheta_{n,\tau})-h(\vartheta_{n,\tau}))$$

is a weighted sum of independent random variables $\dot{h}(\vartheta_{n,\tau})-h(\vartheta_{n,\tau})$, which according to Theorem 15 have variance $h^2(\theta_{n,\tau})\frac{\mathbb{E}[\hat{\varepsilon}_{n,\tau}^4]-1}{K}$. It follows that,

$$\begin{aligned}\mathbf{Var}\left[\ddot{h}(\vartheta)\right] &= \sum_{n,\tau}^{N,T} w_{n,\tau}^2\mathbb{E}\left[(\dot{h}(\vartheta_{n,\tau})-h(\vartheta_{n,\tau}))^2\right] \\ &= \sum_{n,\tau}^{N,T} w_{n,\tau}^2 h^2(\theta_{n,\tau})\frac{\mathbb{E}\left[\hat{\varepsilon}_{n,\tau}^4\right]-1}{K},\end{aligned}$$

which matches our claim. □

E Manifold regularization via graph total variation

Lemma 14. *Given an open set $\mathcal{S}$ and $\int_{\boldsymbol{x}\in\mathcal{S}} ||A(\boldsymbol{x})| - |B(\boldsymbol{x})|| \, d\mu_{\mathcal{S}}(\boldsymbol{x}) = 0$, we have*

$$\int_{\boldsymbol{x}\in\mathcal{S}} |A(\boldsymbol{x})| d\mu_{\mathcal{S}}(\boldsymbol{x}) = \int_{\boldsymbol{x}\in\mathcal{S}} |B(\boldsymbol{x})| d\mu_{\mathcal{S}}(\boldsymbol{x})$$

Proof. A simple observation shows

$$\left| \int_{\boldsymbol{x}\in\mathcal{S}} |A(\boldsymbol{x})| \mathrm{d}\mu_{\mathcal{S}}(\boldsymbol{x}) - \int_{\mathcal{M}} |B(\boldsymbol{x})| \mathrm{d}\mu_{\mathcal{S}}(\boldsymbol{x}) \right| \leq \int_{\boldsymbol{x}\in\mathcal{S}} ||A(\boldsymbol{x})| - |B(\boldsymbol{x})|| \, \mathrm{d}\mu_{\mathcal{S}}(\boldsymbol{x},) = 0 which implies the desired result.$$

□

Lemma 15. *Given $\boldsymbol{x} \in \mathbb{R}^k$, we have*

$$\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{x}\|^2}{t}} \|\boldsymbol{x}\|_2^2 \, d\boldsymbol{x} = \frac{k}{2} t^{\frac{k+2}{2}} \pi^{\frac{k}{2}}. \tag{E.1}$$

Proof. Let us use an induction. First For $k = 1$, we integrate by parts to obtain the result.

$$\int_{x\in\mathbb{R}} e^{\frac{-x^2}{t}} x^2 \mathrm{d}x = \int_{x\in\mathbb{R}} \left(x e^{\frac{-x^2}{t}} \right) x \mathrm{d}x = \int_{x\in\mathbb{R}} \frac{t}{2} e^{\frac{-\|x\|^2}{t}} \mathrm{d}x = \frac{t}{2} (\pi t)^{\frac{1}{2}} = \frac{t^{\frac{3}{2}} \pi^{\frac{1}{2}}}{2}$$

Then, supposing that the relation is true for $\boldsymbol{x} \in \mathbb{R}^k$, we need to prove it for $\boldsymbol{x} \in \mathbb{R}^{k+1}$. For

convenience the i element of the vector $\boldsymbol{x}$ is written x_i. We have

$$\begin{aligned}
\int_{\boldsymbol{x}\in\mathbb{R}^{k+1}} e^{-\frac{\|\boldsymbol{x}\|^2}{t}} \|\boldsymbol{x}\|_2^2 \mathrm{d}\boldsymbol{x} &= \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{-\frac{x_1^2+\cdots+x_k^2}{t}} \left(x_1^2+\cdots+x_k^2\right) \mathrm{d}x_1 \ldots \mathrm{d}x_k \int_{x_{k+1}\in\mathbb{R}} e^{-\frac{x_{k+1}^2}{t}} \mathrm{d}x_{k+1} \\
&+ \int_{\boldsymbol{x}\in\mathbb{R}^{k+1}} e^{-\frac{x_1^2+\cdots+x_k^2+x_{k+1}^2}{t}} x_{k+1}^2 \mathrm{d}x_1 \ldots \mathrm{d}x_k \mathrm{d}x_{k+1} \\
&= \int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|x\|^2}{t}} \|x\|_2^2 \mathrm{d}x \int_{x\in\mathbb{R}} e^{-\frac{x_{k+1}^2}{t}} \mathrm{d}x_{k+1} \\
&+ \left(\int_{x\in\mathbb{R}} e^{\frac{-x_{k+1}^2}{t}} x_{k+1}^2 \mathrm{d}x_{k+1}\right)\left(\int_{x\in\mathbb{R}} e^{-\frac{x^2}{t}} x^2 \mathrm{d}x\right)^k \\
&= \frac{k}{2} t^{\frac{k+2}{2}} \pi^{\frac{k}{2}} \frac{(t\pi)^{\frac{1}{2}}}{2} + \frac{t^{\frac{3}{2}}\pi^{\frac{1}{2}}}{2}\left((t\pi)^{\frac{1}{2}}\right)^k = \frac{k+1}{2} t^{\frac{k+3}{2}} \pi^{\frac{k+1}{2}},
\end{aligned}$$

which concludes the induction and hence the proof. □

Lemma 16. *Given $\boldsymbol{x}\in\mathbb{R}^k$, we have*

$$\int_{\mathbb{R}^k} e^{\frac{-\|x\|^2}{t}} \|\boldsymbol{x}\|_2^4 \, d\boldsymbol{x} = \frac{k^2+2k}{4} t^{\frac{k+4}{2}} \pi^{\frac{k}{2}} \tag{E.2}$$

Proof. Let us use an induction. First For $k=1$, we integrate by parts to obtain the result.

$$\begin{aligned}
\int_{x\in\mathbb{R}} e^{\frac{-x^2}{t}} x^4 \mathrm{d}x &= \int_{x\in\mathbb{R}} \left(x e^{\frac{-x^2}{t}}\right) x^3 \mathrm{d}x \\
&= -\frac{t}{2} e^{\frac{x^2}{t}} x^3 \Big|_{-\infty}^{\infty} + \frac{3t}{2} \int_{x\in\mathbb{R}} x^2 e^{\frac{-x^2}{t}} \mathrm{d}x \\
&= \frac{3t}{2} \frac{t^{\frac{3}{2}}\pi^{\frac{1}{2}}}{2} = \frac{3}{4} t^{\frac{5}{2}} \pi^{\frac{1}{2}},
\end{aligned}$$

where we take advantage of Lemma 15. Then, supposing that the relation is true for $\boldsymbol{x}\in\mathbb{R}^k$, we need to prove it for $\boldsymbol{x}\in\mathbb{R}^{k+1}$. For convenience the i element of the vector $\boldsymbol{x}$ is written x_i. We

have

$$
\begin{aligned}
&\int_{\boldsymbol{x}\in\mathbb{R}^{k+1}} e^{-\frac{\|\boldsymbol{x}\|^2}{t}} \|\boldsymbol{x}\|_2^4 \,\mathrm{d}\boldsymbol{x} \\
= &\int_{\boldsymbol{x}\in\mathbb{R}^{k+1}} e^{-\frac{x_1^2+\cdots+x_k^2+x_{k+1}^2}{t}} \left(x_1^2+\cdots+x_k^2+x_{k+1}^2\right)^2 \mathrm{d}x_1\ldots\mathrm{d}x_k\mathrm{d}x_{k+1} \\
= &\int_{\boldsymbol{x}\in\mathbb{R}^{k+1}} e^{-\frac{x_1^2+\cdots+x_k^2}{t}} e^{-\frac{x_{k+1}^2}{t}} \left[\left(x_1^2+\cdots+x_k^2\right)^2 + 2x_{k+1}^2\left(x_1^2+\cdots+x_k^2\right) + x_{k+1}^4\right] \mathrm{d}x_1\ldots\mathrm{d}x_k\mathrm{d}x_{k+1} \\
= &\int_{\boldsymbol{x}\in\mathbb{R}^{k}} e^{-\frac{x_1^2+\cdots+x_k^2}{t}} \left(x_1^2+\cdots+x_k^2\right)^2 \mathrm{d}x_1\ldots\mathrm{d}x_k \int_{x_{k+1}\in\mathbb{R}} e^{-\frac{x_{k+1}^2}{t}} \mathrm{d}x_{k+1} \\
+ &\int_{\boldsymbol{x}\in\mathbb{R}^{k}} e^{-\frac{x_1^2+\cdots+x_k^2}{t}} \mathrm{d}x_1\ldots\mathrm{d}x_k \int_{x_{k+1}\in\mathbb{R}} x_{k+1}^4 e^{-\frac{x_{k+1}^2}{t}} \mathrm{d}x_{k+1} \\
+ &\;2\int_{\boldsymbol{x}\in\mathbb{R}^{k}} \left(x_1^2+\cdots+x_k^2\right) e^{-\frac{x_1^2+\cdots+x_k^2}{t}} \mathrm{d}x_1\ldots\mathrm{d}x_k \int_{x_{k+1}\in\mathbb{R}} x_{k+1}^2 e^{-\frac{x_{k+1}^2}{t}} \mathrm{d}x_{k+1} \\
= &\frac{k^2+2k}{4}(\pi t)^{\frac{1}{2}} t^{\frac{k+4}{2}} \pi^{\frac{k}{2}} + (\pi t)^{\frac{k}{2}} \frac{3}{4} t^{\frac{5}{2}} \pi^{\frac{1}{2}} + 2\frac{k}{2} t^{\frac{k+2}{2}} \pi^{\frac{k}{2}} \frac{t^{\frac{3}{2}}\pi^{\frac{1}{2}}}{2} \\
= &\frac{k^2+4k+3}{4} t^{\frac{k+5}{2}} \pi^{\frac{k+1}{2}} = \frac{(k+1)^2+2(k+2)}{4} t^{\frac{(k+1)+4}{2}} \pi^{\frac{k+1}{2}},
\end{aligned}
$$

which concludes the induction and hence the proof. □

Lemma 17. *Given $\boldsymbol{p}\in\mathbb{R}^k$, we have :*

$$
\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{x}\|^2}{t}} |\langle \boldsymbol{p},\boldsymbol{x}\rangle| d\boldsymbol{x} = t^{\frac{k+1}{2}} \pi^{\frac{k-1}{2}} \left\|\boldsymbol{p}\right\|_2 \tag{E.3}
$$

Proof. The function $e^{\frac{-\|\boldsymbol{x}\|^2}{t}}$ is isotropic as it does not depend on the direction of $\boldsymbol{x}$, but only it's norm. To solve the integral, we make the following change of coordinates $\boldsymbol{y} = \boldsymbol{R}\boldsymbol{x}$ with $\boldsymbol{R}$ a unitary rotation matrix such that $\langle \boldsymbol{p},\boldsymbol{x}\rangle = \left\|\boldsymbol{p}\right\|_2 \langle \boldsymbol{\delta}_1,\boldsymbol{y}\rangle$. The integration leads to the result

$$
\begin{aligned}
\int_{\boldsymbol{x}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{x}\|^2}{t}} |\langle \boldsymbol{p},\boldsymbol{x}\rangle| \mathrm{d}\boldsymbol{x} &= \|\boldsymbol{b}\|_2 \int_{\boldsymbol{y}\in\mathbb{R}^k} e^{\frac{-\|\boldsymbol{y}\|^2}{t}} \langle \boldsymbol{\delta}_1,\boldsymbol{y}\rangle \mathrm{d}\boldsymbol{y} \\
&= \|\boldsymbol{b}\|_2 \int_{\mathbb{R}^{k-1}} e^{\frac{-y_2^2-y_3^2-\cdots-y_k^2}{t}} \mathrm{d}y_2\mathrm{d}y_3\ldots\mathrm{d}y_k \int_{\mathbb{R}} e^{\frac{-y_1^2}{t}} |y_1| \mathrm{d}y_1 \\
&= \|\boldsymbol{b}\|_2 \left(\int_{\mathbb{R}} e^{\frac{-x_2^2}{t}} \mathrm{d}x_2\right)^{k-1} \int_0^\infty e^{\frac{-x_1^2}{t}} 2x_1 \mathrm{d}x_1 \\
&= \|\boldsymbol{b}\|_2 \left(\sqrt{t\pi}\right)^{k-1} t = t^{\frac{k+1}{2}} \pi^{\frac{k-1}{2}} \|\boldsymbol{b}\|_2 .
\end{aligned}
$$

□

List of publications

During my thesis I authored or co-authored the following papers and technical reports:

Graph signal processing

Journals

1. N. Perraudin and P. Vandergheynst. Stationary signal processing on graphs. *IEEE Transactions on Signal Processing*, 2017

2. N. Perraudin, B. Ricaud, D. Shuman, and P. Vandergheynst. Global and local uncertainty principles for signals on graphs. *unpublished, arXiv:1603.03030*, 2016

3. N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst. Fast robust pca on graphs. *Journal of Selected Topics in Signal Processing*, 10(4):740–756, 2016

4. N. Shahid, N. Perraudin, G. Puy, and P. Vandergheynst. Compressive pca for low-rank matrices on graphs. *Transactions on Signal and Information Processing over Networks*, 2016

5. A. Loukas and N. Perraudin. Stationary time-vertex signal processing. *unpublished, arXiv:1611.00255*, 2016

Conference papers

6. N. Shahid, N. Perraudin, V. Kalofolias, B. Ricaud, and P. Vandergheynst. Pca using graph total variation. In *2016 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4668–4672. IEEE, 2016

7. F. Grassi, N. Perraudin, and B. Ricaud. Tracking time-vertex propagation using dynamic graph wavelets. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017

8. N. Perraudin, A. Loukas, F. Grassi, and P. Vandergheynst. Towards stationary time-vertex signal processing. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017

Technical reports

9. A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst. Accelerated filtering on graphs using lanczos method. *unpublished, arXiv:1509.04537*, 2015

10. N. Shahid, N. Perraudin, G. Puy, and P. Vandergheynst. Low-rank matrices on graphs: Generalized recovery and applications. *unpublished, arXiv:1605.05579*, 2016

11. A. Loukas and N. Perraudin. Predicting the evolution of stationary graph signals. *unpublished, arXiv:1607.03313*, 2016

Audio signal procssing

Journals

12. N. Perraudin, N. Holighaus, P. Majdak, and P. Balazs. Audio inpainting with similarity graphs. *unpublished, arXiv:1607.06667*, 2016

13. N. Perraudin, N. Holighaus, P. L. Søndergaard, and P. Balazs. Designing gabor windows using convex optimization. *unpublished, arXiv:1401.6033*, 2014

Conference papers

14. N. Perraudin, N. Holighaus, P. Soendergaard, and P. Balazs. Gabor dual windows using convex optimization. In *Proceeedings of the 10th International Conference on Sampling theory and Applications (SAMPTA 2013)*, 2013

15. N. Perraudin, P. Balazs, and P. L. Sondergaard. A fast griffin-lim algorithm. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), Workshop on*, pages 1–4. IEEE, 2013

Toolboxes

16. N. Perraudin, D. Shuman, G. Puy, and P. Vandergheynst. Unlocbox a matlab convex optimization toolbox using proximal splitting methods. *unpublished, arXiv:1402.0779*, 2014

17. N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond. Gspbox: A toolbox for signal processing on graphs. *unpublished, arXiv:1408.5781*, 2014

[1] A. Agaskar and Y. M. Lu. An uncertainty principle for functions defined on graphs. In *SPIE Optical Engineering+ Applications*, pages 81380T–81380T. International Society for Optics and Photonics, 2011.

[2] A. Agaskar and Y. M. Lu. Uncertainty principles for signals defined on graphs: Bounds and characterizations. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3493–3496. IEEE, 2012.

[3] A. Agaskar and Y. M. Lu. A spectral graph uncertainty principle. *Transactions on Information Theory*, 59(7):4338–4356, 2013.

[4] L. Babai, D. Y. Grigoryev, and D. M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 310–324. ACM, 1982.

[5] M. S. Bartlett. Periodogram analysis and continuous spectra. *Biometrika*, pages 1–16, 1950.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[7] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine learning*, 56(1-3):209–239, 2004.

[8] M. Belkin and P. Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. In *International Conference on Computational Learning Theory*, pages 486–500. Springer, 2005.

[9] M. Belkin and P. Niyogi. Convergence of laplacian eigenmaps. In *NIPS*, pages 129–136, 2006.

[10] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. *Mij*, 1:2, 2003.

[11] M. Benzi. Localization in matrix computations: Theory and applications. 2016.

[12] M. Benzi and N. Razouk. Decay bounds and O(n) algorithms for approximating functions of sparse matrices. *Electron. Trans. Numer. Anal.*, 28:16–39, 2007.

[13] M. Benzi and V. Simoncini. Decay bounds for functions of hermitian matrices with banded or kronecker structure. *SIAM Journal on Matrix Analysis and Applications*, 36 (3):1263–1282, 2015.

[14] S. Brooks and E. Lindenstrauss. Non-localization of eigenfunctions on large regular graphs. *Israel Journal of Mathematics*, 193(1):1–14, 2013.

[15] P. J. Cameron et al. Automorphisms of graphs. *Topics in algebraic graph theory*, 102: 137–155, 2004.

[16] E. J. Candes and J. Romberg. Quantitative robust uncertainty principles and optimally sparse decompositions. *Foundations of Computational Mathematics*, 6(2):227–254, 2006.

[17] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical imaging and vision*, 20(1-2):89–97, 2004.

[18] A. Chan and C. D. Godsil. Symmetry and eigenvectors. In *Graph symmetry*, pages 75–106. Springer, 1997.

[19] S. P. Chepuri and G. Leus. Subsampling for graph power spectrum estimation. In *Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pages 1–5. IEEE, 2016.

[20] O. Christensen. *Frames and Bases*. Birkhäuser, 2008.

[21] F. Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005.

[22] F. Chung. The diameter and laplacian eigenvalues of directed graphs. *Electronic Journal of Combinatorics*, 13(4), 2006.

[23] F. R. Chung. *Spectral graph theory*, volume 92. AMS Bookstore, 1997.

[24] R. R. Coifman and M. Maggioni. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, 21(1):53–94, 2006.

[25] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

[26] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[27] J. K. Cullum and R. A. Willoughby. *Lanczos algorithms for large symmetric eigenvalue computations. Vol. 1*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002.

[28] D. M. Cvetković, M. Doob, and H. Sachs. *Spectra of graphs: theory and application*, volume 87. Academic Press, 1980.

[29] Y. Dekel, J. R. Lee, and N. Linial. Eigenvectors of random graphs: Nodal domains. *Random Structures & Algorithms*, 39(1):39–58, 2011.

[30] D. L. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *Transactions on Information Theory*, 47(7):2845–2862, 2001.

[31] D. L. Donoho and P. B. Stark. Uncertainty principles and signal recovery. *SIAM Journal on Applied Mathematics*, 49(3):906–931, 1989.

[32] E. R. Dougherty. *Random processes for image and signal processing*. SPIE Optical Engineering Press, 1999.

[33] I. Dumitriu, S. Pal, et al. Sparse regular random graphs: spectral density and eigenvectors. *The Annals of Probability*, 40(5):2197–2235, 2012.

[34] M. Elad and A. M. Bruckstein. A generalized uncertainty principle and sparse representation in pairs of bases. *Transactions on Information Theory*, 48(9):2558–2567, 2002.

[35] A. Elmoataz, O. Lezoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing. *Transactions on Image Processing*, 17(7):1047–1060, 2008.

[36] L. Erdős, A. Knowles, H.-T. Yau, and J. Yin. Spectral statistics of erdős-rényi graphs ii: Eigenvalue spacing and the extreme eigenvalues. *Communications in Mathematical Physics*, 314(3):587–640, 2012.

[37] L. Erdős, A. Knowles, H.-T. Yau, J. Yin, et al. Spectral statistics of erdős–rényi graphs i: Local semicircle law. *The Annals of Probability*, 41(3B):2279–2375, 2013.

[38] P. Erdös and A. Rényi. On random graphs, i. *Publicationes Mathematicae (Debrecen)*, 6: 290–297, 1959.

[39] H. G. Feichtinger, D. Onchis-Moaca, B. Ricaud, B. Torrésani, and C. Wiesmeyr. A method for optimizing the ambiguity function concentration. In *Signal Processing Conference (EUSIPCO), Proceedings of the 20th European*, pages 804–808. IEEE, 2012.

[40] G. B. Folland and A. Sitaram. The uncertainty principle: a mathematical survey. *Journal of Fourier analysis and applications*, 3(3):207–238, 1997.

[41] A. Frommer, S. Güttel, and M. Schweitzer. Efficient and stable arnoldi restarts for matrix functions based on quadrature. *SIAM Journal on Matrix Analysis and Applications*, 35 (2):661–683, 2014.

[42] A. Gadde and A. Ortega. A probabilistic interpretation of sampling theory of graph signals. In *Acoustics, Speech and Signal Processing (ICASSP), International Conference on*, pages 3257–3261. IEEE, 2015.

[43] E. Gallopoulos and Y. Saad. Efficient solution of parabolic equations by krylov approximation methods. *SIAM Journal on Scientific and Statistical Computing*, 13(5):1236–1264, 1992.

[44] B. Girault. *Signal Processing on Graphs-Contributions to an Emerging Field*. PhD thesis, Ecole normale supérieure de lyon, 2015.

[45] B. Girault. Stationary graph signals using an isometric graph translation. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pages 1516–1520. IEEE, 2015.

[46] B. Girault, P. Goncalves, E. Fleury, and A. S. Mor. Semi-supervised learning for graph to signal mapping: A graph signal wiener filter interpretation. In *Acoustics, Speech and Signal Processing (ICASSP), International Conference on*, pages 1115–1119. IEEE, 2014.

[47] B. Girault, P. Gonçalves, and E. Fleury. Translation on graphs: an isometric shift operator. *Signal Processing Letters*, 22(12):2416–2420, 2015.

[48] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins University Press, Baltimore, MD, Fourth edition, 2013.

[49] L. J. Grady and J. Polimeni. *Discrete calculus: Applied analysis on graphs for computational science*. Springer, 2010.

[50] F. Grassi, N. Perraudin, and B. Ricaud. Tracking time-vertex propagation using dynamic graph wavelets. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017.

[51] R. Gribonval and M. Nielsen. Sparse representations in unions of bases. *Transactions on Information theory*, 49(12):3320–3325, 2003.

[52] D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[53] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):328–340, 2005.

[54] V. Kalofolias. How to learn a graph from smooth signals. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 920–929, Cadiz, 09–11 May 2016. PMLR.

[55] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Matrix completion on graphs. *unpublished, arXiv:1408.1717*, 2014.

[56] S. M. Kay. *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.

[57] D. J. Klein and M. Randić. Resistance distance. *Journal of Mathematical Chemistry*, 12 (1):81–95, 1993.

[58] N. Komodakis and J.-C. Pesquet. Playing with duality: An overview of recent primal? dual approaches for solving large-scale optimization problems. *IEEE Signal Processing Magazine*, 32(6):31–54, 2015.

[59] J. Kovačević and A. Chebira. Life beyond bases: The advent of frames (part I). *Signal Processing Magazine*, 24(4):86–104, Jul. 2007.

[60] J. Kovačević and A. Chebira. Life beyond bases: The advent of frames (part II). *Signal Processing Magazine*, 24(5):115–125, Sep. 2007.

[61] S. Lafon and A. B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *Transactions on pattern analysis and machine intelligence*, 28(9):1393–1403, 2006.

[62] N. Leonardi and D. Van De Ville. Wavelet frames on graphs defined by fmri functional connectivity. In *International Symposium on Biomedical Imaging: From Nano to Macro*, pages 2136–2139. IEEE, 2011.

[63] N. Leonardi and D. Van De Ville. Tight wavelet frames on multislice graphs. *Transactions on Signal Processing*, 61(13):3357–3367, 2013.

[64] E. H. Lieb. Integral bounds for radar ambiguity functions and wigner distributions. *Journal of mathematical physics*, 31(3):594–599, 1990.

[65] A. Loukas and N. Perraudin. Predicting the evolution of stationary graph signals. *unpublished, arXiv:1607.03313*, 2016.

[66] A. Loukas and N. Perraudin. Stationary time-vertex signal processing. *unpublished, arXiv:1611.00255*, 2016.

[67] H. Maassen and J. B. Uffink. Generalized entropic uncertainty relations. *Physical Review Letters*, 60(12):1103, 1988.

[68] S. G. Mallat. *A Wavelet Tour of Signal Processing, 3rd ed.* Academic Press, 2008.

[69] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro. Stationary graph processes and spectral estimation. *unpublished, arXiv:1603.04667*, 2016.

[70] P. N. McGraw and M. Menzinger. Laplacian spectra as a diagnostic tool for network structure and dynamics. *Physical Review E*, 77(3):031102, 2008.

[71] B. D. McKay. The expected eigenvalue distribution of a large regular graph. *Linear Algebra and its Applications*, 40:203–216, 1981.

[72] J. Mei and J. M. Moura. Signal processing on graphs: Causal modeling of unstructured data. *Transactions on Signal Processing*, 2016.

[73] R. Merris. Laplacian matrices of graphs: a survey. *Linear algebra and its applications*, 197:143–176, 1994.

[74] R. Merris. Laplacian graph eigenvectors. *Linear algebra and its applications*, 278(1): 221–236, 1998.

[75] B. Metzger and P. Stollmann. Heat kernel estimates on weighted graphs. *Bulletin of the London Mathematical Society*, 32(04):477–483, 2000.

[76] Y. Nakatsukasa, N. Saito, and E. Woei. Mysteries around the graph laplacian eigenvalue 4. *Linear Algebra and its Applications*, 438(8):3231–3246, 2013.

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

[78] A. Papoulis and S. U. Pillai. *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, 2002.

[79] J. Paratte, N. Perraudin, and P. Vandergheynst. Compressive embedding and visualization using graphs. *unpublished, arXiv:1702.05815*, 2017.

[80] B. Pasdeloup, R. Alami, V. Gripon, and M. Rabbat. Toward an uncertainty principle for weighted graphs. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pages 1496–1500. IEEE, 2015.

[81] V. I. Paulsen and M. Raghupathi. *An introduction to the theory of reproducing kernel Hilbert spaces*, volume 152. Cambridge University Press, 2016.

[82] N. Perraudin and P. Vandergheynst. Stationary signal processing on graphs. *IEEE Transactions on Signal Processing*, 2017.

[83] N. Perraudin, P. Balazs, and P. L. Sondergaard. A fast griffin-lim algorithm. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), Workshop on*, pages 1–4. IEEE, 2013.

[84] N. Perraudin, N. Holighaus, P. Soendergaard, and P. Balazs. Gabor dual windows using convex optimization. In *Proceeedings of the 10th International Conference on Sampling theory and Applications (SAMPTA 2013)*, 2013.

[85] N. Perraudin, N. Holighaus, P. L. Søndergaard, and P. Balazs. Designing gabor windows using convex optimization. *unpublished, arXiv:1401.6033*, 2014.

[86] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond. Gspbox: A toolbox for signal processing on graphs. *unpublished, arXiv:1408.5781*, 2014.

[87] N. Perraudin, D. Shuman, G. Puy, and P. Vandergheynst. Unlocbox a matlab convex optimization toolbox using proximal splitting methods. *unpublished, arXiv:1402.0779*, 2014.

[88] N. Perraudin, N. Holighaus, P. Majdak, and P. Balazs. Audio inpainting with similarity graphs. *unpublished, arXiv:1607.06667*, 2016.

[89] N. Perraudin, B. Ricaud, D. Shuman, and P. Vandergheynst. Global and local uncertainty principles for signals on graphs. *unpublished, arXiv:1603.03030*, 2016.

[90] N. Perraudin, A. Loukas, F. Grassi, and P. Vandergheynst. Towards stationary time-vertex signal processing. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017.

[91] I. Pesenson. Variational splines and paley–wiener spaces on combinatorial graphs. *Constructive Approximation*, 29(1):1–21, 2009.

[92] G. Peyré, S. Bougleux, and L. Cohen. Non-local regularization of inverse problems. In *Computer Vision–ECCV 2008*, pages 57–68. Springer, 2008.

[93] G. M. Phillips. *Interpolation and approximation by polynomials*. Springer, New York, 2003.

[94] M. A. Pinsky. *Introduction to Fourier Analysis and Wavelets*. Vol. 102 of the Graduate Studies in Mathematics, American Mathematical Society, 2002.

[95] M. Puschel and J. M. Moura. Algebraic signal processing theory: Foundation and 1-d time. *Transactions on Signal Processing*, 56(8):3572–3585, 2008.

[96] G. Puy, N. Tremblay, R. Gribonval, and P. Vandergheynst. Random sampling of bandlimited signals on graphs. *Applied and Computational Harmonic Analysis*, 2016.

[97] C. E. Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

[98] M. Reed and B. Simon. *Methods of Modern Mathematical Physics, Vol. 2.: Fourier Analysis, Self-Adjointness*. Academic Press, 1975.

[99] A. Rényi et al. On measures of entropy and information. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 547–561, 1961.

[100] B. Ricaud and B. Torrésani. Refined support and entropic uncertainty inequalities. *Transactions on Information Theory*, 59(7):4272–4279, 2013.

[101] B. Ricaud and B. Torrésani. A survey of uncertainty principles and some signal processing applications. *Advances in Computational Mathematics*, 40(3):629–650, 2014.

[102] B. Ricaud, G. Stempfel, B. Torrésani, C. Wiesmeyr, H. Lachambre, and D. Onchis. An optimally concentrated gabor transform for localized time-frequency components. *Advances in Computational Mathematics*, 40(3):683–702, 2014.

[103] D. Romero, M. Ma, and G. B. Giannakis. Kernel-based reconstruction of graph signals. *Transactions on Signal Processing*, 65(3):764–778, 2017.

[104] N. L. Roux, Y. Bengio, P. Lamblin, M. Joliveau, and B. Kégl. Learning the 2-d topology of images. In *Advances in Neural Information Processing Systems*, pages 841–848, 2008.

[105] Y. Rubner and C. Tomasi. The earth mover's distance. In *Perceptual Metrics for Image Database Navigation*, pages 13–28. Springer, 2001.

[106] N. Saito and E. Woei. On the phase transition phenomenon of graph laplacian eigenfunctions on trees. *RIMS Kokyuroku*, 1743:77–90, 2011.

[107] A. Sakiyama and Y. Tanaka. Oversampled graph laplacian matrix for graph filter banks. *Transactions on Signal Processing*, 62(24):6425–6437, 2014.

[108] A. Sandryhaila and J. M. Moura. Discrete signal processing on graphs. *Transactions on signal processing*, 61:1644–1656, 2013.

[109] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

[110] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst. Fast robust pca on graphs. *Journal of Selected Topics in Signal Processing*, 10(4):740–756, 2016.

[111] N. Shahid, N. Perraudin, V. Kalofolias, B. Ricaud, and P. Vandergheynst. Pca using graph total variation. In *2016 International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4668–4672. IEEE, 2016.

[112] N. Shahid, N. Perraudin, G. Puy, and P. Vandergheynst. Compressive pca for low-rank matrices on graphs. *Transactions on Signal and Information Processing over Networks*, 2016.

[113] N. Shahid, N. Perraudin, G. Puy, and P. Vandergheynst. Low-rank matrices on graphs: Generalized recovery and applications. *unpublished, arXiv:1605.05579*, 2016.

[114] 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. *Signal Processing Magazine*, 30(3):83–98, 2013.

[115] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst. Spectrum-adapted tight graph wavelet and vertex-frequency frames. *Transactions on Signal Processing*, 63 (16):4223–4235, 2015.

[116] D. I. Shuman, M. J. Faraji, and P. Vandergheynst. A multiscale pyramid transform for graph signals. *Transactions on Signal Processing*, 64(8):2119–2134, 2016.

[117] D. I. Shuman, B. Ricaud, and P. Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260–291, 2016.

[118] D. Slepian and H. O. Pollak. Prolate spheroidal wave functions, fourier analysis and uncertainty—i. *Bell System Technical Journal*, 40(1):43–63, 1961.

[119] S. Smale and D.-X. Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1(01):17–41, 2003.

[120] S. Smale and D.-X. Zhou. Shannon sampling ii: Connections to learning theory. *Applied and Computational Harmonic Analysis*, 19(3):285–302, 2005.

[121] S. Smale and D.-X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive approximation*, 26(2):153–172, 2007.

[122] A. J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning theory and kernel machines*, pages 144–158. Springer, 2003.

[123] O. Smolyanov, H. von Weizsäcker, and O. Wittich. Brownian motion on a manifold as limit of stepwise conditioned standard brownian motions. *Stochastic processes, physics and geometry: new interplays, II*, 29:589–602, 2000.

[124] O. G. Smolyanov, H. v. Weizsäcker, and O. Wittich. Chernoff's theorem and discrete time approximations of brownian motion on manifolds. *Potential Analysis*, 26(1):1–29, 2007.

[125] G. Strang. The discrete cosine transform. *SIAM review*, 41(1):135–147, 1999.

[126] A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst. Accelerated filtering on graphs using lanczos method. *unpublished, arXiv:1509.04537*, 2015.

[127] G. J. Tee. Eigenvectors of block circulant and alternating circulant matrices. 2005.

[128] D. Thanou, D. I. Shuman, and P. Frossard. Learning parametric dictionaries for signals on graphs. *Transactions on Signal Processing*, 62(15):3849–3862, 2014.

[129] L. V. Tran, V. H. Vu, and K. Wang. Sparse random graphs: Eigenvalues and eigenvectors. *Random Structures & Algorithms*, 42(1):110–134, 2013.

[130] M. Tsitsvero, S. Barbarossa, and P. Di Lorenzo. Signals on graphs: Uncertainty principle and sampling. *Transactions on Signal Processing*, 64(18):4845–4860, 2016.

[131] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416, 2007.

[132] U. Von Luxburg, M. Belkin, and O. Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[133] P. Welch. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *Transactions on audio and electroacoustics*, pages 70–73, 1967.

[134] N. Wiener. Generalized harmonic analysis. *Acta mathematica*, 55(1):117–258, 1930.

[135] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, MA, 1949.

[136] C. K. Williams. Prediction with gaussian processes: From linear regression to linear prediction and beyond. In *Learning in graphical models*, pages 599–621. Springer, 1998.

[137] K. Yu, J. Bi, and V. Tresp. Active learning via transductive experimental design. In *Proceedings of the 23rd international conference on Machine learning*, pages 1081–1088. ACM, 2006.

[138] C. Zhang, D. Florêncio, and P. A. Chou. Graph signal processing–a probabilistic framework. *Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2015-31*, 2015.

[139] D. Zhou and B. Schölkopf. A regularization framework for learning from graph data. 2004.

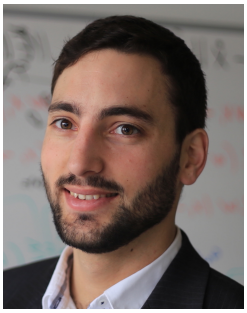
[140] D. Zhou, J. Huang, and B. Schölkopf. Learning from labeled and unlabeled data on a directed graph. In *the 22nd international conference*, pages 1036–1043, New York, New York, USA, 2005. ACM Press.

[141] D. Zhou, J. Huang, and B. Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in neural information processing systems*, pages 1601–1608, 2006.

Nathanaël Perraudin
La Pâle 15
1934 Le Cotterg

Phone number : +41 76 822 42 60
Personal e-mail : nathanael.perraudin@gmail.com
Website : http://perraudin.info

Date of birth: 16/09/1987
Nationality: Swiss
Languages: English, French, German, Spanish



Summary

Objective	Looking for a job involving challenging problems solving, machine learning and data science.
Profile	Research scientist with experience in **machine learning, optimization, graph theory and data science.**

Technical skills

Data Science	Data modeling, signal processing, graph-based data science, data mining, probabilistic modeling, sampling
Machine Learning	Supervised and unsupervised learning, recommendation systems, graph-based learning, deep learning, statistics
Optimization	Large-scale convex and non-convex optimization, proximal methods, neural networks, algorithm design
Programming	MATLAB, Python, C++, Git
Web Development	PHP, MySQL, HTML, CSS, Joomla

Working experience

04. 2013 - Today

Ph.D. in signal processing and machine learning at LTS2 - EPFL (Ecole Polytechnique Fédérale de Lausanne): supervised by Prof. **Pierre Vandergheynst**: I investigate and develop graph-based algorithms with application in data processing and machine learning. My thesis provides insights in three topics:

- **Graph quality assessment** using uncertainty principles (How much information is available to help the learning process?)
- **Probabilistic models** that leverage a graph structure (How can we model data having a graph structure?)
- **Graph based semi-supervised learning algorithms** (How to learn from the data and predict new outcomes?)

09. 2014
06. 2015

Summer school organizer: **"Key Insights on Networks and Graphs"** A summer school on spectral graph theory and distributed computing

- Team leader and coordinator of the organization committee
- 5 days events gathering 30 participants and 4 international speakers

09. 2012 - 03. 2013

Research Scientist in ARI (Acoustic Research Institute)

- Leveraged optimization techniques to accelerate and improve a phase reconstruction algorithm
- Designed and implemented an automatic audio reconstruction algorithm

Education

2013 - Today	**EPFL Ph.D.** in signal processing with graph-based applications in machine learning.
2010 - 2012	**EPFL Master of Science** in Electrical and Electronic Engineering (Specialisation in Information Technologies) **Ranking 3rd**, (out of 42) with a GPA of 5.80/6
2010 - 2012	**EPFL Minor Diploma** in Energy with a GPA of 5.61/6
2007 - 2010	**EPFL Bachelor of Science** in Electrical and Electronic Engineering, awarded with the excellence scholarship (GPA of 5.52/6)
2002 - 2007	**High School Diploma** "Lycée Collège de la Royale Abbaye de Saint-Maurice" (Specialisation in Physics and Mathematics)

Prizes

2010 - 2011	**Excellence scholarship**, award granted to a maximum of one student per section, based mostly on academic performance.

Software contributions

UNLocBoX	I am the founder and the lead contributor (80% of the code-base) of an **open-source convex optimization library**. I have designed the system and implemented many **state-of-the-art methods**. Its main advantage over other libraries is that it combines **high performance** with ease of use, transparency and modularity. Link: `https://lts2.epfl.ch/unlocbox/`
GSPBox	I am founder and lead contributor (50% of the code-base) of the **only library available online** to perform graph-based signal processing. GSPBox contains various **large scale** machine learning algorithms for semi-supervised learning, filtering and network inference. Link: `https://lts2.epfl.ch/gsp/`
Audio Inpainting	I am the designer and the main contributor of the **only algorithm able to automatically recover long missing parts of a song**. The algorithm first captures the music structure and then finds an appropriate replacement for the missing part. Demo: `https://lts2.epfl.ch/web-audio-inpainting/`

Language Skills

French	Mother tongue
English	C1 (Good operational command)
German	B2 (Generally effective command)
Spanish	A2 (Basic command)

Extra-Curricular Activities

2013 - Today	Tango (Argentine) dancing
2007 - 2014	Member and committee member at "Voix de Lausanne" (choir)
2005 - 2012	Ski instructor at the Swiss Ski School of Verbier (ESSV) (First Degree of Swiss Ski Patent)
2001 - 2007	Trainer in Basketball Club Bagnes