Distributed Graph Filters

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K. C. A. M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op dinsdag 11 March 2015 om 15:00 uur

door

Andreas Loukas

informaticus
geboren te Athene, Griekenland.

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. K. G. Langendoen
Prof. PhD M. A. Zúñiga

Samenstelling promotiecommissie:

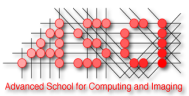
Rector Magnificus,	voorzitter
Prof. dr. K. G. Langendoen,	Technische Universiteit Delft, promotor
M. A. Zúñiga, PhD	Technische Universiteit Delft, copromotor
Prof. J. Gao, PhD	Stony Brook University
Prof. dr. A. Liotta,	Technische Universiteit Eindhoven
Prof. dr. ir. G. J. T. Leus,	Technische Universiteit Delft
Prof. dr. ir. H. J. Sips,	Technische Universiteit Delft
Prof. dr. N. Trigoni	University of Oxford

Prof. J. Gao Doe heeft als begeleider in belangrijke mate aan de totstandkoming van het proefschrift bijgedragen.

ISBN 978-94-6203-806-6

Printed by CPI-Wöhrmann Print Service – Zutphen.

An electronic version is available at `http://repository.tudelft.nl/`.



This work was carried out in the ASCI graduate school.
ASCI dissertation series number 324.

This work was funded by D^{2}S^2,
a project sponsored by Technologiestichting STW.

"I may not have gone where I intended to go,
but I think I have ended up where I needed to be."

—Douglas Adams,
The Long Dark Tea-Time of the Soul

Contents

Summary

We have recently seen a surge of research focusing on the processing of graph data. The emerging field of signal processing on graphs focuses on the extension of classical discrete signal processing techniques to the graph setting. Arguably, the greatest breakthrough of the field has been the extension of the Fourier transform from time signals and images to graph signals, i. e., signals defined on the nodes of irregular graphs. Analogously to how the Fourier transform allows us to decompose complex signals in terms of their fundamental frequencies, the spectral transform describes signals in terms of their relation to the underlying graph.

The rigorous examination of the relation between signal and graph has lead to the design of distributed graph filters, graph analogues of classical filters. Graph filters enable us to observe graph data at different scales, effectively separating fine details from inherent signal trends. For instance, a low-pass graph filter controls the size of observable signal structures, attenuating structures of small size, such as those attributed to noise. Beyond noise removal, graph filters are useful for revealing communities (low-pass), identifying event-regions (band-pass), and detecting anomalies (high-pass).

Yet, despite their interesting properties, current distributed graph filters have so far been limited. To begin with, it is currently assumed that all data remain static for the duration of computation. When the signal is time-varying and the graph topology dynamic, the computation becomes challenging. Even further, filtering efficiency depends on the correct choice of scale—roughly the number of hops a filter takes into account. To choose the scale correctly however, one must have *a-priori* information about the observed phenomenon, as well as of the instrument of observation—in our case, the graph topology; information which is rarely available and often changes over time.

The main contribution of this thesis is tackling the above limitations. First, we relax the computational assumptions posed by current graph filters. We propose distributed graph filters that converge fast, even in the presence of dynamics. Our filters are shown robust to message loss, and able to track time-varying signals and graphs. Second, we set the foundations of distributed scale-invariant analysis of

graph signals. According to classical scale-space theory, if no *a-priori* information about a signal is known, one must observe it at all possible scales. In an analogous way, we show that the scale-invariant observation of a graph signal entails filtering it with a small family of graph filters. Scale-space analysis is therefore possible on graphs, and incurs an overhead equivalent to that of a distributed graph filter.

We demonstrate the usefulness of our algorithms by applying them to a number of important information processing problems in sensor networks. Among others, our filters are shown to expand the scope of potential-field search methods, to enhance the detection accuracy of spatial event regions and boundaries, and to improve the identification of signal peaks and pits. Simulations and experiments, demonstrate that our algorithms are robust to the difficult conditions posed by wireless communications (such as asymmetric links, phantom effects, message loss, and asynchrony), and that they scale to very large networks.

Samenvatting

Het verwerken van graaf-gebaseerde data heeft recentelijk een grote vlucht genomen. Het opkomende vakgebied van signaalbewerking op grafen richt zich op het toepasbaar maken van klassieke discrete signaalbewerkingen voor graafstructuren. Recentelijk is een grote doorbraak bereikt met de uitbreiding op de Fouriertransformatie van het tijds- en beeld-domein naar grafen, dwz. signalen aanwezig op de knooppunten van irreguliere grafen. Analoog aan hoe de Fouriertransformatie complexe signalen opdeelt in hun fundamentele frequenties, beschrijft de spectraal transformatie de relatie tussen signaal en de eigenschappen van de onderliggende graaf.

De uitvoerige bestudering van de relatie tussen signaal en graaf heeft geleid tot de notie van gedistribueerde graaffilters, analoog aan klassieke signaalfilters. Deze graaffilters maken het mogelijk om graafdata te beschouwen op diverse schaalgroottes, waarbij de fijne details van de hoofdlijnen gescheiden kunnen worden. Zo zal een laagdoorlaatfilter voor grafen, bijvoorbeeld, de fijne structuren (hoogfrequente ruis) onderdrukken om een goed beeld te geven van de hoofd trend in de graafdata. Naast deze ruisonderdrukking kunnen graaffilters ook gebruikt worden om clusters (laagdoorlaatfilter), event-regions (banddoorlaatfilter) en afwijkingen (hoogdoorlaatfilter) zichtbaar te maken.

Ondanks de geschetste mogelijkheden zijn de huidige gedistribueerde graaffilters nog erg beperkt in hun toepassingen. Ten eerste gaat men ervan uit dat de data constant is gedurende de berekening van het graaffilter. Echter als de data varieert met de tijd, en ook de topologie verandert, dan wordt er (te) veel gevraagd van de berekenmethodes. Ten tweede is de effectiviteit van een filter afhankelijk van de correcte instelling van de schaalgrootte, grofweg de reikwijdte (het aantal stappen) in de graaf dat het filter in ogenschouw neemt. De optimale schaalgrootte hangt af van de data en de topologie, en is dus onmogelijk *a priori* vast te stellen als er van enige dynamiek sprake is.

Dit proefschrift draagt in belangrijke mate bij aan het verminderen, dan wel opheffen, van bovengenoemde beperkingen. Allereerst introduceren we gedistribueerde graaffilters die snel convergeren, zelfs als de data en/of topologie aan verandering

onderhevig zijn, en tolerant zijn ten opzichte van het verlies van boodschappen in het communicatienetwerk (message loss). Ten tweede leggen we het fundament onder de gedistribueerde, schaalonafhankelijke analyse van signalen op (dynamische) grafen. Vanuit de klassieke scale-space theorie is bekend dat men alle mogelijke schalen in ogenschouw dient te nemen als er geen *a priori* informatie over de data is. Met behulp van dit inzicht tonen we aan dat men voor het uitvoeren van analyses op grafen kan volstaan met een kleine verzameling van schaalonafhankelijke graaffilters, die bovendien slechts dezelfde rekenkracht vergen als bestaande gedistribueerde filters.

De toepasbaarheid van gedistribueerde graaffilters, en ihb. die van schaalonafhankelijke filters wordt aangetoond middels een reeks van voorbeelden uit het domein van de draadloze sensornetwerken. We laten zien dat onze filters gebruikt kunnen worden om zoekalgoritmen op basis van potentialen te implementeren, om events(grenzen) nauwkeuriger te lokaliseren, en om pieken en dalen (maxima en minima) in de graafdata te identificeren. Middels simulaties en experimenten tonen we aan dat gedistribueerde graaffilters bestand zijn tegen de weerbarstige praktijk van draadloze netwerken waar boodschappen vaak verloren gaan, knopen onderling niet gesynchroniseerd zijn, en ruis in de waargenomen data makkelijk voor fantomen kan zorgen. De efficiëntie van onze algoritmen staat bovendien toe dat de graaffilters ook toegepast kunnen worden op grootschalige netwerken.

Nomenclature

$\mathcal{G}$	a graph
$\mathcal{V}$	node set of $\mathcal{G}$
$\mathcal{E}$	edge set of $\mathcal{G}$
u, v	nodes
$u \sim v$	u and v are adjacent
n	number of nodes
m	number of edges
$\deg(u)$	degree of node u
δ	minimum degree
Δ	maximum degree
$d(u, v)$	shortest path distance between u and v
d	graph diameter
A	graph adjacency matrix
D	diagonal degree matrix
L	discrete graph Laplacian matrix
$\mathcal{L}$	normalized graph Laplacian matrix
$\mathfrak{L}$	generalized graph Laplacian matrix
T	random-walk matrix
P	random-walk normalized Laplacian or averaging matrix
I	identity matrix
x	$n \times 1$ vector, node-defined signal
y	$n \times 1$ vector, output of a filter
$\mathbb{1}$	all ones vector
$\langle x, y \rangle$	inner product of vectors x and y
$\|x\|$	ℓ_2 norm of x
$\lambda_k(M)$	k-th eigenvalue of matrix M
$\lambda_{max}(M)$	largest eigenvalue of matrix M
$\lambda_{min}(M)$	smallest eigenvalue of matrix M
λ_k	k-th eigenvalue of matrix $\mathcal{L}$
μ_k	k-th eigenvalue of matrix P

$\phi_k(M)$	k-th right eigenvector of matrix M
$\phi_k^\top(M)$	k-th left eigenvector of matrix M
ϕ_k	k-th right eigenvector of matrix $\mathcal{L}$
$\phi_k^\top$	k-th left eigenvector of matrix $\mathcal{L}$
ϑ_k	k-th right eigenvector of matrix P
ϑ_k^{-1}	k-th left eigenvector of matrix P
$\sigma_{max}(M)$	largest singular value of matrix M
$\sigma_{min}(M)$	smallest singular value of matrix M
$\mathcal{D}(x)$	set containing the weak nodal domains of x
$\mathfrak{S}(x)$	number of strong nodal domains of x
$\mathfrak{W}(x)$	number of weak nodal domains of x
$\mathcal{R}_M(x)$	Rayleigh quotient of matrix M evaluated at x
$\mathbf{H}_t$	Heat kernel
$\mathbf{T}_t$	Page-rank heat kernel
$\mathbf{P}_t$	Random-walk kernel
$\mathbf{P}_\varphi$	Potential kernel
LoG	Laplacian of Gaussian
DoG	Difference of Gaussians
LoP	Laplacian of Potential
LoG	Difference of Potentials
S	scale-space matrix
$\mathbf{K}_s$	scale-space kernel

1

Introduction

The story so far: In the beginning the Universe was created.
This has made a lot of people very angry and been widely regarded as a bad move.

—Douglas Adams,
The Restaurant at the End of the Universe

DURING the last decades we have seen an unprecedented trend in the history of computing: scientists are gradually shifting their analysis from the study of the individual to that of the collective. We can witness the shift on multiple fronts. By combining data from multiple sources, researchers have began to observe and analyze the collective behavior of crowds, the physical changes of our environment, the social and monetary interactions of populations, the spread of disease, and the consumption of energy in cities—among others.

Undoubtedly, each of these domains exhibits unique assumptions and objectives. Yet in numerous applications, including sensing [9, 91, 92], socioeconomics [36, 56], energy distribution [40], epidemiology [90], and biology [54], one finds a common underlying approach: modeling the dependencies of interconnected entities as graphs. This ubiquity of graphs has lead to a surge of research on graph/network analy-

1

sis[1]. To date, researchers have produced a vast body of work addressing classical problems, such as graph coloring [46], clustering [116], and centrality computation [97]. They have also examined the structural properties of many graph types, such as random-geometric graphs and planar graphs modeling wireless and proximity networks [29], small-world and scale-free graphs modeling social and economical networks [2], as well as trees, regular, and random graphs [37].

The study of graph connectivity is of paramount importance. But it is also of equal importance to analyze the information imbued in them. In social networks, it is not only the interconnection of individuals that is important, but also their preferences, actions and statuses. Analyzing the propagation of rumors for instance is crucial for identifying false claims and their sources [105]. Similarly, in wireless networks the local state and sensor data of each device are crucial for inferring and controlling the global behavior—whether said behavior concerns the network itself (e. g., harvested energy [115], traffic load [61]) or the process the network monitors (e. g., environmental conditions in a greenhouse). We refer to the information held by the nodes of a network as a *graph signal*. Analogous to time or spatial signals, a graph signal is a collection of scalars –one for each node– describing how a process varies across the graph.

Over the last few years, the emerging field of signal processing on graphs has made significant advances on the analysis and processing of graph signals [100, 109]. Arguably, the most significant advance has been the derivation of a spectral framework for analyzing information. Analogous to how the Fourier transform allows us to decompose complex signals in terms of their fundamental frequencies, the spectral transform describes graph signals in terms of their relation to the spectral properties of the underlying graph. This insight has lead to novel algorithms for the filtering and denoising [110, 129], regularization [113], and compression [132] of graph data.

Still, there are many challenges the field has yet to face. *A fundamental challenge entails understanding the relation between the graph topology and information.* That is, what is an intuitive way to quantify the size of signal structures? Furthermore, how can we devise filters that attenuate structures according to their size? *A second challenge relates to computation.* It is convenient to assume that the processing takes place in a single computer and that all data remain static. However, the sheer size of many graphs available today, such as social networks or the Internet, necessitates a distributed approach, in which the computational load is split among many shared-

[1]The term 'graph' refers to a combinatorial structure, whereas 'network' implies a specific application, such as a sensor or social network. In this thesis, the two terms are often used interchangeably.

nothing clusters or clouds, forming themselves a network. Even further, data that change over time render static approaches impractical [110, 129]. The topology and data for example of Twitter are not only massive –many millions of users– but also highly dynamic [68]. To make computation practical, we need distributed algorithms that track changes of the graph signal and topology.

The fundamental and computational challenges of graph signal processing are particularly prominent in the field of sensor networks, where a large number of small, cheap, portable, wirelessly-interconnected devices perform large-scale sensing [1]. Because of their ability to produce continuous streams of data with very high spatial and temporal resolution, sensor networks can be extremely useful. For instance, large networks of battery-powered sensors can monitor *physical phenomena* spanning over large areas –such as oil spills, chemical spills, and pollution clouds– over long periods of time [9, 91, 92]. Sensor networks are also very useful in monitoring *social phenomena*. By organizing personal wearable devices into networks, one obtains a large-scale view of the social activities of humans [85, 86, 87]. The data can be used for example to mine the interactions of inhabitants of a city [18, 55], to understand their mobility patterns [35], and to infer emergent crowd behaviors [99].

So far, research in information (and signal) processing for sensor networks has yielded many notable results, such as distributed algorithms for sensor aggregation [31, 111] and estimation [27, 95], as well as a plethora of geometry-based algorithms [44, 101, 103]. Nevertheless, the timely and accurate detection of global phenomena and their properties from a large distributed sensor network remains a great challenge. Collecting data centrally faces problems, such as bandwidth constraints, high delay, and single points of failure. Performing high-level inference within the network hinges on distributedly solving low-level signal processing problems in a manner both accurate but also robust to the uncertainties and dynamics of wireless networks.

By developing efficient distributed algorithms for the processing of graph signals, this thesis advances the state-of-the-art on two fronts. First, it contributes to the field of signal processing on graphs. Our contribution is threefold: (*i*) we propose asynchronous graph filters that converge fast, even in dynamic graphs and time-varying signals, (*ii*) we provide insights on how graph filters affect the shape of a signal and (*iii*) we set the foundations of scale-invariant signal analysis. Second, the thesis addresses a number of important problems in sensor networks. Our filters form basic processing primitives: they are useful for localized aggregation, expand the scope of greedy search methods, enhance the detection accuracy of spatial event regions and boundaries, and improve the identification of signal structures.

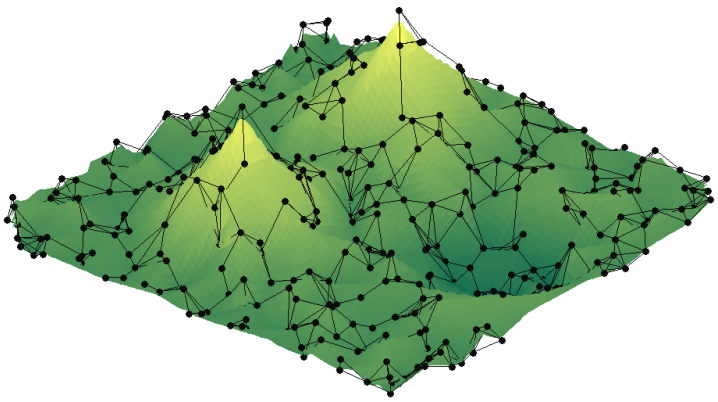


Figure 1.1: The sensor information of a network often exhibits interesting spatial structure. We can think of it as an information landscape with complex features such as pits, peaks, ridges, and passes. The challenge is to infer the spatial structure when the location of nodes is unknown.

1.1. Problem Statement

Signal processing in graphs, and sensor networks in particular, comes with its unique challenges. In the following, we review some fundamental and computational challenges of the field. The discussion paves the way for the central contribution of this thesis—the design of efficient distributed graph filters and their usage in sensor networks.

1.1.1. How Connectivity Affects Information

The first challenge is a fundamental one. We begin with a simple, yet crucial observation: *In networks, the nature of information is connected to and –to a large extent– determined by the network topology.* Basically, it is not only the data that nodes produce that matters, but also how these data are interconnected. Let us clarify this with two examples:

First, we will consider a sensor network that is monitoring a physical phenomenon, e. g., a network of floating drifters monitoring the concentration of pollutants in a lake [50]. Figure 1.1 illustrates –in a simplified way– the network and the information it measures. In the figure, sensor nodes are printed as black dots—their x and y coordinates are determined by their physical location, whereas their z coordinate, as well as the color of the overlaid surface, denotes the value a node senses, e. g., the concentration of the pollutant. Every two nodes in communica-

tion vicinity are joined with a black line. By looking at the figure, one interprets information based on spatial constraints: the surface exhibits two main peaks, a valley in between and one pit. In reality however, neither the nodes themselves, nor the network engineer necessarily possess spatial information. The localization of nodes is feasible algorithmically, but –especially in indoor settings– the process is notoriously imprecise and generally resource hungry. When the location of nodes in space is unknown, one may use the network topology to infer spatial correlation. Though affected by a multitude of factors, the wireless propagation is generally a decreasing function of distance [52, 89]. As such, any two nodes that are in wireless proximity (wireless links are denoted in the figure with black lines) are more likely to observe the same phenomenon. By exploiting this relation, one can infer the spatial properties of information based on the network topology.

The second example concerns the internal operation of a communication network. Large-scale wireless networks are notoriously unreliable—especially in the presence of dynamics. Wireless devices are inherently limited by their resources. Because they typically operate on cheap batteries, nodes have a small energy budget and limited computation capabilities. Furthermore, as the network density increases, the per-node available bandwidth diminishes significantly. These challenges have spawned a significant research effort: researchers seek to optimize network operation by adapting to resource availability information [61, 115, 127]. It is crucial to observe that, even though each node in a communication network is independent, its state is directly affected by its communication with the rest of the network. Nodes for example that lay on the path connecting two otherwise isolated parts of the network are far more likely to use up their energy fast by constantly forwarding packets. As such, to understand and optimize network operation, we need to consider and exploit how resource availability varies throughout the network. The paths chosen by a resource-aware routing protocol, for instance, should be affected by how bandwidth and energy availability varies across bottlenecks.

From the above examples we can see that, *to understand and control networks, we must analyze and process their information in a manner that is inherently tied to their topology.* Though this statement appears intuitive, it is also vague. What is the relation between network topology and information exactly? For example, how is information affected by connectivity? That is, how is its nature affected or determined by the topological properties of a network? Furthermore, what is an appropriate way to quantify the properties of information in terms of scale, i. e., what notion of size is suitable for graphs? In summary, what is a concrete way of thinking about information in networks?

1

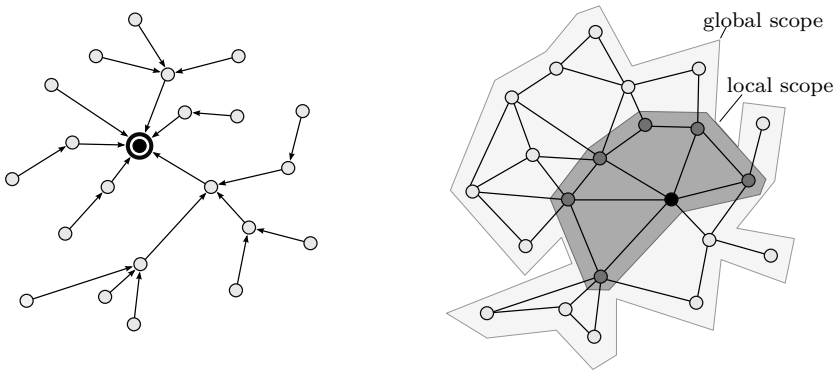
1.1.2. Computational Challenges

It is not sufficient to process network information accurately, one also has to do so efficiently. In the following, we identify three computational challenges pertaining to wireless networks: scalability, dynamics, and the unpredictability of wireless links. Addressing all three is integral to using large-scale sensor networks effectively.

The first computational challenge is designing algorithms that process information in a scalable manner. The classical approach to information processing is centralized. By gathering all available data on a single computer, one obtains a global view, greatly facilitating analysis. However, a centralized approach[2] is not always feasible. The main reason is that collection algorithms cannot scale—see Figure 1.2(a). Though a plethora of algorithms and protocols exist that address the data collection problem [49, 69, 88], these are mostly targeted towards *small-* to *medium-sized* networks (at the order of hundreds of nodes), with relatively *low update-rates* (a few readings every minute). As the size of networks increases, the energy, communication overhead, and delivery delay of collection grows significantly. When the network must respond to its environment in real-time, nodes cannot afford to wait for the directions of a central entity. In summary, the sheer amount of data generated by thousands of devices renders centralized approaches impractical.

The prevalent solution in attaining scalability is distributing the computational load. In a distributed approach, each node acts individually, based on its own perception of the world. But even then, not all distributed approaches scale equally well. In an analogous fashion to how one judges the efficiency of a centralized algorithm based on its computational complexity, the scalability of a distributed algorithm is captured by its time complexity. Roughly defined, the *time complexity* of a distributed algorithm is the amount of time required until the algorithm terminates—for a concrete definition see Chapter 2. In this thesis, we will consider the most efficient and scalable type of distributed algorithms (in terms of time complexity): *local algorithms* [66]. As shown in Figure 1.2(b), local algorithms are distributed algorithms that base their decisions only on partial network information (though the data might pertain multiple hops)—see Section 2.1 for examples. In the figure, the shaded areas depict the information that the node in the center bases its decisions on if it uses a local algorithm (dark gray) and otherwise (light gray). For this reason, local algorithms exhibit constant time complexity and are independent of any measure of network scale, e. g., the number of nodes or the network diameter.

[2] A note on terminology. Strictly speaking, data collection is achieved by a distributed algorithm. Nevertheless, because all data is gathered in a single location, the processing is centralized.



(a) centralized processing (b) distributed processing

Figure 1.2: (a) Though very efficient in small networks, collecting and processing information centrally (the sink is annotated with a ring) is impractical. (b) In large networks, it is generally more efficient to distribute the processing load among many nodes. Still, only algorithms that use partial information (dark shaded area) are truly scalable.

The second computational challenge is designing algorithms that can handle dynamics. We use the term "dynamics" to refer to two distinct phenomena. The first is *node mobility.* Mobility greatly facilitates sensing: it increases sensing coverage [76] and expedites information diffusion [104]. Nevertheless, mobility also makes computation challenging. Nodes in a mobile network have limited time at their disposal to gather information and act, before the situation changes and their decisions become stale. Algorithms therefore must be able to converge fast using partial knowledge. The second type of dynamics is *time-varying information.* It seems straightforward that any algorithm running on a sensor network should be able to track the changes of information. Yet it is often hard to do so in practice. In fact, many competitive algorithms operate under the assumption that the information (and the network) stays invariant for the duration of computation [33, 60]. This renders them difficult to work with in many dynamic scenarios. On the contrary, this thesis centers largely on distributed algorithms that converge to the solution even when the information and the topology constantly change.

Last, in sensor networks, algorithms have to be able to cope with the unpredictability of wireless links. In practice, wireless links are intermittent [126]. Furthermore, they are often poorly correlated with the underlying physical space. Especially in indoor networks, one frequently observes that distant devices can communicate,

were adjacent ones can not. Dealing with such networks is not straightforward. As we will show in Chapter 5, a possible mismatch between the network topology and the physical space can severely hamper processing.

1.2. Thesis Contributions and Outline

This thesis focuses on the design and analysis of distributed algorithms for networks (or more generally graphs) that converge fast with limited information. These algorithms are founded on a simple yet powerful observation: *the graph spectrum is an ideal tool for understanding the properties of graph signals.* This insight forms the basis of a spectral framework for studying and processing information in networks.

The spectral theory of graph filters—Chapter 2. We rely on classical results from spectral graph-theory [14, 26], signal processing [109], and machine learning [65], to demonstrate that the eigenvectors of the graph Laplacian form a natural decomposition basis for network information. This means that the spatial properties of information are fundamentally connected to the topological properties of the underlying graph. In an analogous way to how, in classical signal processing, one studies a signal in the frequency domain, the spectral domain allows us to decompose complex network information according to the size (or variation) of its structures. Based on their spectral order, we can distinguish large phenomena, denoting inherent signal trends, from small variations, such as noise. Furthermore, the definition of size is directly determined by how a graph is connected.

The above insight gives rise to *graph filters*—a spectral approach for information processing on networks. Analogous to classical filters, graph filters attenuate signal structures according to their spectral order. Though standard graph filters exist, we find that they are not appropriate for wireless networks: they hinge on synchronous execution and cannot tolerate dynamics. To this end, this thesis focuses on the design of asynchronous graph filters that converge even when the network topology and the information change over time.

Distributed low-pass graph filters—Chapter 3. The most common filter is a low-pass filter. Low-pass graph filters are very useful in attenuating small signal variations, such as noise and artifacts of irregular wireless propagation. Furthermore, due to their ability to eliminate local signal extrema, low-pass filters facilitate greedy search methods, such as greedy routing and gradient navigation, in which a mobile entity (packet, human or robot) ascends by following the local gradient [119].

We present a novel asynchronous low-pass graph filter, called the *potential kernel*, which, contrary to the well-known heat kernel, is suitable for wireless networks. First, we perform spectral analysis. We show that, similar to the heat kernel, the potential kernel is low-pass (low-pass graph filters attenuate signal structures of high graph frequency) and analyze its ability to eliminate local extrema. We then focus on computation. We characterize convergence of the distributed algorithm in both static and dynamic networks. Our analysis shows that the potential kernel is a local algorithm and that it can handle both network and information dynamics. To demonstrate the feasibility of our approach in real wireless networks, we evaluate the potential kernel with simulations and experiments on a 100+ node testbed.

Distributed band-pass graph filters—Chapter 4. A classical problem of sensor networks is the identification of spatial events and their boundaries—the problem is illustrated graphically by Figure 4.1. Event detection breaks down to two filtering objectives: the computation of signal curvature (low spectral order) and the elimination of noise (high spectral order). The problem can therefore be tackled by designing distributed graph filters that only retain the signal structures of medium spectral order, i. e., band-pass graph filters.

Based on this insight, we propose two families of band-pass filters. The first, which is derived from image processing techniques, exhibits very good resilience to noise but is synchronous and suffers from low detection resolution. By building on the potential kernel, the second family of filters addresses these shortcomings: they are asynchronous and clearly identify event boundaries. We compare and evaluate our findings based on analysis, simulations, and experiments. Our algorithms can track complex time-varying events with high accuracy, even when the nodes are mobile.

Distributed scale-space theory on graphs—Chapter 5. In sensor networks, the high-level inference of global properties often necessitates efficient and accurate solutions to low-level signal processing problems. The identification of the peaks and pits of a sensed signal for example appears recurrently in various problems, including target tracking, mapping, compression, and navigation. Peak and pit identification is a filtering problem. *False peaks*, caused by noise, low sampling resolution, and sparse topologies, are generally eliminated by low-pass graph filters. Yet, simply filtering is not enough. Whereas most signals contain useful information over many scales of observation, graph filters only focus on one scale—potentially missing valuable information. What is more, when the topology of the graph and the signal do not match, filtering causes the creation of *phantom effects*, i. e., effects

not present in the monitored signal, but brought about by the topology. Phantom and false peaks significantly deteriorate identification accuracy.

Inspired by the scale-space approach used in image processing and pattern recognition [75], we tackle the limitations of graph filters by examining signals at multiple scales of observation. Our contributions are twofold: First, we set forth the foundations of distributed scale-space theory on graphs. We identify the scale-space kernels appropriate for graphs and analyze how efficiently they can be computed distributedly. Second, we examine the behavior of peaks and pits across scale. Based on our analysis, we derive distributed algorithms for the identification of false and phantom peaks. As confirmed by simulations and experiments, our approach exhibits superior accuracy to the state-of-the-art.

Chapters 2, 3, 4, and 5 are based on the following publications:

- A. Loukas, M. Woehrle, P. Glatz, and K. G. Langendoen. On Distributed Computation of Information Potentials. In *International Workshop on Foundations of Mobile Computing*, FOMC. ACM, 2012.
- A. Loukas, M. Zúñiga, M. Woehrle, M. Cattani, and K. G. Langendoen. Act Localy, Think Globaly: On the Reshaping of Information Landscapes. In *International Conference on Information Processing in Sensor Networks*, IPSN. ACM/IEEE, 2013.
- A. Loukas, M. Zúñiga, I. Protonotarios, and J. Gao. How to identify global trends from local decisions? Event Region Detection on Mobile Networks. In *International Conference on Computer Communications*, INFOCOM. IEEE, 2014.
- A. Loukas, M. Cattani, M. Zúñiga, and J. Gao. Graph Scale-Space Theory for Distributed Peak and Pit Identification. In *International Conference on Information Processing in Sensor Networks*, IPSN. ACM/IEEE, 2015.

2

The Spectral Theory of Graph Filters

"We demand rigidly defined areas of doubt and uncertainty."

—Douglas Adams,
The Hitchhiker's Guide to the Galaxy

The information carried by a network is undoubtedly connected to its topology. Though this statement seems intuitively clear, it is also very vague—what is the relation between the network topology and information, exactly? This chapter gives a rigorous treatment to this fundamental relation by drawing upon and combining recent work on the fields of spectral graph theory [14, 26], machine learning [65, 113, 131], image [129] and signal processing [100, 109]. The use of spectral graph theory presents a recent, exciting way of thinking about information in networks.

After defining the network and computational models, we will delve into the spectral theory of graphs. The discussion gives rise to the main research question addressed in this thesis: *what is a graph filter and how can it be computed efficiently in a sensor network?*

2.1. Models

Though the theory and methods developed in the field of signal processing on graphs pertain networks of many types, such as sensor networks, socioeconomic networks, and biological networks, this thesis focuses on one type in particular: *sensor networks*. Besides carrying a wealth of information which we can exploit, sensor networks present many algorithmic challenges: they are inherently decentralized, asynchronous, and severely resource constrained[1]. To make these challenges explicit, this chapter begins by defining the *network* and *computational models*. The first details our assumption about how the network communicates, whereas the second formalizes issues of computation and complexity.

2.1.1. Network Model

As customary, we model a sensor network as an undirected[2] graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the node set of cardinality n and $\mathcal{E}$ is the edge set. Nodes $u_i, u_j \in \mathcal{V}$ represent simple devices, minimally imbued with processing capabilities, memory, sensors, and a wireless radio. By transmitting information wirelessly, each node u_i communicates locally with nearby devices u_j, termed the neighbors of u_i. To indicate that two nodes u_i, u_j are in communication vicinity we use the shorthand notation $u_i \sim u_j$. The neighborhood of a node u_i is captured by the neighbor set $\mathcal{V}_{u_i} = \{u_j : u_i \sim u_j\}$ and the adjacent edge set $\mathcal{E}_{u_i} = \{(u_i, u_j) : u_i \sim u_j\}$, both having cardinality equal to u_i's degree, $\deg(u_i)$. Symbols δ and Δ refer to the minimum and maximum degrees, respectively.

A network however also holds information. Though information might pertain either to nodes (sensor data) or edges (e. g., link quality), this thesis focuses on the former. We model node information as a, possibly time-varying, node-defined function $x : \mathcal{V} \to \mathbb{R}$, assuming a real value $x(u_i)$ on each node u_i. To emphasize the analogy of our approach to classical signal processing, we refer to function x as a graph *signal*. Nevertheless, x goes by many names. To give two prominent examples, sensor network literature customarily calls it a *scalar-field*, whereas in graph theory it is usually treated as a *node weight*.

Despite its simplicity, this network model is general enough to capture a wide range of applications, such as the measurements of a sensor network (e. g., temperature, humidity), the statistics of the internal operation of the communication

[1] Indeed, these are the challenges which make the algorithmicist's job stimulating!

[2] Though we limit our presentation to undirected graphs, this is not a fundamental limitation, but a matter of convenience.

stack (e. g., routing load) or of the device itself (e. g., energy) in a communication network, as well as purely graph-theoretic properties (e. g., node degree) and information (e. g., user status and preferences) in an ad-hoc network.

2.1.2. Computational Model

A distributed algorithm running on a wireless node u_i performs two operations in arbitrary order: it exchanges data and computes locally. It is however convenient to impose some additional structure:

Assumption 2.1. *The computation proceeds in discrete rounds (or iterations) t.*

Based on whether or not rounds of neighboring nodes overlap, we distinguish two versions of the model: the *synchronous* and the *asynchronous* version. In practice, wireless networks are asynhronous. Executing a synchronous algorithm is possible, but incurs a recurrent synchronization overhead [4].

Assumption 2.2. *During each round t, u_i sends a message to each $u_j \sim u_i$.*

This assumption implies a reliable local broadcast and can be implemented in either of two ways: deterministically by using a local schedule and probabilistically by repeated beaconing.

Assumption 2.3. *Each node knows a number that uniquely identifies it.*

We quantify the computational efficiency of a distributed algorithm in terms of its *time complexity*.

Definition 2.1 (Time complexity)**.** *The time complexity of a distributed algorithm is the number of rounds until every node in the network terminates in the synchronous model with concurrent initialization.*

Based on their time complexity, we distinguish two types of distributed algorithms: *local* and *global*. The characteristic property of local algorithms –that distinguishes them from global algorithms– is that they terminate in $O(1)$ rounds. As such, local algorithms are extremely scalable; they are independent of any measure of network size, such as the number of nodes n, the number of edges m, or the graph diameter d.

To illustrate the difference consider the following thought experiment. Each node attempts to find out whether its value $x(u_i)$ is the largest by executing the following simple algorithm:

Require: Unique identifier u_i.
1: $t \leftarrow 0$, $y(u_i) \leftarrow x(u_i)$, $isMax \leftarrow 1$ ▷ Node assumes it is a maximum.
2: **while** true **do**
3: $t \leftarrow t + 1$ ▷ Next round.
4: **if** $\max_{u_j \sim u_i} y(u_j) \geq x(u_i)$ **then**
5: $y(u_i) \leftarrow \max_{u_j \sim u_i} y(u_j)$, $isMax \leftarrow 0$ ▷ Node finds out it isn't a maximum.
6: **end if**
7: **end while**

Though node u_i only communicates locally with its neighbors, in the worst case, u_i realizes that its value is not the largest after d (diameter) rounds. The algorithm is therefore global. On the contrary, if one changes the while condition as follows:

2: **while** $t \leq t_{max}$ **do**

then each node limits its search in its t_{max} vicinity, and –because it terminates in $O(t_{max}) = O(1)$ rounds– the algorithm is local.

2.2. Spectral Transforms

One of the most powerful ideas in modern physics and engineering is contributed to Joseph Fourier [42]. Fourier showed that we can easily transform any signal between the time (or spatial) and frequency domains.

Let t represent time and ξ frequency. The Fourier transform $\hat{f}$ of an integrable time-varying signal $f : \mathbb{R} \to \mathbb{C}$ is

$$\hat{f}(\xi) \triangleq \int_{-\infty}^{\infty} f(t) e^{-i2\pi\xi t} dt. \tag{2.1}$$

Furthermore, given $\hat{f}$, one reconstructs f using the following inverse transform

$$f(t) = \int_{-\infty}^{\infty} \hat{f}(\xi) e^{i2\pi\xi t} d\xi. \tag{2.2}$$

By studying signals in the frequency domain one extracts information about the frequency and size of inherent signal structures.

2.2.1. Fourier for Graphs

It is very natural to ask: can we use the Fourier transform on graphs? Even though we cannot apply the Fourier transform directly, we can define an equivalent transformation that is suitable. To do so, we have to first define the space where the graph and the signal live. In the most general case, this is a Hilbert space $\mathbb{H}$ equipped with an inner product $\langle\,,\,\rangle$. However, it is also sufficient to consider the n-dimensional Euclidean space $\mathbb{R}^n$ equipped with the usual inner product

$$\langle x, y\rangle \triangleq \sum_{k=1}^{n} x(u_k)y(u_k). \tag{2.3}$$

From Bessel's inequality, we know that projecting any signal $x \in \mathbb{H}$ onto an orthonormal system $\{e_k\}$ is inexact:

$$\left\|\sum_{k=1}^{n}\langle x, e_k\rangle e_k\right\|^2 \leq \|x\|^2, \text{ where } \{e_k\} \text{ is an orthonormal system} \tag{2.4}$$

However, according to Parseval's identity, if $\{e_k\}$ is an orthonormal basis of the Hilbert space, the projection becomes exact:

$$\left\|\sum_{k=1}^{n}\langle x, e_k\rangle e_k\right\|^2 = \|x\|^2, \text{ where } \{e_k\} \text{ is an orthonormal basis} \tag{2.5}$$

A direct consequence is that, as long as we can find an orthonormal basis, i. e., a set of vectors with a dense linear span in $\mathbb{H}$ for which

$$\langle e_k, e_l\rangle = \begin{cases} 0 & \text{if } k \neq l \\ 1 & \text{if } k = l, \end{cases} \tag{2.6}$$

any $x \in \mathbb{H}$ adheres to the following transformation:

Definition 2.2 (Spectral Transform)**.** *The forward and inverse spectral transforms of a signal $x \in \mathbb{H}$ afforded by orthonormal basis $\{e_k\}$ are*

$$\hat{x}_k \triangleq \langle x, e_k\rangle \quad \textit{and} \quad x = \sum_{k=1}^{n} \hat{x}_k e_k. \tag{2.7}$$

Let us review what we have achieved. The spectral transform transforms a signal from its natural domain $\mathcal{V}$, to the domain of the basis $\{e_k\}$. Similar to the Fourier coefficients $\hat{f}(\xi)$, the *spectral coefficients* $\hat{x}_k$ capture the similarity between x and each of e_k. Last, we can retrieve x exactly by summing all *signal components* $\hat{x}_k e_k$. The spectral transform is therefore a generalization of the Fourier transform, which is suitable for graphs.

2.2.2. The Generalized Laplacian and its Spectrum

Our work however is not yet done. What remains is to choose an *suitable* orthonormal basis. Similarly to the basis used by Fourier, i. e., the complex exponential functions, our basis should capture in an intuitive way how fast the signal varies across the graph. We can construct one such basis based on the spectrum of the discrete Schrödinger operator, commonly referred to as the generalized Laplacian [14].

A symmetric matrix $\mathfrak{L}(\mathcal{G})$ is called the *generalized Laplacian* of graph $\mathcal{G}$ if it has non-positive off-diagonal elements and $\mathfrak{L}_{ij}(\mathcal{G}) \neq 0$ only when $u_i = u_j$ or $u_i \sim u_j$[3]. It is easy to see that the spectrum of $\mathfrak{L}$ adheres to the requirements of the spectral transform: from the spectral theorem, the eigenvectors $\phi_k(\mathfrak{L})$ of $\mathfrak{L}$ form an orthonormal basis of $\mathbb{H}$.

But what does this basis capture? When computed at a node u_i, the generalized Laplacian takes the general form

$$\begin{aligned}(\mathfrak{L}x)(u_i) &= \sum_{u_j \sim u_i} (-\mathfrak{L}_{ij}) \left(x(u_i) - x(u_j)\right) + \left(\mathfrak{L}_{ii} + \sum_{u_j \sim u_i} \mathfrak{L}_{ij}\right) x(u_i) \\ &= \sum_{u_j \sim u_i} (-\mathfrak{L}_{ij}) \left(x(u_j) - x(u_i)\right) + p(u_i) x(u_i). \end{aligned} \tag{2.8}$$

The first term is an elliptic operator and captures the *variation* of x, i. e., to what extent a signal changes at each neighborhood. Matrix element $\mathfrak{L}_{ij}$ therefore corresponds to an edge weight between neighbors u_i and u_j. In a similar fashion, the potential function $p : \mathcal{V} \to \mathbb{R}$ corresponds to a node weight which gives extra significance to each node's value.

The following corollary explains how the notion of variation (inherited by matrix $\mathfrak{L}$) relates to each eigenvector $\phi(\mathfrak{L})$.

Corollary 2.1 (Corollary 2.5 [14], Theorem 4.2.2 [53])**.** *Let $\phi_1, \ldots, \phi_n$ the eigenvectors corresponding to the eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ of $\mathfrak{L}$. Let $\Phi_k = \{\phi_1, \ldots, \phi_k\}$ be the subspace spanned by the first k eigenvectors and $\Phi_k^{\perp}$ its orthogonal complement. Then*

$$\lambda_k = \min_{x \in \Phi_k^{\perp}} \mathcal{R}_{\mathfrak{L}}(x) = \min_{x \in \Phi_k^{\perp}} \frac{\langle x, \mathfrak{L}x \rangle}{\langle x, x \rangle}. \tag{2.9}$$

Moreover, $\mathcal{R}_{\mathfrak{L}}(x) = \lambda_k$ for some $x \in \Phi_k^{\perp}$ iff x is an eigenvector corresponding to λ_k.

[3] For convenience of notation, we omit $\mathcal{G}$ when it is clear from the context.

As we previously mentioned, an appropriate basis should capture how fast the signal varies across the graph, i. e., the signal variation. Corollary 2.1 introduces a normalized measure of signal variation, called the Rayleigh quotient $\mathcal{R}_{\mathfrak{L}}(x)$. Intuitively, the smaller $\mathcal{R}_{\mathfrak{L}}(x)$ is the smoother x is and vice versa. According to the corollary:

- The variation of basis $\{\phi_k\}$ is an increasing function of k.
- The eigenvectors are the signals with minimal variation in each subspace $\Phi_k^{\perp}$. That is because, amongst all x which belong in some subspace $\Phi_k^{\perp}$ of $\mathbb{R}^n$, the signal with minimal variation is $x = \phi_k$.

We can further characterize the variation of eigenvectors, using the concept of discrete nodal domains. As Biyikoglu et al. [14] point out, discrete nodal domains are the analogues of the well known nodal domains appearing in PDE[4] literature. In that context, the nodal sets $\{x : f(x) = 0\}$ are the points $x \in \mathbb{R}^n$ where the eigenfunction f vanishes. Furthermore, the nodal domains of an eigenfunction are the connected components $x : f(x) \neq 0$ bounded by the nodal sets.

To understand the concept of discrete nodal domains, i. e., the nodal domains of graphs, recall that each eigenvector assigns a positive or negative value to each node in a graph. The second eigenvector of the path graph for example, seen in Figure 2.1, assigns a negative value to the two leftmost nodes, a positive value to the two rightmost nodes, and a zero to the node in the middle.

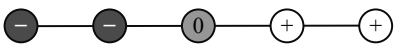


Figure 2.1: The second eigenvector of a path graph has two nodal domains, corresponding to the induced subgraphs with negative (left) and positive (right) values.

Discrete nodal domains –also called *sign graphs*– induce a partition of the graph into maximal induced subgraphs on which a function does not change its sign. In other words, consider a graph where the real values of nodes are mapped to their positive or negative signs, then group connected nodes with the same sign into subgraphs. The number of subgraphs represent the number of nodal domains. Intuitively, the larger the number of nodal domains, the higher the variance of a function. Based on whether the subgraph also includes nodes with zero value, nodal domains are further characterized as *weak* or *strong*. The discrete nodal domain

[4](P)arallel (D)ifferential (E)quations.

theorem establishes an upper bound of the number of weak nodal domains $\mathfrak{W}(\phi)$ and strong nodal domains $\mathfrak{S}(\phi)$ of an eigenvector ϕ of a generalized Laplacian.

Theorem 2.1 (Discrete Nodal Domain Theorem, Davies et al. [30, 48, 14])**.** *Let $\mathfrak{L}$ be a generalized Laplacian of a connected graph with n nodes. Then any eigenvector ϕ_k corresponding to the k-th eigenvalue λ_k with multiplicity r has at most k weak nodal domains and $k + r - 1$ strong nodal domains:*

$$\mathfrak{W}(\phi_k) \leq k \quad \textit{and} \quad \mathfrak{S}(\phi_k) \leq k + r - 1. \tag{2.10}$$

Based on the work Berkolaiko [13], Xu and Yau recently derived a matching lower bound:

Theorem 2.2 (Theorem 1.3 [124])**.** *Let $\mathfrak{L}$ be a generalized Laplacian of a connected graph with n nodes. Let λ_k be the k-th eigenvalue with multiplicity r and ϕ_k the eigenvector corresponsing to λ_k, which is zero on exactly z nodes. Then*

$$\mathfrak{S}(\phi_k) \geq k + r - 1 - l - z, \tag{2.11}$$

where $l = m - n + 1$ is the minimal number of edges that need to be removed from $\mathcal{G}$ to turn it into a tree.

By bounding the number of nodal domains of an eigenvector, the above theorems provide an intuitive characterization of the eigenvector's variation. An important consequence is that the bound is strictly monotonic on k. This matches our intuition that eigenvectors of higher order exhibit higher variation.

2.2.3. Common Graph Laplacians

Though in practice one may use the spectrum of any generalized Laplacian to study a signal, in most cases the analysis focuses on a small set of Laplacians, namely the *discrete* and *normalized* Laplacians, as well as the *random walk* matrices.

The discrete graph Laplacian (L). The simplest is the discrete Laplacian $L \triangleq D - A$, where D is the diagonal degree matrix with $D_{ii} = \deg(u_i)$ and A is the graph adjacency matrix. As seen by its expanded form,

$$(Lx)(u_i) = \sum_{u_j \sim u_i} (x(u_i) - x(u_j)) \tag{2.12}$$

the Laplacian computes the variation of x in a uniform manner. All edges are considered equally important, $\mathfrak{L}_{ij} = -1$, and no extra significance is given to node values, $p(u_i) = 0$. Because of its deep connections to PDE[5], the discrete Laplacian is very popular for many important network types, such as complex networks [114] and Riemannian manifolds [25]. Nevertheless, with the exception of Chapter 4, we will not use it in this thesis.

The normalized graph Laplacian ($\mathcal{L}$). A second, very important Laplacian is Chung's [26] normalized graph Laplacian $\mathcal{L} \triangleq D^{-1/2}LD^{-1/2}$. As the name suggests, the expanded form of $\mathcal{L}$ is normalized by the node degree

$$\begin{aligned}(\mathcal{L}x)(u_i) &= \sum_{u_j \sim u_i} \frac{1}{\sqrt{\deg(u_i)}} \left(\frac{x(u_i)}{\sqrt{\deg(u_i)}} - \frac{x(u_j)}{\sqrt{\deg(u_j)}} \right) \\ &= \sum_{u_j \sim u_i} \frac{x(u_i) - x(u_j)}{\sqrt{\deg(u_i)\deg(u_j)}} + \left(1 - \sum_{u_j \sim u_i} \frac{1}{\sqrt{\deg(u_i)\deg(u_j)}} \right) x(u_i).\end{aligned} \tag{2.13}$$

Matrix $\mathcal{L}$ therefore weighs each edge according to the reciprocal of the square-roots of the adjacent node degrees,

$$\mathfrak{L}_{ij} = -\frac{1}{\sqrt{\deg(u_i)\deg(u_j)}} \tag{2.14}$$

and gives an extra significance of $p(u_i) = 1 + \sum_{u_j \sim u_i} \mathfrak{L}_{ij}$ to each node.

Though an extensive review is beyond the scope of this thesis, as $\mathcal{L}$ appears recurrently throughout this thesis, it is useful to establish some properties of its spectrum. For a more in-depth treatment, the reader should refer to the excellent books by Fan Chung [26] and Biyikoglu et al. [14].

Because it is a symmetric matrix, matrix $\mathcal{L}$ has -at most- n distinct eigenvalues, all of which are real. It is common to sort the eigenvalues in ascending order:

$$0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n < 2. \tag{2.15}$$

As we can see, the eigenvalues are contained in $[0, 2)$, with the smallest eigenvalue –commonly referred to as the *harmonic* or *trivial* eigenvalue– always equal to zero. The corresponding harmonic eigenvector ϕ_1 is a vector equal to $D^{1/2}\mathbb{1}$, where $\mathbb{1}$ is the all-ones vector. The fact that the slowest varying eigenvector is not a constant, but a graph dependent function, is the main criticism towards using $\mathcal{L}$. Nevertheless,

[5] $-L$ is the finite difference approximation to the Laplacian operator.

similar to the first term of the Fourier transform, removing the 1-st signal component –or DC-offset– from the inverse spectral transform results in a signal centered around zero.

Unless mentioned otherwise, in the following we use symbols λ and ϕ to refer to the eigenvalues $\lambda(\mathcal{L})$ and the eigenvectors $\phi(\mathcal{L})$, respectively.

Random-walk matrices (T and P). The *random walk matrix* $T \triangleq AD^{-1}$, is the transition matrix of a unbiased random walk on the graph. Let x be the current probability distribution of a particle moving randomly. At the next step, the distribution is

$$(Tx)(u_i) = \sum_{u_j \sim u_i} \frac{x(u_j)}{\deg(u_j)}. \tag{2.16}$$

Since T is not Hermitian, it is not strictly a generalized Laplacian. Nevertheless, T is similar to a scalar perturbation of the normalized Laplacian, $T = D^{1/2}(I - \mathcal{L})D^{-1/2}$. It therefore has real eigenvalues contained in $(-1, 1]$ and $\lambda_k(T) = 1 - \lambda_k(\mathcal{L})$. We denote T's left and right eigenvectors as $\phi^{-1}(T)$ and $\phi(T)$, respectively.

Last, we often use the *random walk normalized Laplacian* $P \triangleq D^{-1}A = T^\top$. Because it computes the average at each node,

$$(Px)(u_i) = \sum_{u_j \sim u_i} \frac{x(u_j)}{\deg(u_i)}, \tag{2.17}$$

P is also referred to as the *consensus matrix*. It is easy to see that $P = D^{-1/2}(I - \mathcal{L})D^{1/2}$. As such, P also has real eigenvalues $\mu_k = \lambda_k(P) = 1 - \lambda_k(\mathcal{L})$. We denote P's left and right eigenvectors as ϑ^{-1} and ϑ, respectively.

To avoid confusion, we have to note that the use of T and P does not match the wikipedia convention [121].

2.3. Linear Graph Filters

Having established that the spectral transform describes graph signals in terms of their variation, a second question arises: *can we devise algorithms that filter a signal by attenuating its signal components according to their variation?* Well, yes! In the following, we establish the theoretical properties of graph filters and examine how they can be computed distributedly in a network.

2.3.1. The Graph Filter

We start with the definition of a graph filter.

Definition 2.3 (Graph Filter)**.** *Let λ_k and ϕ_k the eigenvalues and eigenvectors of a generalized graph Laplacian matrix $\mathfrak{L}$. A* graph filter $\mathbf{F}$ *is a linear operator that acts upon a signal x by amplifying or attenuating its spectral coefficients $\hat{x}_k$,*

$$y = \mathbf{F}x = \sum_{k=1}^{n} r(\lambda_k)\hat{x}_k\phi_k. \tag{2.18}$$

We say that $\mathbf{F}$ is afforded by (the spectrum of) $\mathfrak{L}$ because the two matrices share the same basis.

Let us examine this definition to provide an initial intuition on how a graph filter operates, from a spectral viewpoint. Analogously to the *frequency response* of a classical filter, the *spectral response*[6] $r : [0, 2) \to \mathbb{R}$ expresses how much the filter attenuates or amplifies the signal in each direction ϕ_k of the decomposition basis:

$$\frac{\langle \mathbf{F}x, \phi_k \rangle}{\langle x, \phi_k \rangle} = \frac{r(\lambda_k)\hat{x}_k}{\hat{x}_k} = r(\lambda_k). \tag{2.19}$$

The spectral response therefore acts as a weight that multiplies the spectral coefficients $\hat{x}_k$. The amplification of the k-th signal component is determined by the spectral response of the filter and the corresponding eigenvalue λ_k.

A natural question is to ask, can we construct a filter with arbitrary response r? The answer is yes, but at high cost. A filter constructed as

$$\mathbf{F} = \sum_{k=1}^{n} r(\lambda_k)\phi_k\phi_k^\top. \tag{2.20}$$

has a spectral response of precisely r. Nevertheless, such a construction necessitates the knowledge of the complete graph spectrum and is impractical for large graphs—the complexity is cubic to the number of nodes. A large portion of this thesis discusses (distributed) methods to approximate graph filters with specific response types.

The heat kernel. The most prominent example of a graph filter is the *heat kernel* [65, 129], i. e., the solution of the heat equation, paused after t steps. Since in our model the computation proceeds in discrete steps (or rounds), the heat kernel

[6] In general, the domain of the spectral response is the smallest interval that contains the eigenvalues of the generalized Laplacian in use. Here we use the normalized Laplacian $\mathcal{L}$.

of a signal x is

$$\begin{aligned}\mathbf{H}_t x &= (I - \mathcal{L})^t x \\ &= \sum_{k=1}^{n} (1 - \lambda_k)^t \hat{x}_k \phi_k,\end{aligned} \tag{2.21}$$

which is the discrete version of the usual definition $\mathbf{H}_t = e^{-t\mathcal{L}}$.

The spectral response of $\mathbf{H}_t$,

$$r(\lambda; t) = (1 - \lambda)^t, \tag{2.22}$$

is a strictly decreasing function. The heat kernel therefore belongs to the class of *low-pass graph filters*. Low-pass filters attenuate high-order signal components, i. e., the faster varying components of a signal[7]. They therefore simplify (or smooth) the signal they filter. In particular, by controlling its parameters one tunes a low-pass filter to smooth a signal at different degrees. The heat kernel for example, is an all pass filter for $t = 0$ and becomes progressively more aggressive as t increases.

A note on terminology. To emphasize the parallelism with classical signal processing and the distributed nature of the algorithms, in this thesis we opt for the term "graph filter". However, in the machine learning literature, graph filters are often referred to as "graph kernels" [65]. Still, not all kernels are filters. Whereas kernels can generally have an arbitrary response, the spectral responses of graph filters follow the guidelines of classical filter design, e. g., they are low-pass, band-pass, and so on. To conform with literature conventions, the two terms are often used interchangeably—we say for example the "heat kernel", not the heat filter.

2.3.2. How a Graph Matters

There is one key differentiating factor between classical filters and graph filters: the graph topology. We split the study of the relation between the graph topology and the filtering algorithm into two research challenges. *The first challenge entails understanding how the graph matters.* That is, how do the topological properties of a graph affect filter behavior? To gain insight, we examine how the eigenvalues and eigenvectors of the Laplacian determine the behavior of a graph filter. The problem thus reduces to characterizing graph spectra. We review relevant results from the spectral graph theory literature and identify open issues. *The second entails finding*

[7] In classical filter design, pass-band filters are designed to follow strict specifications, such as cutoff and sampling frequency, pass-band ripple, and stop-band attenuation. Because of the complexity of filter design in graphs, this thesis adopts a more relaxed definition.

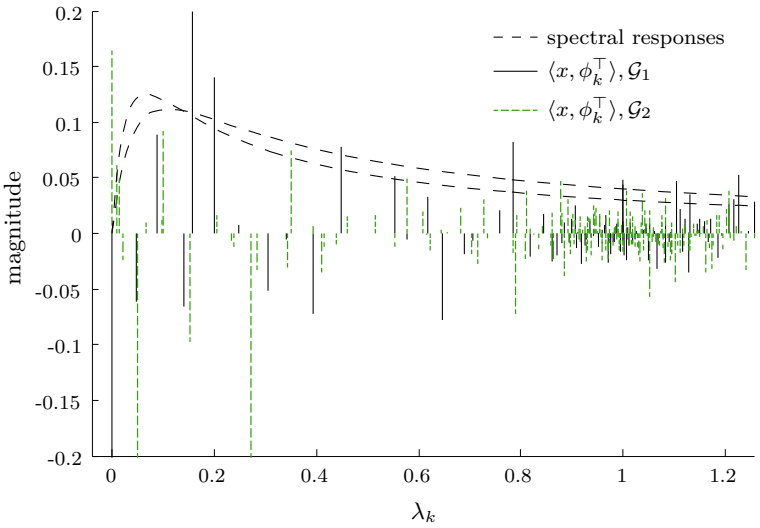


Figure 2.2: Signal components (vertical bars) of two random geometric graphs, with $n_1 = 100$ and $n_2 = 200$. The topology affects the decomposition of the signal (height of bars) and the point of the algorithm's response (dashed lines) each component corresponds to (horizontal position).

a graph-independent description of a filter. Either analytically or using simulation, it is practically impossible to evaluate the behavior of a filter across all possible topologies. In the following, we identify an analytical tool that allows us to analyze and compare filters in a graph-independent fashion.

How a graph matters. We proceed by example. Figure 2.2 illustrates the relation between graph and filter on two example random geometric graphs: graph $\mathcal{G}_1$ consisting of 100 nodes (shown in black) and graph $\mathcal{G}_2$ consisting of 200 nodes (shown in green). The figure entails two types of information. (*i*) The black and green vertical lines show the amplitude of the spectral coefficients $\hat{x}_k$ of a sample signal. (*ii*) The dashed curves represent two example spectral responses.

As seen in the figure, the graph influences filtering in two ways:

1. The basis functions ϕ_k onto which a signal is decomposed are characteristic of the graph topology. Simply put, different graphs decompose a signal in different ways. In Figure 2.2 this decomposition is captured (*i*) by the position of the vertical bars in the horizontal axis, which depend on their corresponding eigenvalues, and (*ii*) by their height, which depend on the projection $\hat{x}_k$ of x onto the k-th eigenvector.

2. How much each signal component $\hat{x}_k$ is affected depends on the corresponding eigenvalue λ_k. The spectral response filters signal components by multiplying each vertical bar with the value of the response function that lies on top of (i. e., that has the same eigenvalue). In Figure 2.2, $r(\lambda)$ favors the signal components that correspond to eigenvalues close to 0.08 and 0.1, respectively. All other signal components are attenuated.

Understanding how a graph matters therefore reduces to studying its eigenvalues and eigenvectors. By studying eigenvalues, we can understand how much a filter favors each signal component. Consider for example the effect of the second smallest eigenvalue (or Fiedler value) on a low-pass filter. It is well known that λ_2 measures how easily a graph is cut in two [21]: graphs with clear community structure (e. g., scale-free graphs [2]) have a small λ_2, where uniformly connected graphs (e. g., random graphs [37]) have large λ_2. This means that a signal resembling the Fiedler vector, i. e., having little mass adjacent to the cut edges, will be attenuated more when the graph is uniformly connected. Significant progress has also been made in bounding higher order eigenvalues of random graphs. Two results are of particular interest:

1. The eigenvalues $\lambda_k(\mathcal{L})$ of random power-law graphs follow the Wigner semicircle law [43, 120] when the minimum expected degree is relatively large [24, 23].

2. For constant-degree graphs either with a constant genus (e. g., planar graphs), or with a constant-sized forbidden minor, $\lambda_k(L) = O(k/n)$ [62].

A direct consequence is that, in a random power-law graph, fast changing signals are decomposed with high probability into signal components with corresponding eigenvalues close to one. To remove noise, it is therefore prudent to use filters with a response that is zero close to one, and one otherwise.

The analysis of filters however also hinges on eigenvectors. Unfortunately, very little is known about them [7, 14, 30]. A case is point is the Courant-Herrmann Conjecture (CHC), which attempts to bound the number of nodal domains of linear combinations of the first b Dirichlet eigenfunctions f_k of elliptic PDEs [28]. The conjecture asserts that the number of nodal domains of $\sum_{k=1}^{b} c_k f_k$ is smaller or equal to b. Consider the significance of this assertion. If CHC is true then one may design filters that bound the variation of a signal—the filter allows any signal to change sign at most b times. Unfortunately, the conjecture was recently proven false in the general case [47, 48]. Providing an alternative to CHC is an open question.

Algorithm 1 Computing $\mathbf{H}_\tau x$ in the synchronous model (running on each node u_i).

Require: Unique identifier u_i.

1: $t \leftarrow 0$, $y(u_i) \leftarrow x(u_i)$ ▷ Initialization.
2: **while** $t \leq \tau$ **do**
3: $t \leftarrow t + 1$
4: $y(u_i) \leftarrow y(u_i) - \sum_{j=1}^{n} \mathcal{L}_{ij}\, y(u_j)$ ▷ Compute matrix power locally.
5: **end while**

How a graph does not matter. Observe that both spectral responses (dashed curves) are independent of the graph. As such, the spectral response provides a very convenient way of studying and comparing the properties of filters, without having to consider all possible topologies. Indeed, in Chapter 4 we will use the spectral response to compare the efficiency of filters in terms of their resilience to noise and their detection resolution.

2.3.3. Distributed Graph Filters

The research community has so far focused on designing graph filters for various settings [110]. This thesis takes a different perspective: We examine how efficiently can a wireless network distributedly filter a signal.

To illustrate the challenge, let us consider the most famous low-pass filter—the heat kernel. As shown previously, $\mathbf{H}_t$ is just the normalized Laplacian raised to some power. This suggests that a very simple distributed algorithm can be used—see Algorithm 1. It is obvious that the algorithm is local. It terminates after exactly $\tau = O(1)$ rounds and exchanges $2\tau m = O(m)$ messages in total. Nevertheless, it has a number of important drawbacks.

The algorithm terminates correctly if:

- all nodes initiate computation simultaneously,
- all nodes iterate synchronously,
- the network stays invariant throughout the computation, and
- the signal stays invariant throughout the computation.

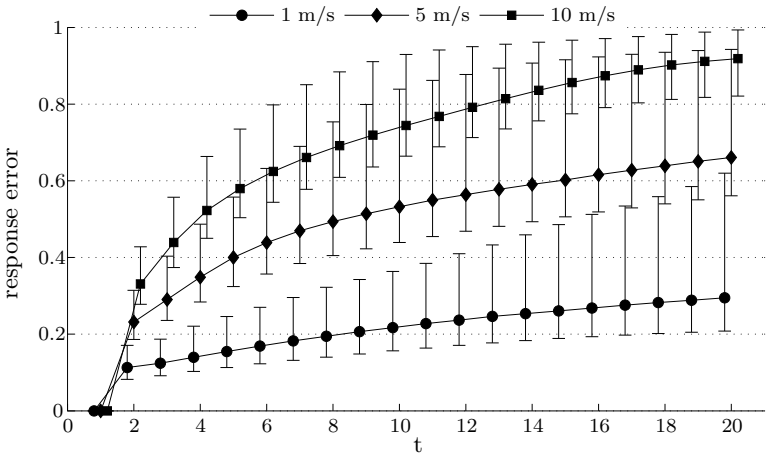


Figure 2.3: The figure illustrates the devastating effect of mobility on the heat kernel $\mathbf{H}_t$ (t is the smoothing parameter) for three different mobility speeds. Each errorbar depicts the 15th, 50th, and 84th percentiles of the normalized difference between $\mathbf{H}_t$'s response computed distributedly and centrally. To improve visibility, we introduced a small horizontal offset.

Even ignoring the overhead that has to be payed to ensure that the network stays synchronized (see Section 2.1.2), Algorithm 1 is very inefficient in the presence of dynamics: when the signal and/or the topology change over time, the computation becomes invalid. To illustrate the impact of mobility on the computation accuracy, we conducted an experiment in which the algorithm was run distributedly by a mobile network. In our experiment, the nodes moved according to the random waypoint model in a confined area of 230 meters, with a speed that varied between 1 and 10 m/s. Figure 2.3 shows the error $\|r(\lambda,t) - r^*(\lambda,t)\| \,/\, \|r^*(\lambda,t)\|$ in $\mathbf{H}_t$'s response when run distributedly ($r(\lambda,t)$) and centrally ($r^*(\lambda,t)$). Especially for large t, the error grows significantly. This motivates us to search for alternative (asynchronous) algorithms which are not only local, but can also track dynamics. Finding such filters forms one of the central contributions of this thesis.

3

Low-Pass Graph Filters

Distributed Aggregation and Search

"Don't Panic!"

—Douglas Adams,
The Hitchhiker's Guide to the Galaxy

This chapter proposes a novel distributed low-pass graph filter, called the *potential kernel*. Contrary to the heat kernel, the proposed filter is asynchronous and can track dynamics, such as time-varying signals and mobility. Our study provides two additional contributions: First, we characterize convergence in static and dynamic networks. We show that the potential kernel converges linearly and, as such, that it reaches ε-close to the steady state in $O(1)$ time. Our analysis captures the impact of key aspects of sensor networks, such as the degree distribution, the addition and deletion of links and nodes (churn), as well as the change of information over time. Second, we study how low-pass graph filters simplify the structure of a signal. The simplification has important implications, for example it increases

Parts of this chapter have been published in IPSN'13, April 2013, Philadelphia, Pennsylvania, USA [82] and in FOMC'12, July 2012, Madeira, Portugal [80].

the search scope of greedy search methods by eliminating local extrema. Simulations and experimental evaluation show that our main findings hold under realistic conditions, with unstable links and message loss.

3.1. Introduction

Wireless networks are often used to monitor network signals and to extract information from them. Such signals may pertain to the physical world, e. g., measuring climate conditions and sensing human activities, or to the internal operation of the network, e. g., measuring traffic congestion. While a network signal provides us with an accurate description of the information in close proximity of each individual node, we are often interested in its value over a larger area. Based on the scope used one obtains a whole spectrum of aggregates. The extremes of this spectrum can be characterized as on the one end only considering the local information and on the other end as averaging all information in the network.

Hitherto, two types of filters have been used for aggregating information in the vicinity of nodes. The first type exploits the (possible) spatial correlation of signals, and requires nodes to aggregate information at a decaying function of distance [45, 63, 101]. Spatial filters require location information –information which is not always available in practice– and are mainly aimed at *physical* signals with strong spatial correlations, such as temperature. For this reason, spatial filters do not perform well when the monitored signal depends more on the network topology than on the inter-node distances, such as for example the traffic-load in load-balanced routing. The second type of filter overcomes the limitations of the first by performing some form of constrained averaging; nodes for example aggregate all values within their k-hop vicinity or iteratively average neighbor values (heat kernel [65, 129]). Even though *graph filters* are indeed location-independent, state-of-the-art graph filters also suffer from drawbacks: constrained averaging algorithms either do not take into account the relative significance of information (k-hop average) or require synchronous computation and cannot tolerate dynamics (heat kernel).

3.1.1. Contributions

Within the context of graph filters, this chapter provides three main contributions.

Contribution 1. *We propose an asynchronous distributed low-pass graph filter, called the potential kernel—Section 3.2.* We introduce a subtle yet important change

to traditional constrained averaging algorithms. Similar to the heat kernel, the potential kernel requires no location information, it is simple, decentralized, and sensitive to the relative significance of information, as well as to the network topology. Yet, it is also asynchronous and can track dynamics.

Contribution 2. *We perform convergence analysis for both static and dynamic signals and networks—Sections 3.4 and 3.5.* Our study provides the following findings:

- The potential kernel reaches ε-close to the steady state in $O(1)$ rounds in the synchronous model (linear convergence). It is therefore approximately local.
- The convergence is faster in dense networks, especially when the degree distribution exhibits small variance.
- A critical aggregation scope (parameterization) exists under which network and signal dynamics hardly matter.

In practice, these findings provide some *dos and don'ts* for practical deployments. Avoid concave shapes (dents), void regions and radio transceivers with high sensitivity variance because they increase degree heterogeneity (irregularity). Adding nodes to a deployment helps more if they are uniformly distributed. Prior to a deployment, analyze the dynamics of the network, which might be caused by the movement of people, the failure/addition of nodes, etc.; if the dynamics are high, it is better to avoid using very wide aggregation scopes. If the dynamics are low, the extent of the aggregation scope does not play a major role, i. e., feel free to explore the trade-off between amount-of-information and message complexity.

Contribution 3. *We study how a low-pass graph filter simpifies the structures of a signal—Section 3.3.* To understand the practical importance of this result, consider a noisy signal with frequent peaks formed by nodes with abnormally high (or low) values with respect to their neighbors. In effect, this means that a signal can be smoothened to any desired level: the more aggressive the filter, the smoother the landscape. In previous work it was observed experimentally that, beyond a point, the continuous elimination of high-order eigenvectors leads to the elimination of local extrema, until only a single maxima is present (unimodality) [80]. Nevertheless, we show analytically that, in the general case, no graph filter exists that renders any signal unimodal.

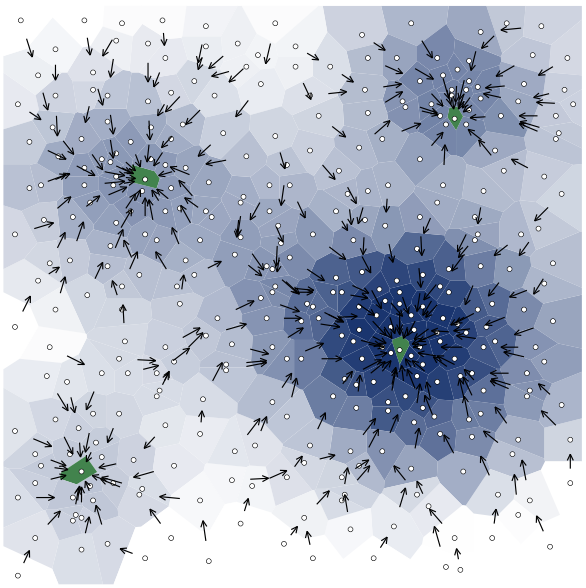


Figure 3.1: An information potential field which considers node degree. Nodes within green cells have the maximum value – highest relative degree – amongst their neighbors. The arrows reveal the potential gradient.

Section 3.6 validates some of our analysis by simulations and experiments on a 100+ node testbed.

3.1.2. Applications

The quint-essential application of low-pass graph filters is the removal of small signal perturbations, such as noise. Nevertheless, due to their ability to eliminate local extrema, low-pass filters have found a second use: the facilitation of greedy search methods [103].

Greedy routing. In greedy routing, nodes forward information greedily on the gradient of potential fields—see Figure 3.1. Potential fields that are not virtually constructed [119], but pertain to a monitored phenomenon, are called *information potentials*. In these scenarios, the presence of local maxima is undesirable as it prevents packets from arriving at the intended destination. A low-pass filter deteriorates this issue by eliminating local maxima (Contribution 3). It is important to remark that the work of Lin et al. [71] is also not affected by local extrema. However, their approach, which is based on harmonic fields, relies on a larger set of assumptions.

A detailed discussion of the similarities and differences of our approach with this and other studies is presented in Section 3.7.

In a more general context, information potentials (*i*) facilitate the discovery of areas of high (low) level of information, and (*ii*) provide mobile entities an efficient way of navigating towards these areas. The mobile entities can be packets, people, robots, cars, etc. In our previous work, we showed through simulations how our algorithm can be used for rendezvous coordination, where a swarm of mobile entities identify the nodes with the largest potential, and the swarm moves towards their closest high-potential node in real time [80]. Below, we describe two applications that can benefit from our work.

Crowd management. Our interest in information potential fields was sparked by their ability to support the management of large crowds in open air festivals. In these type of festivals, crowds of hundreds of thousands gather in city centers within confined spaces. For safety reasons, the crowds should not exceed densities above a given margin[1]. As part of two larger projects involving several institutions (D^2S^2 and EWiDS), we focus on the facilitation of crowd management through self-monitoring. We aim at providing attendees with coin-size wearable devices, each equipped with sensors, actuators (light), and a wireless transceiver. In this way, information potentials will be able to monitor the density of people in real time and warn attendees if their surrounding density exceeds safe levels. The attendees will also have access to the gradients formed by the potentials to move to areas with lower densities.

Traffic management. Parking management is a highly important topic in public policy. As reported by Shou [108], on average 30% of traffic in downtown areas is due to vehicles looking for parking. Cities such as San Francisco and London are installing wireless sensors on individual parking spots that report information to a central server to help drivers in finding available parking[2]. Information potentials are a low-cost efficient distributed alternative for the current centralized methods. Neighbouring meters could exchange their occupancy information and guide cars towards free parking areas by communicating with the drivers' smartphones via bluetooth. Information potentials have the added advantage of *aggregating* the in-

[1] In 2010, a crowd rush in a popular electronic dance festival in Germany ended up with 21 deaths and more than 500 injured people.

[2] The system for San Francisco is SFpark (`spark.org`) and for London is the Bay Sensor Technology (`www.westminster.gov.uk/services/transportandstreets/parking/bay-sensor-technology`)

formation of nearby free parking spots into "more/less crowded" regions, as opposed to tracking individual spaces.

Overall, we believe that distributed graph filters have an important role to play in the future of Smart Cities. As more and more wireless devices are embedded in our daily surroundings and more data is harvested from them, a central problem will be how to guide the various mobile entities towards the areas that they are mostly interested in. This work describes an alternative that is simple, decentralized, and generic enough such that it can be shaped to suit the needs of the application at hand.

3.2. The Potential Kernel

This section focuses on the algorithmic aspects of the potential kernel. It describes the distributed algorithm and explains how the kernel captures information with respect to the topology of the network.

3.2.1. Algorithm

In a nutshell, the potential kernel is an algorithm that maps local information to values more meaningful within the global network context. For instance, the local information may be the traffic load of the node, and the potential could be a relative value that states how high (or low) this traffic is with respect to other areas in the network—to facilitate, for instance, load balanced routing.

The computation of the potential kernel –given by Algorithm 2– is iterative. For the duration of each round, nodes $u_i \in \mathcal{V}$ exchange their value $y(u_i)$ with their neighbors (lines 3 and 6). Since nodes do not exchange $x(u_i)$, they affect their neighbors indirectly by changing the neighborhood average. At the end of a round, nodes update their value to the weighted sum of their signal value $x(u_i)$ and of the average over the most recent neighbor values, including their own (line 9). Intuitively, nodes behave like anchors, each pulling the neighborhood average towards its own $x(u_i)$ with a force that is proportional to the difference between $x(u_i)$ and the average. The force also depends on a parameter which we call *inhibiting factor*, and lies in $0 < \varphi \leq 1$. In the final step (line 10), all received values are discarded and the round ends. The algorithm converges when all forces are balanced. Furthermore, as we show in Section 3.2.2, the steady state (the value $y(u_i)$ that u_i converges to) is independent of initialization (the value $y(u_i)$ that u_i uses at the first round). Note that the algorithm includes no termination condition; it runs indefinitely, continuously

Algorithm 2 Potential kernel running on node u_i.

Require: Parameter $\varphi \in (0, 1]$ and unique identifier u_i.

1: $y(u_i) \leftarrow x(u_i)$, $\mathcal{S}(u_i) \leftarrow \{y(u_i)\}$ ▷ Initialization.

2: **event** OnTransmit

3: broadcast $\{u_i, y(u_i)\}$ to all neighbors ▷ Transmit to all neighbors.

4: **end event**

5: **event** OnReceive$(u_j, y(u_j))$

6: $\mathcal{S}(u_i) \leftarrow \mathcal{S}(u_i) \cup \{y(u_j)\}$ ▷ Keep latest neighbor value.

7: **end event**

8: **event** OnRoundEnd

9: $y(u_i) \leftarrow (1 - \varphi) \sum_{u_j \in \mathcal{S}(i)} \frac{y(u_j)}{|\mathcal{S}(u_i)|} + \varphi\, x(u_i)$ ▷ Update value.

10: $\mathcal{S}(u_i) \leftarrow \{y(u_i)\}$ ▷ Clear state.

11: **end event**

adapting to any network or information dynamics. If no dynamics are expected, termination is decided locally by comparing the difference of $y(u_i)$ at consecutive rounds against some error threshold.

The reuse of the information $x(u_i)$ throughout the computation (line 9) differentiates our work from: (*i*) the *average consensus*, in which y is initialized to x and, at each round t, nodes simply average their values, $y^{(t+1)} = Py^{(t)}$ (see Section 2.3.3), as well as from (*ii*) *broadcast consensus* [5], in which each node u_i computes a weighted average after every received broadcast from $u_j \sim u_i$, $y^{(0)}(u_i) = x(u_i)$ and $y^{(t+1)}(u_i) = \gamma y^{(t)}(u_i) + (1 - \gamma)y^{(t)}(u_j)$. As we show in the following, this simple alteration of average consensus gives rise to very interesting properties which relate to affinity spaces (Section 3.2.2) and the spectrum of the graph Laplacian (Section 3.3), and increases the algorithm's resilience to message loss and asynchrony (Section 3.6).

3.2.2. Algorithmic Analysis

We will now describe how this simple distributed algorithm captures the topology of the network. For the sake of clarity and conciseness, our analysis assumes that (*i*) nodes operate in synchronous rounds, (*ii*) at the end of which nodes have received at least one message from each of their neighbors. In Section 3.6 we show that, *in practice, neither assumption is necessary for the correct and timely operation of*

the algorithm. We will also assume, for now, that the signal is time-invariant. This assumption will be lifted in Section 3.4.2, which studies convergence for time-varying signals. Formula (3.1) rewrites Algorithm 2 in an iterative matrix form, where $y^{(t)}$ is the y vector after t rounds using the averaging matrix P.

$$y^{(t+1)} = (1-\varphi)Py^{(t)} + \varphi\, x \tag{3.1}$$

At round t, the potential $y^{(t)}$ is given by

$$y^{(t)} = (\psi P)^t y^{(0)} + \varphi \sum_{k=0}^{t-1} (\psi P)^k x, \tag{3.2}$$

where, for brevity, we set $\psi = 1-\varphi$. When t grows to infinity, the potential kernel $\mathbf{P}_\varphi$ takes the following closed-form expression:

$$y = \mathbf{P}_\varphi x = \lim_{t\to\infty} y^{(t)} = \varphi \sum_{k=0}^{\infty} (\psi P)^k x \tag{3.3}$$

Note that the dynamical system described in (3.2) is stable. As t grows larger, $(\psi P)^t$ approaches asymptotically zero because $\psi < 1$ and P^t is a row-stochastic matrix. The decay of $(\psi P)^t$ removes any influence of the initial state $y^{(0)}$ on the potential y. In Sections 3.4 and 3.5 we analyze the convergence rate in detail; the rest of this section provides more insights on the kernel itself.

Affinity spaces. In contrast to definitions of affinity that result from other metrics of distance, such as euclidean distance [63], the potential kernel aggregates information based on a type of affinity that is very sensitive to the topology of a network. As we will observe, this topological sensitivity is achieved because nodes diffuse information in a sort of "random-walk" manner.

Formula (3.3) expresses the kernel as an infinite sum that, at each round k, changes information x with a weighting factor of $(\psi P)^k$. Element P^k_{ij} expresses the probability of a randomly moving particle starting from node u_i and reaching node u_j in k steps. The better the connectivity between u_i and u_j and the shorter the path, the higher the probability. In other words, instead of averaging the information within the k-th range of a node, our method assigns higher significance to the information residing in nearby nodes (close connectivity) and in nodes with higher centrality (better connectivity). In most graphs, centrality is an important metric that captures the importance of the information.

The inhibiting factor φ determines the aggregation scope. Let us consider the two extremes of the aggregation spectrum:

1. When $\varphi = 1$, there is no exchange of information and hence the network topology plays no role. The potential kernel is the identity matrix and $y = x$.

2. When $\varphi \to 0^+$, the aggregation scope is global. The potential kernel reduces to average consensus[3]. As $t \to \infty$, y gets closer to $\mathbb{1}\pi^\top x$, where $\mathbb{1}$ is the $n \times 1$ vector with all elements equal to one and π is the stationary distribution of the random walk.

Within these extremes, the network has ample flexibility to shape information potentials according to the requirements of the application. In Section 3.3, we will analyze this characteristic in more detail and its impact on greedy search techniques. Note that, since the geometric series ψ^k converges to $1/\varphi$, the multiplication with φ serves as a normalization.

It is important to highlight that while the analysis considers the global connectivity matrix P, *the algorithm only requires communication with 1-hop neighbors.* The advantage of our simple and distributed algorithm is that it entails such global behavior inherently.

3.3. Spectral Analysis

Thus far we have analyzed the impact of the topology through the use of affinity spaces. In this section we approach the algorithmic analysis from a spectral perspective. We show that the proposed algorithm belongs to the class of low-pass graph filters, as specified in Section 2.3. We derive the filter's spectral response and we explain how, using the inhibiting factor φ, one reshapes a signal and eliminates its extrema.

3.3.1. Spectral Transform

Our main result ties the potential kernel to the spectrum of the normalized Laplacian.

Theorem 3.1. *Let (λ_k, ϕ_k) be the k-th eigenpairs of the normalized Laplacian $\mathcal{L}$ of a graph $\mathcal{G}$. For any signal x,*

$$y = \mathbf{P}_\varphi x = D^{-1/2} \sum_{k=1}^{n} \left(\frac{1-\varphi}{\varphi} \lambda_k + 1 \right)^{-1} \hat{z}_k \, \phi_k, \tag{3.4}$$

[3]Also referred to as a normalized Laplacian random walk.

where $\hat{z}_k = \langle D^{1/2}\, x, \phi_k \rangle$ are the spectral coefficients of the degree-normalized signal.

Proof. We exploit the spectral relations between the transition matrix P and the Laplacian $\mathcal{L}$. The reader can refer to the text by Biyikoglu et al. [14] for more details on the topic. Let μ and ϑ be an eigenvalue and eigenvector of matrix P, respectively. In connected graphs, P has a unique largest eigenvalue $\mu_1 = 1$ and all other eigenvalues $|\mu_k| < 1$, with $k = 2, \ldots, n$. Since $\psi < 1$, matrix ψP has eigenvalues that are strictly smaller than one and we re-write (3.3) as

$$y = \varphi\, (I - \psi P)^{-1} x. \tag{3.5}$$

As shown next, matrix $I - \psi P$ has the same eigenvectors as P and its eigenvalues are equal to $1 - \psi\mu_k$,

$$\begin{aligned} P\, \vartheta_k &= \mu_k\, \vartheta_k \\ (I - \psi P)\, \vartheta_k &= (1 - \psi\mu_k)\, \vartheta_k. \end{aligned} \tag{3.6}$$

It is well known that invertible matrices have the same eigenvectors as their inverse and eigenvalues that are the reciprocal of the eigenvalues of their inverse. Formula (3.5) can therefore be re-written through the spectral expansion of the inverse of $I - \psi P$ as

$$y = \sum_{k=1}^{n} \frac{\varphi}{1 - \psi\mu_k} \vartheta_k \vartheta_k^{-1}\, x, \tag{3.7}$$

where ϑ_k^{-1} is the k-th left eigenvector of P. From [14], we know that $\vartheta_k = D^{-1/2}\phi_k$, $\vartheta_k^{-1} = \phi_k^{\top} D^{1/2}$ and $1 - \lambda_k = \mu_k$. Substituting these equalities into (3.7) concludes our proof. ■

Theorem 3.1 asserts that the proposed algorithm is a graph-filter connected to the normalized graph Laplacian. Because the filter's spectral response

$$r(\lambda; \varphi) = \left(\frac{1 - \varphi}{\varphi} \lambda_k + 1 \right)^{-1}, \tag{3.8}$$

is a decreasing function with $0 \leq r(\lambda; \varphi) \leq 1$, the filter is *low-pass*. Small values of φ reduce the influence of high-order signal components, i. e., the components of x with higher variation. By fine-tuning φ, we can thus smoothen a signal and, as a consequence, reduce the number of its local extrema.

3.3.2. The Elimination of Local Extrema

By attenuating high-order signal components, a low-pass graph filter eliminates the local extrema of a signal. This is the reason why such filters are ideal for increasing the search scope of greedy search methods.

Unfortunately, an exact characterization of the number of extrema of a filtered signal is, to this point, intractable. We can however provide an upper bound on the number of extrema of each signal component $\hat{x}_k\phi_k$. The lemma interprets positive and negative nodal domains as the peaks and pits of eigenvectors. It then bounds the number of extrema by showing that a positive (negative) nodal domain has at most one maximum (minimum).

Lemma 3.1. *Let $\phi_k(\mathfrak{L})$ the k-th eigenvector of a generalized graph Laplacian matrix $\mathfrak{L}$ corresponding to eigenvalue λ_k with multiplicity r, and which is zero at exactly z nodes. The k-th signal component $\hat{x}_k\phi_k(\mathfrak{L})$ of a connected graph has at least $k + r - 1 - (m - n + z + 1)$ and at most $k + r - 1$ extrema.*

Proof. The proof proceeds by method of contradiction. An eigenvector of a generalized graph Laplacian cannot have a nonnegative local minimum or a non-positive local maximum [48, 51]. Without loss of generality, assume a negative strong nodal domain and suppose there are two local minima residing in the domain. This necessitates the existence of a negative local maximum between the two minima, which contradicts the first observation. Therefore, a single minimum (maximum) must exist at every negative (positive) nodal domain. Since the number of strong nodal domains of $\phi_k(\mathfrak{L})$ is bounded by Theorems 2.1 and 2.2, so are the number of extrema of $\phi_k(\mathfrak{L})$ multiplied by any real scalar. ■

Lemma 3.1 provides the first step towards bounding the number of extrema of a filtered signal. We now know that each signal component has at most $k + r - 1$ extrema. By attenuating the k-th signal component, a low-pass filter is therefore expected to eliminate at most $(k + r - 1)$-signal extrema. Nevertheless, it is yet unclear how to proceed. Most signals are linear combinations of many signal components. Bounding the number of extrema of such linear combinations is an open problem, strongly connected to the Courant-Hermann Conjecture [28, 47, 48].

We have to note that a filter changes a signal in more subtle ways than simply eliminating its extrema. As shown in 3.2.2, the value of a node depends on the values of all the nodes in the node's affinity space, as well as on the underlying connectivity. Therefore, it is possible that a node that is an extremum of y, but

not of x. For example, a unimodal signal (i. e., possessing a single extremum) will remain unimodal when filtered. The position of the extremum however will not necessarily be the same. The extremum of x is the node with the maximum value, which is not always equivalent to the node with the largest value in its surroundings. This phenomenon is captured in our evaluation.

3.3.3. The Unimodality Property

A logical implication of Lemma 3.1 is that, for any graph and signal, a critical value φ exists for which the potential becomes *unimodal*. The phenomenon is depicted graphically in Figure 3.4—for interpretation, see Section 3.6.2. Formally, a signal defined on a graph is unimodal if it has a single extremum and, for each node, at least one path to the extremum exists on which the values of the signal are monotonic. Obviously, the unimodality property is extremely useful for gradient search—in unimodal search spaces, greedy search cannot get stuck in local optima[4].

But does the unimodality property -always- hold? In the continuous setting, it has been shown that it does not [78]. By adapting their proof method, it is easy to provide an equivalent impossibility result for graph filters.

Corollary 3.1. *No graph filter exists which renders any signal unimodal.*

Proof. Lets denote by $\mathbf{F}$ the graph filter in consideration and $\phi_k(\mathbf{F})$ its k-th eigenvector. A simple signal exists that breaks $\mathbf{F}$'s unimodality. Choose $x = \phi_n(\mathbf{F})$. Under some sufficient conditions on $\mathcal{G}$, this is a signal with at least $2n-m-1$ nodal domains [13]. Clearly, when $n > 1 + m/2$, x is not unimodal. Nevertheless, being an eigenvector of $\mathbf{F}$, the filter only changes the signal's amplitude, not its nodal domains:

$$\mathbf{F}\phi_n(\mathbf{F}) = \lambda_n(\mathbf{F})\,\phi_n(\mathbf{F}).$$

Since the above holds independently of the filter in consideration, we conclude that the unimodalily property does not hold in the general case. ■

Though the above result is unfortunate, it has been observed experimentally that the unimodality property does hold when considering typical signals. Indeed, in Section 3.6 we find that signal $x(u_i) = \deg(u_i)$ becomes unimodal when filtered appropriately. In the continuous setting it has been shown that any non-negative functions with compact support eventually become unimodal when smoothed with

[4]The space of unimodal functions includes convex/concave functions.

the gaussian kernel [78]. We suspect that this is a property that also holds for low-pass graph filters.

We would like to conclude this section with some criticism. Though unimodality appears vital for greedy search, in practice it is not always desirable: To begin with, one has to pay in terms of convergence time for the increased search scope. In very large networks, the critical parameter that turns a signal unimodal often renders computation impractical. Moreover, despite its sub-optimality, greedy search in multimodal potentials yields more spatially relevant results—found extrema exhibit a higher spatial correlation to node location.

3.4. Convergence in Static Graphs

In this section we bound the rate of convergence in static graphs for invariant and time-varying signals. We limit our analysis to $\varphi \in (0, 1)$ as for $\varphi = 1$ the algorithm converges instantaneously. We will first present our theoretical analysis, and then, discuss its implications. In our study, we define the ℓ_2-distance of an information potential $y^{(t)}$ as $\varepsilon^{(t)} = \left\|y - y^{(t)}\right\|$, that is, its distance to the steady state.

3.4.1. Time-invariant Signals

We first consider the case that the signal stays constant over time and the graph is static. In the next sections, we will remove these constraints.

Theorem 3.2. *After t rounds, the ℓ_2-distance of dynamical system (3.1) to its steady state on a static graph $\mathcal{G}$ and signal x is bounded by $\varepsilon^{(t)} \leq e^{-\varphi t}(c^2 + c)\ \|x\|$, where $c = \frac{n}{2m}\Delta + (1 - \lambda_2)\sqrt{\frac{\Delta}{\delta}}$.*

Proof. The ℓ_2-distance at the t-th round is

$$\begin{aligned}
\varepsilon^{(t)} &= \left\|y - y^{(t)}\right\| \\
&= \left\| \varphi \sum_{k=0}^{\infty} (\psi P)^k x - \varphi \sum_{k=0}^{t-1} (\psi P)^k x - (\psi P)^t y^{(0)} \right\| \\
&= \left\| (\psi P)^t \Big(\varphi \sum_{k=0}^{\infty} (\psi P)^k x - y^{(0)} \Big) \right\|. \qquad (3.9)
\end{aligned}$$

Using a known bound on $P^t x$ [26], we get that for $t \geq 0$ and an arbitrary vector x,

$$\begin{aligned}\left\|(\psi P)^t x\right\| &\leq \left(\psi^t \frac{n}{2m}\Delta + (\psi(1-\lambda_2))^t \sqrt{\frac{\Delta}{\delta}}\right) \|x\| \\ &\leq \psi^t c \, \|x\| \,. \end{aligned} \tag{3.10}$$

Above, $(1-\lambda_2)$ is the second eigenvalue of the transition matrix P and $\sqrt{\Delta/\delta}$ measures the degree irregularity of $\mathcal{G}$. In the last step, $(\psi(1-\lambda_2))^t \leq \psi^t(1-\lambda_2)$ because $0 \leq \lambda_2 < 2$ and $0 \leq \psi < 1$. This loosens the error bound when $\lambda_2 < 1$, but allows to estimate the necessary number of rounds t until the algorithm converges ε-close to the steady state. One can achieve a tighter bound if an estimate of t is not required. Substituting Inequality (3.10) into Formula (3.9) we have that

$$\varepsilon^{(t)} \leq \psi^t c \left\| \varphi \sum_{k=0}^{\infty} (\psi P)^k x - y^{(0)} \right\| . \tag{3.11}$$

From Formula (3.2), we observe that the choice of the initial state $y^{(0)}$ is irrelevant to the steady state y. Nevertheless, a reasonable step is to set $y^{(0)} = x$. The normed difference in Inequality (3.11) then is simplified to

$$\begin{aligned}\left\| \varphi \sum_{k=0}^{\infty} (\psi P)^k x - x \right\| &\leq \varphi \sum_{k=0}^{\infty} \left\|(\psi P)^k x\right\| + \|x\| \\ &\leq (\varphi \sum_{k=0}^{\infty} \psi^k c + 1) \, \|x\| \\ &= (c+1) \, \|x\| \,. \end{aligned} \tag{3.12}$$

The substitution of (3.12) into (3.11)

$$\frac{\left\|y - y^{(t)}\right\|}{\|x\|} < \psi^t c \, (c+1) \leq e^{-\varphi t}(c^2 + c),$$

concludes our proof. ■

Since the ratio $\varepsilon^{(t)}/\varepsilon^{(t-1)}$ is constant, this type of convergence is commonly referred to as linear convergence. A direct consequence is that the necessary number of rounds until the algorithm converges ε-close to y (for convenience, $\|x\| = 1$) is given by

$$t \geq \varphi^{-1} \log\left(\frac{c^2 + c}{\varepsilon}\right) = O(1). \tag{3.13}$$

The algorithm therefore reaches ε-close to the steady state in constant time.

3.4.2. Time-varying Signals

In many practical applications, the signal changes over time. For example, a sensor network monitoring the presence of animals will change its measurements when the animal moves. We proceed to examine the behavior of such time-varying signals.

Lemma 3.2. *Let dynamical system* (3.1) *be at steady state y, when x changes to $\hat{x}$, with $\|\hat{x} - x\| \leq \delta_x$. After $\tau > 0$ steps, the ℓ_2-distance to the new steady state $\hat{y}$ is bounded by $\varepsilon^{(\tau)} \leq e^{-\varphi\tau} c\, \delta_x$.*

Proof. Without loss of generality, assuming that the change from x to $\hat{x}$ occurs at time t, the ℓ_2-distance to $\hat{y}$ after τ rounds is bounded by

$$\begin{aligned}
\varepsilon^{(\tau)} &= \left\| \hat{y} - y^{(t+\tau)} \right\| \\
&= \left\| \varphi \sum_{k=0}^{\infty} (\psi P)^k \hat{x} - \varphi \sum_{k=0}^{\tau-1} (\psi P)^k \hat{x} - (\psi P)^\tau y \right\| \\
&= \varphi \left\| \sum_{k=\tau}^{\infty} (\psi P)^k \hat{x} - \sum_{k=\tau}^{\infty} (\psi P)^k x \right\| \\
&\leq \varphi \sum_{k=\tau}^{\infty} \left\| (\psi P)^k (\hat{x} - x) \right\| \leq e^{-\varphi\tau} c\, \delta_x.
\end{aligned}$$

■

From Lemma 3.2, we have that the minimum τ for which the algorithm manages to converge ε-close to the new steady state $\hat{y}$ is

$$\tau \geq \varphi^{-1} \log\left(\frac{c\, \delta_x}{\varepsilon}\right). \tag{3.14}$$

3.4.3. Analysis Insights

Theorem 3.2 and Lemma 3.2 provide us with four important insights:

1. *The convergence error decreases exponentially.* The inhibiting factor determines the rate of convergence. Smaller values of φ aggregate the values over an exponentially larger subgraphs and as a consequence exhibit slower convergence.

2. *For $\varphi > 0$, the algorithm converges faster than average consensus.* The common ratio of the geometric series which upper bounds the convergence error

decreases from $1-\lambda_2$ (average consensus) to $(1-\varphi)(1-\lambda_2)$ (potential kernel). For $\varphi = 0$, the two algorithms are exactly the same.

3. *Signal dynamics proportionally increase the ℓ_2-error.* This effect should be taken into account when choosing the value of the inhibiting factor.

4. *Convergence is faster in dense graphs with degree distribution having small variance.* Constant c captures the influence of the network topology to the convergence. Through c we derive that convergence is faster for dense graphs, as well as for graphs with small node degree variations (quantified by Δ/δ, i. e., the ratio of the largest and the smallest node degrees).

3.5. Convergence in Dynamic Graphs

In this section, we study the algorithmic behavior in the context of graphs that change over time. We model graph dynamics as a sequence of edge and node operations: *Edge operations* describe the addition or deletion of edges between pairs of nodes. *Node operations* model nodes joining or leaving the graph. Through edge and node operations we capture a wide range of network dynamics, such as node and link failures, as well as the dynamics of open networks where the network is subject to mobility and churn. To ensure that the problem remains tractable, we assume time-invariant information.

We will first derive a bound on the convergence error given any change in the graph. The bound, which is stated in Theorem 3.3, is general enough to hold for any possible graph dynamics. On the down side, the bound depends on the specifics of the graph dynamics χ in question. We gain further insight by characterizing χ for edge and node operations in Lemmas 3.3 and 3.4, respectively.

Theorem 3.3. *Let dynamical system (3.1) be at steady state y on graph $\mathcal{G}$, when an adversary changes the graph to $\hat{\mathcal{G}}$, with $\left\|\hat{P} - P\right\| \leq \chi$. After $\tau > 0$ steps, the ℓ_2-distance to the steady state is bounded by*

$$\varepsilon^{(\tau)} \leq e^{-\varphi(\tau+1)}\hat{c}\left(\chi\frac{1-\varphi}{\varphi}\hat{c}\,c + \frac{\varphi}{1-\varphi}\right)\|x\|\,, \tag{3.15}$$

where c (respectively $\hat{c}$) quantifies the connectivity properties of $\mathcal{G}$ (respectively $\hat{\mathcal{G}}$).

Proof. Consider that, at time t, the algorithm has converged to a steady state y on a graph $\mathcal{G}$. An adversary then changes the graph to $\hat{\mathcal{G}}$. In the following we annotate

symbols that relate to the new graph $\hat{G}$ with a hat; as such $\hat{P}$ is the new transition matrix, $\hat{n}$ is the number of nodes in $\hat{\mathcal{V}}$, and so on. As in previous proofs, we capture convergence error through the ℓ_2-distance between the state after τ rounds and the new steady state $\hat{y}$. By substituting the analytic expression of y into Formula (3.9), we get

$$\varepsilon^{(\tau)} = \left\| \hat{y} - y^{(t+\tau)} \right\| = \left\| \varphi \, (\psi \hat{P})^{\tau} \left(\sum_{k=1}^{\infty} \psi^k \big(\hat{P}^k - P^k \big) \, x - x \right) \right\|. \tag{3.16}$$

Using Inequality (3.10) and after some algebraic manipulation we have that

$$\varepsilon^{(\tau)} \leq \psi^{\tau} \varphi \, \hat{c} \left(\left\| \sum_{k=1}^{\infty} (\psi \hat{P})^k x - \sum_{k=1}^{\infty} (\psi P)^k x \right\| + \|x\| \right) \tag{3.17}$$

$$\begin{aligned} &= \psi^{\tau} \varphi \, \hat{c} \left(\left\| (\psi \hat{P} - \psi P) \sum_{k=1}^{\infty} (\psi \hat{P})^k \sum_{l=k+1}^{\infty} (\psi P)^{l-k-1} x \right\| + \|x\| \right) \\ &\leq \psi^{\tau+1} \varphi \, \hat{c} \left(\chi \sum_{k=1}^{\infty} (\psi^k \hat{c}) \sum_{l=0}^{\infty} (\psi^l c) + \psi^{-1} \right) \|x\| \\ &\leq e^{-\varphi(\tau+1)} \hat{c} \left(\chi \, \frac{1-\varphi}{\varphi} \hat{c} \, c + \frac{\varphi}{1-\varphi} \right) \|x\|, \end{aligned} \tag{3.18}$$

which concludes our proof. ■

Solving for τ we find that the least number of rounds until $\varepsilon^{(\tau)} \leq \varepsilon$ is at most

$$\tau \geq \varphi^{-1} \log \left(\frac{\hat{c} \left(\chi \frac{1-\varphi}{\varphi} \hat{c} \, c + \varphi \right) \|x\|}{\varepsilon} \right) - 1. \tag{3.19}$$

Theorem 3.3 draws a relation between the influence of graph dynamics in convergence time and the value of φ. Depending on whether $(1-\varphi)/\varphi < \varphi/(1-\varphi)$ and thus $\varphi > 1/2$, the theorem distinguishes two regions of convergence:

1. For $\varphi > 1/2$ and as φ grows larger, the graph dynamics χ become irrelevant. The convergence becomes independent of the relation between G and $\hat{G}$; convergence depends solely on the new graph $\hat{G}$.

2. For $\varphi < 1/2$ and as φ gets closer to 0, the rate of convergence becomes slow enough such that the graph dynamics do matter. In this region, convergence depends heavily on the nature of edge operations performed.

We proceed to examine how edge and node operations influence convergence by computing χ for each case.

3.5.1. Edge Operations

An adversary adds or deletes edges $\mathcal{E}_+$ and $\mathcal{E}_-$, respectively to the graph, such that either $\hat{\mathcal{E}} = \mathcal{E} \cup \mathcal{E}_+$ (edge addition), or $\hat{\mathcal{E}} = \mathcal{E} \setminus \mathcal{E}_-$ (edge deletion). We also place the constraint that the symmetric edge set difference, $\hat{\mathcal{E}} \ominus \mathcal{E} = \mathcal{E}_+ \cup \mathcal{E}_-$, contains at most one edge (u_i, u_j) for each node u_i in $\mathcal{V}$. The constraint demands that the adversary performs at most one edge operation in the vicinity of each node. Multiple edge alterations on the same node are modeled as consecutive operations. As expected, self-loops cannot be deleted, that is $\deg(u_i) \geq 1$ for all u_i in $\mathcal{V}$. We prove the following bound.

Lemma 3.3. *For any graphs $\hat{\mathcal{G}}$, $\mathcal{G}$, with identical node sets, $\mathcal{V} = \hat{\mathcal{V}}$, and edge sets $\hat{\mathcal{E}} \neq \mathcal{E}$ that have at most one edge difference in the vicinity of each node, $|(u_i, u_j) \in \hat{\mathcal{E}} \ominus \mathcal{E}| \leq 1$ for all $u_i \in \mathcal{V}$,*

$$\chi = \frac{1}{\delta} + \frac{\sigma_{max}(A) + 1}{\min\limits_{u_i \in \mathcal{V}_\pm} \left\{ \hat{deg}(u_i)\, deg(u_i) \right\}}, \tag{3.20}$$

with $\mathcal{V}_\pm = \left\{ u_i \in \mathcal{V} : |(u_i, u_j) \in \hat{\mathcal{E}} \ominus \mathcal{E}| = 1 \text{ for some } u_j \in \mathcal{V} \right\}$ the set of nodes that have different neighbors in $\hat{\mathcal{G}}$ and $\mathcal{G}$, $\hat{deg}(u_i)$ and $deg(u_i)$ their respective densities in each graph, and $\sigma_{max}(A)$ the largest singular value of matrix A.

Proof. The adjacency matrix of $\hat{\mathcal{G}}$ can be written as

$$\hat{A} = A \;+ \sum_{(u_i,u_j)\in\mathcal{E}_+} \big(E_{ij} + E_{ji}\big) \;- \sum_{(u_i,u_j)\in\mathcal{E}_-} \big(E_{ij} + E_{ji}\big), \tag{3.21}$$

where matrix E_{ij} has only element (i, j) equal to one and the rest zero. The inverse degree matrix of $\hat{G}$ can in turn be written as

$$\hat{D}^{-1} = D^{-1} + \sum_{(u_i,u_j)\in\mathcal{E}_+} \big(a_i E_{ii} + a_j E_{jj}\big) \;- \sum_{(u_i,u_j)\in\mathcal{E}_-} \big(b_i E_{ii} + b_j E_{jj}\big), \tag{3.22}$$

where

$$a_i = \frac{1}{\deg(u_i)\,(\deg(u_i)+1)} \quad \text{and} \quad b_i = \frac{1}{\deg(u_i)\,(\deg(u_i)-1)}.$$

Matrix E_{ij} has the useful property of $\|E_{ij}\| = 1$ for all $u_i, u_j \in \mathcal{V}$. Due to $\hat{\mathcal{E}} \ominus \mathcal{E}$ containing at most one edge (u_i, u_j) for each node u_i in $\mathcal{V}$ and because $\hat{A} - A$ is a symmetric projection matrix,

$$\left\|\hat{A} - A\right\| = 1. \tag{3.23}$$

As in Lemma 3.3, $\mathcal{V}_\pm = \left\{u_i \in \mathcal{V} : |(u_i, u_j) \in \hat{\mathcal{E}} \ominus \mathcal{E}| = 1 \text{ for some } j \in \mathcal{V}\right\}$ is the set of nodes that were affected by an edge operation. Matrix $\hat{D}^{-1} - D^{-1}$ is diagonal and its norm is the maximum diagonal element in absolute value.

$$\left\|\hat{D}^{-1} - D^{-1}\right\| = \min_{i \in \mathcal{V}_\pm} \left\{\hat{\deg}(u_i) \deg(u_i)\right\}^{-1} \tag{3.24}$$

Observe that $\hat{\deg}(u_i) \deg(u_i) \geq \delta(\delta - 1) \geq 2$, for all $u_i \in \mathcal{V}_\pm$. The first equality is satisfied when one of the endpoints of a deleted edge was connected to the node with the minimum degree, while the second equality iff $\delta = 2$. We re-write the ℓ_2-distance between the two random walk matrices as

$$\begin{aligned}\left\|\hat{P} - P\right\| &= \left\|\hat{D}^{-1}\hat{A} - D^{-1}A\right\| \\ &\leq \left(\|D^{-1}\| + \left\|\hat{D}^{-1} - D^{-1}\right\|\right)\left\|\hat{A} - A\right\| + \left\|\hat{D}^{-1} - D^{-1}\right\| \|A\| \end{aligned} \tag{3.25}$$

The required bound is derived if we substitute (3.23) and (3.24) into $\left\|\hat{P} - P\right\|$.

$$\chi = \|D^{-1}\| + \frac{\|A\| + 1}{\min\limits_{u_i \in \mathcal{V}_\pm} \left\{\hat{\deg}(u_i) \deg(u_i)\right\}} = \frac{1}{\delta} + \frac{\sigma_{max}(A) + 1}{\min\limits_{u_i \in \mathcal{V}_\pm} \left\{\hat{\deg}(u_i) \deg(u_i)\right\}}, \tag{3.26}$$

where in the last step we exploited that $\|A\| = \sigma_{max}(A)$. ■

Let us reflect on the influence of edge operations on the convergence:

1. *Edge operations affect dense graphs to a greater extent.* That is due to $\sigma_{max}(A) \leq n$, with the equality satisfied for fully connected networks. While in sparse networks the effects of dynamics tend to be isolated, in dense networks there is a higher likelihood that any single change affects more nodes.

2. *Nevertheless, networks with few links mitigate the effects of network dynamics slower.* Our method compensates for edges added or deleted in well connected areas of the network (i. e., $\deg(u_i) >> \delta$ for $u_i \in \mathcal{V}_\pm$) faster than in areas where the network is sparsely connected. The effect is understood by the property of information to diffuse faster in dense than in sparse areas.

3. *Convergence also depends on the degree irregularity.* The algorithm exhibits the fastest convergence when running on regular networks.

4. *Last, even though edge operations affect multiple edges, the error bound is independent of the exact number of affected edges.* The error depends instead solely on the edge that connects to the least connected node.

3.5.2. Node Operations

Node operations are operations on the set $\mathcal{V}$ of graph nodes. In node additions (deletions), an adversary adds (deletes) nodes $\mathcal{V}_+$ ($\mathcal{V}_-$), such that $\hat{\mathcal{V}} = \mathcal{V} \cup \mathcal{V}_+$ ($\hat{\mathcal{V}} = \mathcal{V} \setminus \mathcal{V}_-$). We provide an upper bound of χ for the simultaneous addition and deletion of nodes.

Lemma 3.4. *Let $\hat{\mathcal{G}}$ be the graph that results from the addition and deletion of sets $\mathcal{V}_+$ and $\mathcal{V}_-$ from $\mathcal{G}$, respectively. Given that (i) at most one edge joins each node in the graph intersection to the symmetric graph difference, $|(u_i, u_j) \in \hat{\mathcal{E}} \ominus \mathcal{E}| \leq 1$ for all $u_i \in \hat{\mathcal{V}} \cap \mathcal{V}$, and (ii) that no two added nodes are adjacent, $(u_i, u_j) \notin \hat{\mathcal{G}}$ for all $u_i, u_j \in \mathcal{V}_+$, then*

$$\chi = \sum_{u_i \in \hat{\mathcal{V}} \ominus \mathcal{V}} \frac{\sqrt{deg(u_i)}}{\delta} + \max_{u_i \in \hat{\mathcal{V}} \ominus \mathcal{V}} \left\{ \frac{deg(u_i) - 1}{deg(u_i)} \right\} \left(\sigma_{max}(A) + \sum_{u_i \in \hat{\mathcal{V}} \ominus \mathcal{V}} \sqrt{deg(u_i)} \right), \tag{3.27}$$

where $\sigma_{max}(A)$ is the largest singular value of A.

Proof. In the derivation of χ, we cannot reuse our previous results. Deleting a node is equivalent to deleting all edges between the deleted node and its neighbors. The number of deleted edges can therefore be larger than one, which violates our constraint that at most one edge is deleted from each node. To avoid a loose bound, we do not model node deletion as a sequence of edge deletions. Instead, we redefine the constraint for node operations to allow multiple edge deletions if the edges are adjacent to a deleted node. More formally, at most one edge should join any node in the graph intersection to the symmetric graph difference. This constraint definition applies to both additions and deletions. An added node can simultaneously connect to multiple nodes, given that the nodes it connects to are not affected by other operations. For simplicity, we also assume that added nodes cannot be adjacent to each other.

We now express $\hat{A}$ and $\hat{D}^{-1}$ as a function of A and D^{-1}, respectively:

$$\hat{A} = A + \sum_{u_i \in \mathcal{V}_+} \sum_{u_j \sim u_i} \left(E_{ij} + E_{ji}\right) - \sum_{u_i \in \mathcal{V}_-} \sum_{u_j \sim u_i} \left(E_{ij} + E_{ji}\right) \tag{3.28}$$

and

$$\begin{aligned} \hat{D}^{-1} = D^{-1} &+ \sum_{u_i \in \mathcal{V}_+} \frac{\deg(u_i) - 1}{\deg(u_i)} E_{ii} + \sum_{u_i \in \mathcal{V}_+} \sum_{u_j \sim u_i} a_j E_{jj} + \dots \\ &+ \sum_{u_i \in \mathcal{V}_-} \frac{1 - \deg(u_j)}{\deg(u_j)} E_{ii} - \sum_{u_i \in \mathcal{V}_-} \sum_{u_j \sim u_i} b_j E_{jj}. \end{aligned} \tag{3.29}$$

Naturally, a deleted node can not be simultaneously added or vice versa. Also, added nodes cannot be adjacent to deleted nodes. Last, the constraint guarantees that at most one edge changes for any of the nodes that are not added or deleted. Therefore, at most one term is added or subtracted to each diagonal element. Given that $\hat{D}^{-1} - D^{-1}$ is diagonal and because $\left|\frac{\deg(u_i)-1}{\deg(u_i)}\right| = \left|\frac{1-\deg(u_i)}{\deg(u_i)}\right| \geq \max\{a_j, b_j\}$ for all $u_i \in \hat{\mathcal{V}} \ominus \mathcal{V}$ and $u_j \sim u_i$,

$$\left\|\hat{D}^{-1} - D^{-1}\right\| = \max_{i \in \hat{\mathcal{V}} \ominus \mathcal{V}} \left\{ \frac{\deg(u_i) - 1}{\deg(u_i)} \right\}. \tag{3.30}$$

For similar reasons, matrix $\hat{A} - A$ is symmetric with non-zero elements equal to one.

$$\left\|\hat{A} - A\right\| \leq \sum_{u_i \in \hat{\mathcal{V}} \ominus \mathcal{V}} \sqrt{\deg(u_i)} \tag{3.31}$$

Substituting Inequalities (3.30) and (3.31) into the definition of χ and factoring the result we get the desired bound. ■

The above bound provides three insights:

1. *Added and deleted nodes $i \in \hat{\mathcal{V}} \ominus \mathcal{V}$ introduce a convergence error that is proportional to the square root of the number of nodes they connect to.* On the left, $\sqrt{\deg(u_i)}$ is normalized to the minimum network degree. On the right, $\sqrt{\deg(u_i)}$ is weighted by the maximum value of $(\deg(u_i) - 1)/\deg(u_i)$ which tends to one as $\deg(u_i) \to \infty$.

2. *It is the most connected node involved in a node operation which effects convergence the most.* Since the right term incurs the most significant error, we deduce that the convergence error after a node operation depends to a large extent on the degree of the most connected node of the operation; *whether the node was deleted or added is of no significance.*

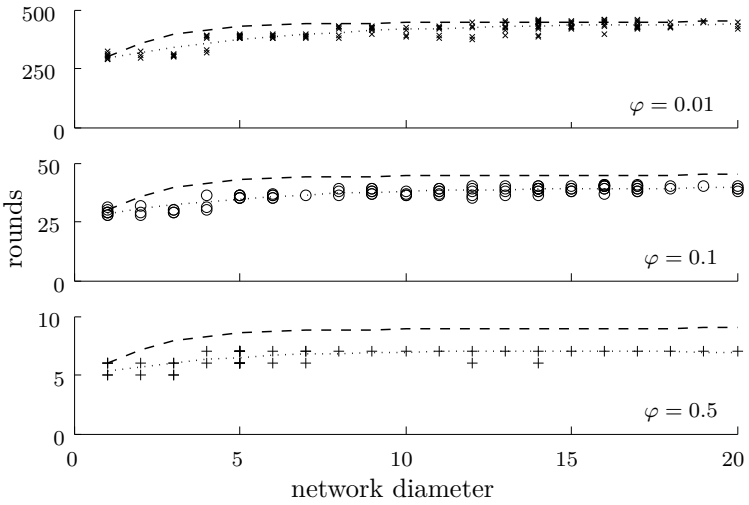


Figure 3.2: Number of rounds until the convergence error becomes smaller than 0.05 versus network diameter. Simulation results are depicted with markers connected by dotted lines and analytical results with dashed lines.

3. *It is degree irregularity that affects convergence the most, not the value itself.* This seems to be an inherent property of dynamical system (3.1) as it consistently arises in all the convergence bounds we have derived so far (see Sections 3.4 and 3.5). In contrast to regular spaces such as continuous domains and regular graphs, diffusing information in irregular spaces incurs a price.

3.6. Evaluation

Our analysis provides several insights. Here, we focus on the five which we consider most important. *We perform a controlled simulation-based evaluation* to: (*i*) evaluate the convergence of our algorithm under different settings and to validate the analytical results of 3.4.1, and (*ii*) challenge the assumptions on synchronous execution and no data losses posed in Section 3.2.2. *Furthermore, we perform a short testbed evaluation* to: (*iii*) show that the potential kernel can be implemented in resource-constrained devices, (*iv*) exemplify that it is resilient to normal operating conditions such as link variability and node failures, and (*v*) showcase its capability to make signals unimodal (Section 3.3.3). Note that our evaluation serves only as a demonstration of feasibility. A robust protocol implementation would need to consider the specific requirements of the application at hand.

3.6.1. Simulation Results

We used the COOJA simulator, a widespread tool for wireless network simulation [96]. We chose the information values arbitrarily, by setting $x(u_i) = 0$ for some random $u_i \in \mathcal{V}$ and $x(\mathcal{V} \setminus u_i) = 1$, otherwise. The nodes were deployed uniformly at random and the unit-disk model was used to establish connectivity. To capture a wide range of connectivity properties, we tested four network sizes: 10, 50, 100 and 150 nodes, with four different transmission ranges. We evaluated 14 instances for each $\langle$*size*, *range*$\rangle$ tuple, resulting in 224 different networks. Overall, the networks had average degrees between 3.4 and 17.24, diameters between 1 and 20, and clustering coefficients between 0.27 and 1. We evaluated three representative values of the inhibiting factor, i. e., $\varphi = 0.01$, 0.1, and 0.5, respectively.

Asynchrony. Our first set of experiments show that synchrony is not a critical assumption; the provided convergence bounds still hold for in the *asynchronous* model. Figure 3.2 summarizes our experiments. The markers represent simulation results and the dashed lines represent the convergence bounds from Section 3.4.1. Each marker represents a $\langle$*topology*, $\varphi\rangle$ tuple, and for each tuple, we record the round when $\left\|y - y^{(t)}\right\| \leq 0.05$, i. e., when it converges. The iterative calculation of $y^{(t)}$ used our distributed algorithm, while the ground-truth y was calculated assuming an oracle's view of the network. To test our method under the worst possible circumstances, we intentionally bootstrap nodes to a value that is far from the steady state, $y^{(0)} = 0 \ll x, y$. It is important to highlight the trade-offs of the inhibiting factor φ. Our method is better suited for large-scale networks. The number of rounds to reach convergence does not increase significantly with the network's diameter (for any value of φ). On the other hand, small values of φ are not suitable for small networks. Observe that, even in fully connected networks (1-hop diameter), our method incurs an overhead that is on the order of $1/\varphi$, which translates to more than 250 rounds for $\varphi = 0.01$. Setting the inhibiting factor to very small values only makes sense for large networks or in cases of high information variation. For small networks, high values of φ or simpler aggregation schemes may be preferable.

Robustness to data loss. Observe that some markers in Figure 3.2 are over the bounds. The observed difference suggests that message loss, i. e., our second assumption, has an effect on the algorithmic operation. Message losses require more rounds than expected to reach convergence. Figure 3.3 distills the results of a set of experiments which characterize the effect of imperfect knowledge. In the experi-

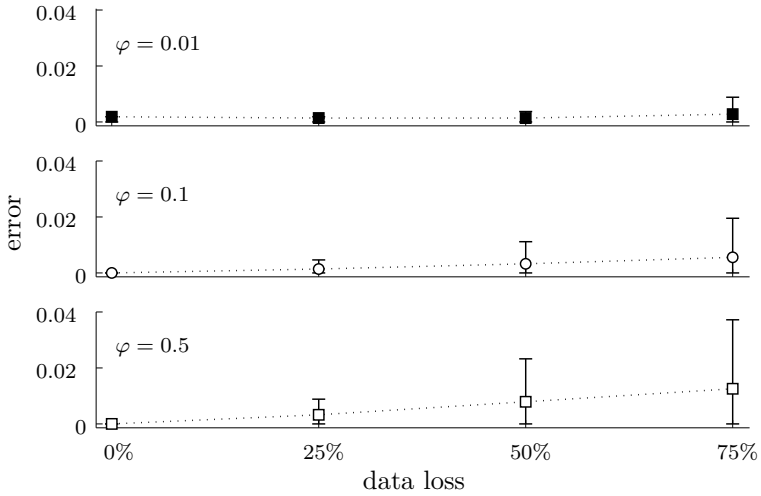
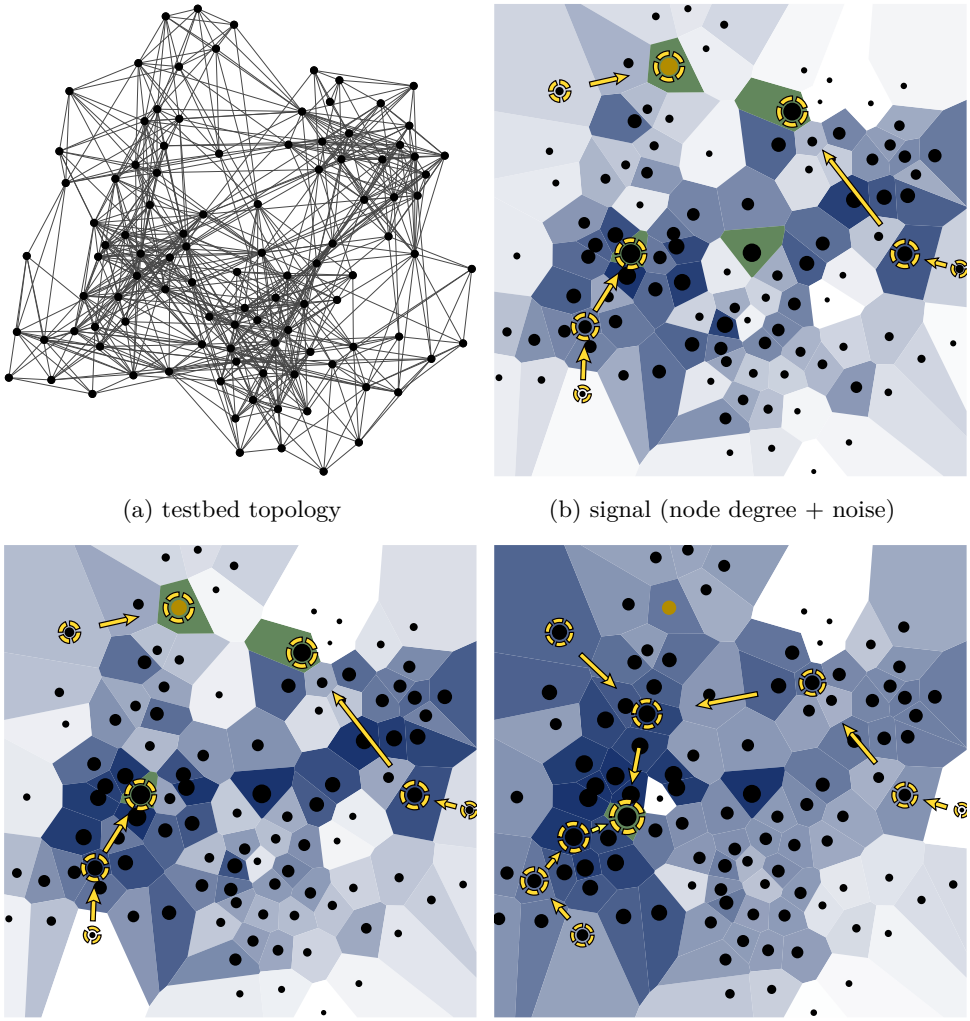


Figure 3.3: Error after convergence versus percentage of unknown neighboring values to the total neighbors at the end of each round. The algorithmic robustness increases for small φ.

ment, nodes disregarded a specific, randomly selected, percentage of their received values $\mathcal{S}_i$ just before recomputing their value (line 9 of Algorithm 2). Before measuring the normalized potential error of a given topology, we waited for a sufficient number of rounds until the algorithm had converged. For each $\langle \varphi, \textit{data loss} \rangle$ tuple, we summarize the errors across time from all 224 topologies, by the corresponding median and 68.2% confidence interval. Even under severe loss, the algorithm exhibits an error that is smaller than 0.04. The algorithm is robust because it is not based on algorithmic primitives that are sensitive to message loss, e. g., mass conservation, or to partial knowledge, e. g., counting. Instead, it employs averaging as a statistical measure that, in non-skewed distributions, approximates well the central tendency, even with small sample sizes. Comparing the three subfigures we also notice that, for large percentages of data loss ($\varphi = 0.01$), the topmost subfigure reports a median error that is approximately two times smaller that the one in the bottommost subfigure ($\varphi = 0.5$). This phenomenon stems from the tendency of small φ to limit the magnitude of change between consecutive computation rounds. As such, the sensitivity of the algorithm to the high variation effects that accompany data loss decreases with φ.



(a) testbed topology (b) signal (node degree + noise)

(c) $\varphi = 0.50$, 3 maxima (d) $\varphi = 0.01$, 1 maximum

Figure 3.4: Information potentials (c–d) and signal (b) on a testbed of 105 nodes (a). Local maxima reside in green Voronoi cells. The orange node is an artificially injected maximum. As the aggregation scope widens, our method progressively eliminates local extrema.

3.6.2. Empirical Results

This section evaluates our method in a wireless testbed of 105 nodes. The testbed is deployed in the ceiling of our floor in TU Delft. The devices are equipped with a MSP430 micro-controller and a CC1101 radio chip, with the transmission power set to $-30\,\mathrm{dBm}$. The algorithm was implemented on Contiki OS. For medium access

control, we used NullMAC, a simple random-access MAC protocol with carrier-sense capabilities that is part of the standard Contiki OS. Each computation round lasted for 0.5 seconds, over which each node exchanged an average of 3.5 messages. An instance of the connectivity graph is shown in Figure 3.4a. As also revealed by previous investigations [123], the connectivity was highly variable over time due to the well-known volatility of low-power wireless links. Figure 3.4b depicts the nodes' degree. The darker the Voronoi cell color (the larger the disc), the higher the degree. The maxima are shown in green cells and the yellow arrows represent greedy searches that reach their respective local maxima. Due to the non-uniform coverage of radio transceivers, nearby nodes may not have a link, while far away nodes may – this is typical in testbeds and real-world deployments. This implies that nodes in adjacent Voronoi cells are not necessarily neighbors (in terms of distance) and the reader should refrain from interpreting Voronoi diagrams as continuous fields.

To demonstrate that the potential kernel eliminates phenomena of high variation, such as noise and false extrema, (*i*) we corrupted the information process by adding noise to each node's degree. The noise was uniformly distributed between [0, δ]. Also, (*ii*) we introduced a false maximum at a randomly chosen "faulty" node – colored orange. Figures 3.4(c–d) show the computed information potentials for $\varphi = 0.50$ and 0.01, respectively. Each experiment was run for ten minutes, during which we experienced a high variability of links and some (normal) node failures between the evaluations of $\varphi = 0.50$ and 0.01. This test highlights the resilience of our method to real network phenomena. Observe how our method progressively eliminates local maxima, thus making the potential unimodal. Figure 3.4(c) eliminates one of the four local maxima and Figure 3.4(d) has a single extrema. Two of the three search queries get stuck in local maxima in Figures 3.4(b–c), while in Figure 3.4(d) all queries reach the global maximum. Observe also that the position of the global maximum is not the same across all figures, which confirms our analysis in Section 3.3. Our method uncovers the node with the largest information in its vicinity, which is not necessarily the one with the largest information (discovering paths towards a maximum value can be easily solved using a max-consensus algorithm and a distance gradient).

3.7. Related Work

The computation of potential fields is inspired by the natural process of *chemotaxis*, in which cells respond to the concentrations of chemicals in their environment [6], and has been exploited to achieve the coordination of swarms [94]. In the following,

we group related work into three subcategories. First, we discuss the connections to spatial filters. Second, we focus on unimodality and thus on information discovery. Last, we classify and relate our work with respect to the family of consensus algorithms.

Spatial filters. In sensor networks, potential fields are information specific and vicinity based; nodes consider surrounding information with a significance that decays with distance [45, 63, 101]. Gao et al. [45], use quad-trees to store data, such that each node is aware of the data in its vicinity. In their seminal paper [63], Kempe et al. propose *spatial gossiping* algorithms, in which any two nodes gossip with probabilities that decrease polynomially to their distance. Sarkar et al. [101] extend spatial gossiping to compute multi-resolution representations of information. Their algorithm computes information aggregates over exponentially enlarging areas centered at each node. All of these approaches however use physical distance to define information affinity. Our method defines affinity using the random movement of particles on the network. As a result, it is *sensitive to the network topology and independent of any knowledge of physical location or distance.*

Information discovery and potential fields. A number of recent works have also employed information potentials as mediums of discovery [45, 71, 103, 112]. The closest to our approach is the work by Lin et al. [71]. In their paper, the authors construct smooth harmonic gradients towards node subsets (sources) such that local forwarding guarantees their discovery. Their method achieves the absence of local extrema by keeping the information of sources constant, fixing the values of boundary nodes to zero, and averaging in between. The number of extrema however is equal to the number of sources. In comparison, our method guarantees unimodality irrespectively of the number of sources. It does not require knowledge of the network boundary. It is also more flexible as it aggregates values that stem from a real-valued monitored process, as compared to the boolean distinction between sources and non-sources. Sarkar et al. [103] design query mechanisms for general information fields, which support the use of more advanced operations, such as iso-contour queries and value restricted routing. Their approach is complementary to ours, as it does not concern the landscape formation but advanced methods of information discovery.

Consensus algorithms and graph filters. From an algorithmic point of view, this chapter proposes a variant of the well known consensus algorithms [5, 38]. In a strict sense however, our algorithm does not solve the consensus problem; the steady state of a consensus algorithm is $\alpha\mathbb{1}$, where $\alpha \in \mathbb{R}$ is usually the average or the maximum of the information. In our case, each node converges to a distinct

value; the collection of values form landscapes that support information discovery. However, Khan et al. recently proposed a wider family of consensus algorithms, referred to as *higher dimensional consensus algorithms*, under which our method can be classified [64]. We also have to note that, unlike most consensus and gossip algorithms that are used in sensor networks [5, 15], our method is not randomized and does not sacrifice accuracy in the presence of communication loss. Throughout the computation, information works as an anchor that steers the network towards the correct steady state. As witnessed by our evaluation, message loss mainly increases the variance of the steady state and has little effect on the mean error.

Due to its connection to the graph Laplacian, our method is also related to data clustering algorithms, such as mean shift clustering [19] and spectral clustering [116]. Choa et al. [20] independently proposed a similar approach to compute the modes of a graph. Even though their paper concerns the processing of images, it is a special case of our method as they consider only the case of $x(u_i) = \deg(u_i)$.

3.8. Conclusions

Hitherto, distributed low-pass graph filters have relied on synchronous initialization, computation, and termination. In this chapter, we introduced a novel filter that overcomes these limitations. By *anchoring* the value of each node to their *original* value – as opposed to letting it *evolve "freely"* at each round, the potential kernel becomes robust to computation inaccuracies such as those resulting from dynamics, asynchronous execution, and communication loss. It is also simple, decentralized, and converges ε-close to the steady state in constant time (linear convergence). We analyze the potential kernel and provide valuable guidelines for practical deployments: (*i*) dense regular networks provide faster convergence rates; (*ii*) expected information and network dynamics can be used to derive appropriate aggregation scopes.

Motivated by the use of low-pass graph filters for the computation of information potential fields, i. e., network signals used for greedy search, we use spectral graph theory to gain insights into how low-pass graph filters affect the extrema of a signal. We provide two main results. First, we bound the number of extrema of each signal component of the spectral transform. The bound is the first step towards characterizing the number of extrema of an information potential. We then study the limiting behavior of extrema and the unimodality property. We show that even though no linear graph filters exists which renders every signal unimodal, in practice unimodality often occurs.

4

Band-Pass Graph Filters

Event Region and Boundary Detection

"I've always had this strange unaccountable feeling that something was going on in the world, something big, even sinister, and no one would tell me what it was."

"No," said the old man,
"that's just perfectly normal paranoia. Everyone in the Universe has it."

—Douglas Adams,
The Hitchhiker's Guide to the Galaxy

The decentralized detection of event regions and their boundaries is a fundamental building block for monitoring and reasoning about spatial phenomena. However, so far the problem has been studied almost exclusively for static networks. This chapter proposes a theoretical framework with which we can analyze event detection algorithms suitable for large-scale mobile networks. Our analysis builds on the following insight: *the inherent trends of spatial events are well captured by the spectral domain of the network graph.* Using this framework, we propose novel graph

Parts of this chapter have been published in INFOCOM'14, April 2014, Toronto, Canada [81].

filters that are location-free; that work with mobile nodes and dynamic events; that operate on 3D topologies; and that are simple to implement. We are not aware of event detection algorithms (or filters) possessing all these traits. Simulations based on complex oil spill traces showcase the resilience and robustness of our filters. Additionally, we demonstrate their validity for practical scenarios by evaluating them on a 100+ node testbed.

4.1. Introduction

The real-time monitoring of spatial phenomena, like oil spills, pollution clouds, or chemical spills, is one of the key applications fostering the development of sensor and robotic networks [8]. Recent studies have demonstrated that sensor networks can distributedly perform high-level inference tasks, such as identifying "*how many distinct events are currently occurring*" or recognizing if "*two regions are adjacent*" [34, 59]. To reason about spatial events however, a network must first identify and track the events of interest – see Figure 4.1. Indeed, the event detection problem has attracted significant interest [32, 33, 60, 70, 84, 93, 107, 117]. Nevertheless, the overwhelming majority of these works focus on the static case. When the network is mobile and the events dynamic, event detection becomes an even more challenging problem. Each node needs to distinguish the underlying signal trends (spatial events) from noise by using only local information and, preferably, without knowing its location.

This chapter focuses on the design and evaluation of local event detectors for mobile networks. Our basic insight is that, the decomposition of event signals in the spectral domain –the domain spanned by the eigenvectors of the graph Laplacian– captures fundamental signal trends. In fact, the identification of spatial regions can be achieved by filtering the event's signal in the spectral domain. Based on this observation, we propose a spectral framework for event detection. This is a generic framework that can be used to model a number of algorithms for event region and boundary detection, such as those using basic differentials and differentials of diffusions (e. g., LoG and DoG filters [11]). By decoupling the operation of the algorithm from the underlying topology, our framework allows a systematic comparison of event detectors (e. g., with respect to their resilience and resolution), while also providing a deep insight into the effect of fundamental topological properties (e. g., spectral gap).

Our analysis has important consequences on the design of algorithms for event detection. We propose decentralized and asynchronous band-pass graph filters that

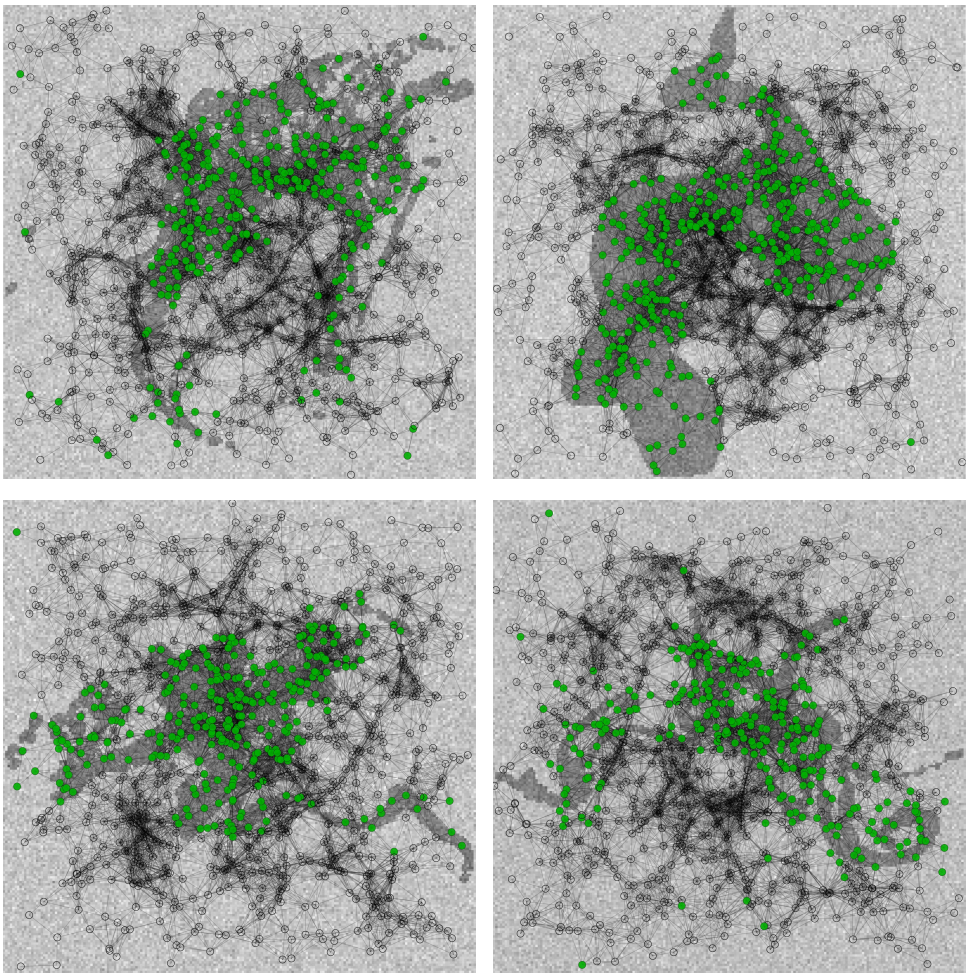


Figure 4.1: The objective of an event detector is to locally distinguish between spatial events and noise. Above, green nodes have identified the presence of an oil spill – shown in dark gray. Green nodes with non-green neighbors are boundaries. The datasets are forecasts of the Deepwater Horizon oil spill [77]. A video showcasing the robustness of our filters to node mobility and event dynamics is posted at `http://youtu.be/oDyg4gO7Flo`.

do not require location coordinates, can track fast changing signals, are little affected by mobility, can handle 3D topologies, and are simple to implement. To the best of our knowledge these are the first algorithms for event detection that have *all* these characteristics.

4.1.1. Related Work

Given the rich literature in the area of event detection, we focus our discussion on the most relevant and recent results in the field. Distributed event region and boundary detection has two main research thrusts: the identification of network boundaries [118, 130] and of events and their boundaries [32, 33, 60, 70, 84, 93, 107, 117]. While in the first case the aim is to solely identify the nodes at the edge of the network, in the case of events, an algorithm must first detect the event's region *within* the network. The two approaches can be seen as complementary. Once event regions are detected, network boundary algorithms can be used to enhance the identification of boundaries. The focus of this work is on event detection.

Even though the event detection problem has been studied comprehensively on static networks, previous approaches are incompatible with mobility. This is attributed to three main assumptions:

Computation. Some studies assume that the network has a hierarchical structure [93] or that it can offload its data to a central server, where the event detection is performed [84]. Hierarchical approaches suffer from scalability problems and are not amenable to frequent changes in the network topology. Clearly, a local model of computation is preferable.

Location. The majority of previous algorithms require the geographical coordinates of nodes [32, 70, 107, 125]. Obtaining/maintaining location information is possible but incurs a high and recurring overhead, and introduces imprecision. Hence, we advocate a location-free approach, where algorithms exploit network connectivity to infer spatial correlation, such as [33, 60, 84, 117] (these location-free algorithms however suffer from either the *computation* or the *stationarity* constraint). If localization services are available, our algorithms additionally localize events.

Stationarity. We have found only a handful of location-free approaches that are local [33, 60, 117]. Nevertheless, they assume that the events are either stationary [33, 60] or change slowly [117]. The stationarity assumption is a problem not only because real events are time varying, but also because, when the position of nodes change over time, nodes observe different events at different times and need to adapt promptly. Our analysis and simulations show that our algorithms can track fast changing signals and that they degrade gracefully with mobility.

4.1.2. Contributions

Our work advances the area of event region detection by providing a novel perspective – the use of spectral graph theory to analyze and develop simple distributed algorithms. Our main research contributions are:

Contribution 1. *We propose a spectral framework to reason about event region detection—Section 4.3.* Our framework captures the fundamental properties of detectors, as well as the influence of the network topology.

Contribution 2. *To benchmark our algorithms with respect to state-of-the-art Lapla-cian-based detectors, we modify two established band-pass filters used in image processing to work in wireless networks: the Laplacian of a Gaussian (LoG) and Difference of Gaussians (DoG) [11]—Section 4.4.1.* The resulting detectors are location-free, scalable, suitable for 3D topologies, and simple to implement; but they have low resolution and can not operate in mobile networks with dynamic events (see Figure 2.3).

Contribution 3. *To overcome the limitations of LoG and DoG, we introduce two novel band-pass graph filters based on the potential kernel Chapter 3, the Laplacian of a Potential (LoP) and the Difference of Potentials (DoP)—Section 4.4.2.* Our analysis and simulations, based on complex oil spill traces, validate the superior resolution of DoP, as well as its robustness to network and event dynamics (Section 4.5).

Contribution 4. *We implement DoP on the Contiki OS and evaluate it on a testbed with 105 nodes—Section 4.5.* The experiments validate the simplicity and accuracy of our algorithms in real-world conditions.

4.2. The Filtering Problem

The detection of event regions and their boundaries breaks down to two objectives: the computation of *signal curvature* and the elimination of *small and flat regions*. This section explains how both objectives can be tackled using distributed graph filters.

4.2.1. Signal Curvature

As exemplified in Figure 4.1, the primary objective of an event detector is to annotate each node as being part of a spatial event region or not. The problem input is a real-valued signal x defined on the nodes of a graph $\mathcal{G}$. Sharp signal transitions in x are used to identify region boundaries. These sharp signal transitions occur on nodes over which the first derivative of x is maximized (or minimized) with respect to their neighborhood. This process is equivalent to finding "inflection point nodes" where the second derivative of x is zero. Inflection points can be computed locally using the discrete graph Laplacian[1], given in (2.12). Vector Lx is therefore a measure of curvature, which is always positive in concave regions, negative in convex regions, and zero in inflection points. After the Laplacian is computed, nodes can locally decide whether they belong to an event region or not using the decision process given in Algorithm 3.

Algorithm 3 Detecting event regions with the Laplacian.

if $(Lx)(u) > 0$ **then** ▷ The signal is concave.
  if $\exists v \sim u : (Lx)(v) < 0$ **then**
    u is on the interior of the boundary.
  else
    u is within an event region.
  end if
else if $(Lx)(u) < 0$ **then** ▷ The signal is convex.
  if $\exists v \sim u : (Lx)(v) > 0$ **then**
    u is on the exterior of the boundary.
  else
    u is outside an event region.
  end if
else if $\forall v \sim u : (Lx)(u) = (Lx)(v)$ **then** ▷ Flat region.
  inconclusive.
else ▷ Inflection point.
  u is on the boundary.
end if

[1] Formally, the Laplacian is the discrete and negative version of the Laplace-Beltrami operator and it equals the negative trace of the Hessian matrix, which contains the partial second derivatives of x.

Due to convention, we refer to regions with positive (negative) Laplacian values as *events* (*non-events*). The region's *boundaries* are inflection points that separate a region from its environment. Notice that, due to the discretization of the underlying space, a Laplacian signal might go from positive to negative values without crossing zero. The above algorithm thus also accounts as boundaries the nodes within/outside each event region with at least one neighbor outside/within the region. Summarizing, events can be seen as the maximal induced subgraphs over which Lx does not change sign.

It is important to remark that L is a local operator that is robust to incomplete information. Even though the computation of $(Lx)(u)$ assumes the values of all neighbors, if – e. g., due to message loss – some values are missing, the (incomplete) computed quantity is still a good estimator of $(Lx)(u)$. What's more, in contrast to threshold-based approaches [60], the Laplacian requires no preconceived knowledge of the signal range.

4.2.2. Small and Flat Regions

The Laplacian has two main pitfalls. First, it is very sensitive to noise as it cannot differentiate between sharp signal transitions and those attributed to noise. Second, it suffers from a "flat-region ambiguity problem". If the signal stays constant over some region, it is impossible to decide locally whether the region is part of an event or not.

The prevalent approach to overcoming these problems is the use of a low-pass filter. Before the computation of curvature and the identification of events, the signal is diffused with a low-pass filter $\mathbf{K}_\circ$, parametrized on $\circ$. Low-pass graph filters disseminate the information in the signal x over multiple hops to reveal underlying curvature trends. They smooth out sudden signal transitions and make flat regions curved.

Most filter-based detectors can detect event regions over a range of scales. The choice of parameter $\circ$ controls the size of detectable regions. In general, the more a signal is diffused, the larger the size of regions that can be detected. If an event is larger than the largest detectable region, multiple events are detected. Conversely, events smaller than the smallest detectable region are not recognized.

Figure 4.2 depicts the size of detectable events for a sample LoG filter (described in Section 4.4) when presented with an *impulse signal* $\delta_u(v) = 1$ if $u = v$ and $\delta_u(v) = 0$ otherwise. The smallest region is the induced subgraph that has as

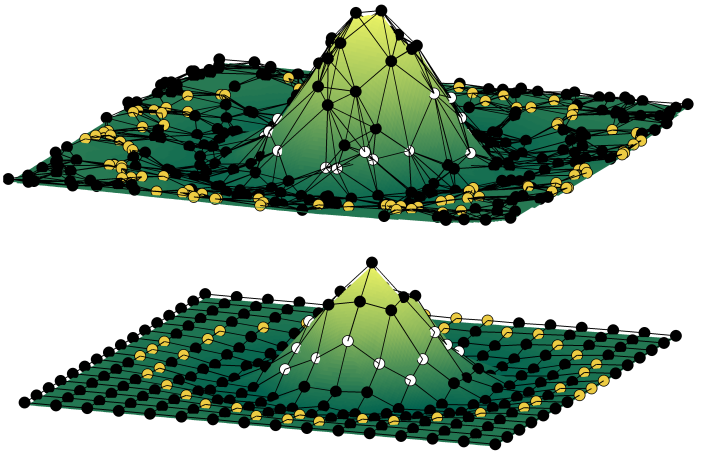


Figure 4.2: The impulse response of a $\mathbf{LoG}_{10}$ filter for a grid topology (bottom) versus a randomly distributed topology (top). The boundaries of the smallest and largest detectable regions (shown with white and orange markers, respectively) are complex and depend on the topology.

boundary the inflection points of the impulse response (white inner nodes). The largest region is the induced subgraph that has as boundary the out-most nodes with non-zero response (orange outer nodes). The exact influence of the network topology on the event signal x and the filter $\mathbf{K}$ is encoded in the spectrum of the graph Laplacian and is non-trivial. The analysis of these relations is one of the main contributions of our work and is described next.

4.3. The Spectral Anatomy of an Event Detector

Studying event region detection algorithms based on their impulse response –see Figure 4.2– only provides insight on how an algorithm operates in a specific topology. This section generalizes the approach by proposing a spectral framework that decouples event detectors from the topology. Based on the results provided here, in Section 4.4 we will quantify basic detection properties, such as resilience to noise and detection resolution[2].

[2]A high resolution facilitates the detection of fine-grained details at the event's boundary.

4.3.1. The Regions of Eigenvectors

A complex event region can be described by the union of simpler regions of different sizes. Large regions compose the event's main body and small regions the details of its shape (or perhaps noise). From this point of view, an event region detector is a graph filter that distinguishes the simple regions of a complex event and filters them according to their size: the detector favors large over small regions, such as noise, and eliminates the influence of very large regions, such as the signal's DC-offset. An event region detector is therefore a band-pass graph filter.

In the following, we show that the spectral transform inherently captures such a decomposition. We explain that each eigenvector partitions a graph into a different number of subgraphs (nodal domains). This partitioning induces scale: the higher the order of the eigenvector, the smaller the size of the regions. The magnitude of the k-th signal component then quantifies the overlap between the signal and the subgraphs partitioned by the k-th eigenvector.

As we saw in Chapter 2, a strong *nodal domain* is –roughly defined– the maximally induced subgraph over which a signal does not change sign. Let random variable $\mathcal{D}_{\phi_k(\mathfrak{L})}$ describe the number of nodes in a nodal domain of the k-th eigenvector of a generalized graph Laplacian matrix $\mathfrak{L}$. These nodal domains can be interpreted as "the regions of eigenvectors". We exploit this property to associate a minimum region size to each signal component.

Proposition 4.1 (Region Size)**.** *Let $\phi_k(\mathfrak{L})$ be the k-th eigenvector of a generalized graph Laplacian $\mathfrak{L}$, which is zero at z nodes and has multiplicity r. Moreover, denote by $\mathcal{D}_{\phi_k(\mathfrak{L})}$ the random variable which describes the number of nodes in a nodal domain of $\phi_k(\mathfrak{L})$. Since,*

$$\frac{n}{k+r-1} \leq \mathrm{E}[\mathcal{D}_{\phi_k(\mathfrak{L})}] \leq \frac{n}{k+r-1-(m-n+z+1)}. \tag{4.1}$$

the size of the regions of the k-th signal component is (in expectation) bounded and decreases with k.

Proof. The proof directly follows from Theorem 2.1 and Theorem 2.2, by exploiting the fact that the sum of nodes for all strong nodal domains of an eigenvector (the summation of all realizations of the random variable) is always equal to n. ■

It is also important to show that all the nodal domains of each eigenvector have similar sizes, otherwise they would not capture scale. We next show that this is indeed true for the eigenvectors of the normalized graph Laplacian.

Proposition 4.2 (Region Size Variance)**.** *Let ϕ_k be the k-th eigenvector of the normalized graph Laplacian and $\mathcal{D}_{\phi_k}$ a random variable which describes the number of nodes in a nodal domain of ϕ_k. The variance of $\mathcal{D}_{\phi_k}$ is small.*

Proof. The evidence that $\mathrm{Var}(\mathcal{D}_{\phi_k})$ is small is a consequence of graph partitioning [116]. Recall that, according to the *Ncut* problem, partitioning a graph into k subgraphs is equivalent to finding the subgraphs $\mathcal{G}_1, \ldots, \mathcal{G}_k$ that minimize

$$Ncut(\mathcal{G}_1, \ldots, \mathcal{G}_k) = \sum_{i=1}^{k} \frac{|\{uv \in \mathcal{E} : u \in \mathcal{V}_i \text{ and } v \in \bar{\mathcal{V}}_i\}|}{|\mathcal{V}_i|},$$

with $\bar{\mathcal{V}}_i$ being the complement of $\mathcal{V}_i$. Since each cut is normalized by the size of the partition, the objective function penalizes uneven partitions and $\mathrm{Var}(\mathcal{V}_i)$ is small.

In their seminal work, Shi and Malik [106] showed that, for the relaxed *Ncut* problem, the graph partitions are exactly the nodal domains of the eigenvectors of the random-walk normalized laplacian matrix P. It is therefore obvious that $\mathrm{Var}(\mathcal{D}_{\vartheta_k})$ is small. We can also extend this result to the eigenvectors ϕ_k of the normalized Laplacian. Since $\phi_k = D^{1/2}\vartheta_k$, the nodal domains of vectors ϕ_k and ϑ_k are identical and $\mathrm{Var}(\mathcal{D}_{\phi_k}) = \mathrm{Var}(\mathcal{D}_{\vartheta_k})$. ■

Though we lack a formal proof, we suspect that the same holds for any generalized Laplacian.

4.3.2. Parametrization

Parametrizing a graph filter analytically is a difficult task. The operation of a filter depends on a number of factors, such as the shape of its spectral response and the spectrum of the graph it operates on. Nevertheless, when the graph spectrum is known, it is possible to obtain parametrization guidelines in the form of bounds: we can tune a filter to only favor regions larger or smaller than some value σ.

For simplicity, we will consider an ideal band-pass filter $\mathbf{B}_{\{\alpha,\beta\}}$, which only allows signal components with $\lambda_k \in [\alpha, \beta]$ to pass.

Proposition 4.3. *Let $\mathbf{B}_{\{\alpha,\beta\}}$ be an ideal band-pass filter operating on the spectrum of some generalized graph Laplacian matrix $\mathfrak{L}$.*

1. *If α is equal to the $\left(\frac{n}{\sigma} + 1 + l + z_{max}\right)$-th smallest eigenvalue $\mathfrak{L}$, the filter will, in expectation, tend to ignore event regions larger than σ.*

2. *If β is equal to the $\left(\frac{n}{\sigma}+1-r_{max}\right)$-th smallest eigenvalue of $\mathfrak{L}$, the filter will, in expectation, tend to ignore event regions smaller than σ.*

Above, r_{max} is the maximum multiplicity of any eigenvalue and z_{max} is the maximum number of times any eigenvector crosses zero.

Proof. Suppose σ is the expected size of some nodal domain of an eigenvector and $k = g(\sigma)$ is the eigenvector's order. That is, g is a function that maps expected size of nodal domains to eigenvector order. Inverting inequality (4.1), we have that

$$\frac{n}{\sigma}+1-r_{max} \leq \frac{n}{\sigma}+1-r \leq g(\sigma) \leq \frac{n}{\sigma}+1-r+l+z \leq \frac{n}{\sigma}+1+l+z_{max} \quad (4.2)$$

where r_{max} is the maximum multiplicity of any eigenvalue and z_{max} is the maximum number of times any eigenvector crosses zero. What is left is to map the order to the response's domain. From definition, $f : k \to \lambda_k$ is a strictly increasing function. It follows that

$$f\left(\frac{n}{\sigma}+1-r_{max}\right) \leq (f \circ g)(\sigma) \leq f\left(\frac{n}{\sigma}+1+l+z_{max}\right). \quad (4.3)$$

As such, if β is set equal to the lower bound of $(f \circ g)(\sigma)$, $\mathbf{B}_{\{\alpha,\beta\}}$ will tend to ignore events with size smaller than σ. Conversely, if α is set equal to the upper bound, $\mathbf{B}_{\{\alpha,\beta\}}$ will tend to ignore events with size larger than σ. ■

4.4. Band-Pass Filters for Event Detection

As previously discussed, the Laplacian needs to be preceded by a low-pass filter to overcome its sensitivity to noise and its ambiguity of flat regions. In this section, we present two types of filters. The first, which we call *heat-based*, are derived from image processing techniques and are used as benchmarks. These filters exhibit very good resilience to noise, but suffer from two significant drawbacks: they require node synchronization, which limits their use in dynamic settings, and have low detection resolution. The second type use the potential kernel as a diffusion primitive [82]. *Potential-based filters* are asynchronous, have high resolution and are not significantly affected by noise.

4.4.1. Heat-based Filters

Heat-based filters use heat diffusion to eliminate sharp signal transitions, such as noise. We start by discussing the fundamentals of heat diffusion and then present

two heat-based band-pass filters for event region detection, the Laplacian of a Gaussian (LoG) and the Difference of Gaussians (DoG). Since both filters are graph-counterparts of well-known algorithms for image processing [11], our presentation focuses mainly on the effect of the topology and on their decentralized implementation.

Heat kernel. As discussed in Section 2.3, the heat diffusion of a signal x after t rounds is given by the discrete heat kernel

$$\mathbf{H}_t x = \sum_{k=1}^{n} (1 - \lambda_k)^t \langle x, \phi_k \rangle \phi_k, \tag{4.4}$$

where t is a user defined parameter and, λ and ϕ are the eigenvalues and eigenvectors of $\mathcal{L}$, respectively. The number of rounds t is a parameter with which one controls how much a signal is diffused. Heat diffusion eliminates the high-order signal components and imposes a restriction on the minimum size of detectable event regions. In short, as t grows, "small regions" caused by the event itself or by noise are filtered out.

Contrary to image processing applications, which are usually solved in a single computer, we need an efficient and distributed computation of $\mathbf{H}_t x$ for large-scale wireless networks. One possibility is to use Algorithm 1, which is both simple and decentralized. Still, a node using Algorithm 1 requires at each step not only the values of its neighbors, but also their degrees[3]. A simpler alternative is to substitute line 4 of Algorithm 1, with

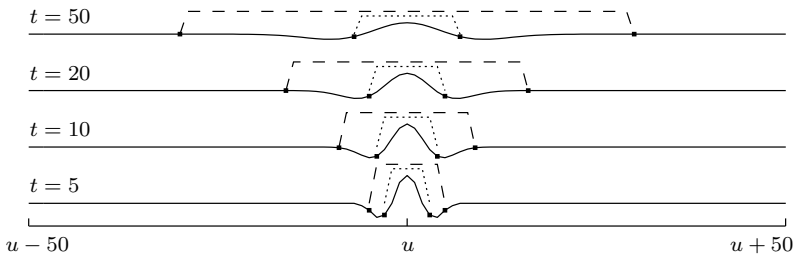
$$y(u_i) \leftarrow \sum_{u_j \sim u_i} \frac{y(u_j)}{\deg(u_i)}$$

resulting to the modified kernel:

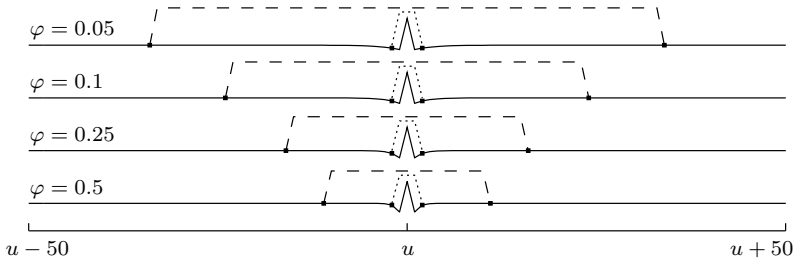
$$\tilde{\mathbf{H}}_t x = \sum_{k=1}^{n} (1 - \lambda)^t \langle x, \vartheta_k^{-1} \rangle \vartheta_k, \tag{4.5}$$

where ϑ_k is the k-th eigenvector of the well-known matrix $P = D^{-1}A$. The difference with Formula 4.4 matters little because $\vartheta_k = D^{-1/2}\phi_k$. For the rest of this chapter, when we refer to $\mathbf{H}_t$ we mean $\tilde{\mathbf{H}}_t$.

[3]Since in our model each node requires one round to count its neighbors, the use of node degree proves problematic in case of dynamics. To illustrate this, suppose that during round t node u receives the degree of its neighbor v and updates its value $y(u)$. If the degree of v at rounds $t-1$ and t is not the same, u divides $y(v)$ with the wrong value.



(a) LoG



(b) LoP

Figure 4.3: LoG and LoP impulse responses on a 100 node path-graph. The t and φ parameters determine the smallest (dotted lines) and largest (dashed lines) event regions the two filters detect.

Laplacian of a Gaussian. We now present LoG, our first band-pass filter. LoG entails computing the Laplacian of a signal that has been filtered using the heat kernel:

$$\mathbf{LoG}_t x \triangleq L\mathbf{H}_t x. \tag{4.6}$$

LoG is very robust to noise. By first smoothing a signal appropriately, only transitions that correspond to significant trends are identified. Furthermore, flat regions become either convex or concave, which eliminates the flat-region ambiguity problem. It is also obvious that LoG can be computed locally. The computation terminates in $t + 1$ rounds and after the network exchanges $2m(t + 1)$ messages, each containing two numerical fields, the node id and the current average, that is $msg_t(u) = [u, (\mathbf{H}_t x)(u)]$.

Figure 4.2 shows LoG's impulse response for two sample topologies. In a *regular grid* (which is the case for images), LoG approximates a Mexican hat. The smallest and largest event regions LoG detects are discs with radius close to $\sqrt{t}$ and t, respectively. The effect of t on the size of detectable regions is more clearly seen in a *path graph*, such as the one depicted in Figure 4.3a. In *non-regular topologies*,

the kernel exhibits more complex geometry. We can have a better understanding of how LoG works in different graphs through its spectral response. But to do that, we have to first express LoG in a spectral form. From definition,

$$\mathcal{L}\phi_k = \lambda_k \phi_k$$
$$D^{-1/2} L D^{-1/2} \phi_k = \lambda_k \phi_k$$

$$L D^{-1/2}(D^{1/2}\vartheta_k) = \lambda_k D^{1/2}(D^{1/2}\vartheta_k)$$
$$L\vartheta_k = \lambda_k D \vartheta_k.$$

Substituting this in (4.6), we get LoG's spectral response:

$$r_{\text{LoG}}(\lambda; t) = \lambda(1-\lambda)^t, \tag{4.7}$$

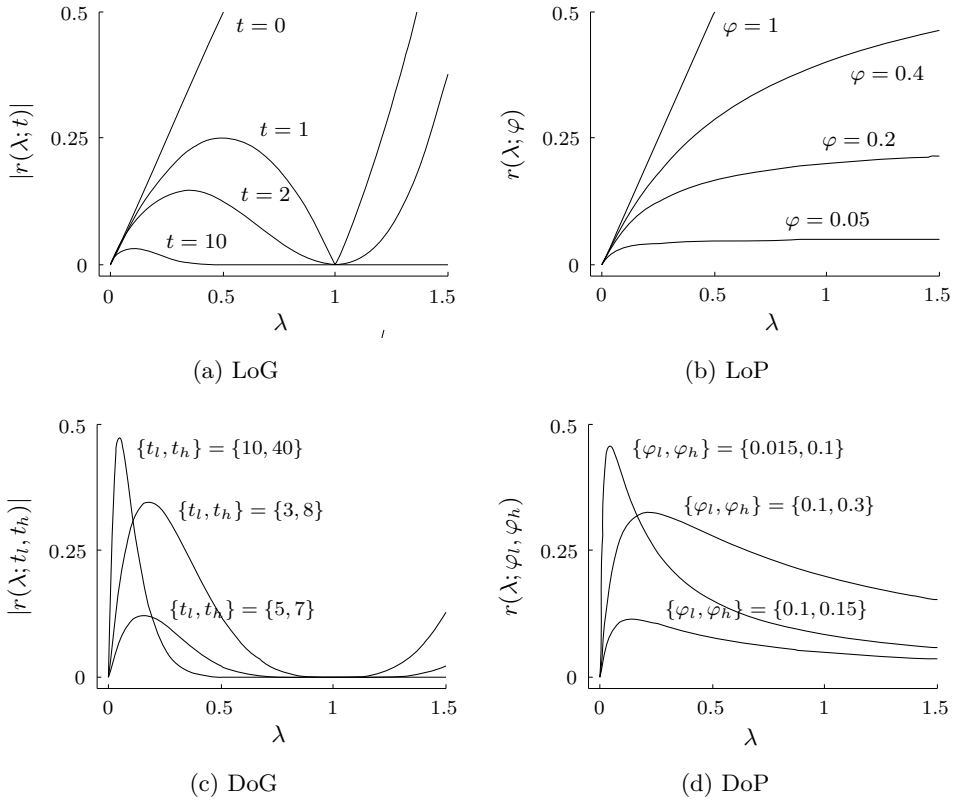
where the eigenvectors are normalized by the degree matrix D. The spectral response is shown in Figure 4.4a for representative values. Observe that, irrespectively of parametrization, LoG removes the signal's DC-offset, i. e., the first signal component. This is a common characteristic of all detectors and is essential for the identification of curvature. Furthermore, as t increases, signal components with large eigenvalues –and therefore small event regions– are progressively attenuated. The sudden increase after $\lambda = 1$ is insignificant as rarely $\lambda > 1.2$ and only occurs for small t. In Section 4.4.3 we study LoG's spectral response further and compare it with the other algorithms presented in this chapter.

Difference of Gaussians. In LoG, the sizes of detectable events is controlled by t, which is a single parameter. We now examine a filter where the size of the smallest and largest events are determined independently.

Instead of using the Laplacian directly, we estimate it by subtracting two heat kernels of different widths

$$\mathbf{DoG}_{\{t_l, t_h\}} x \triangleq \mathbf{H}_{t_l} x - \mathbf{H}_{t_h} x, \tag{4.8}$$

with parameters $0 \leq t_l < t_h$. Intuitively, parameter t_l controls how much noise is removed from the signal, while t_h eliminates very slow changing signal components. The information that lies in between the two diffused signals is preserved. When the difference between t_l and t_h becomes small, DoG approximates LoG. Moreover, DoG has similar message complexity to LoG. Since the heat kernels can be computed jointly, the algorithm terminates in t_h rounds and after the exchange of $2mt_h$ messages, with messages of type $msg_t(u) = [u, (\mathbf{H}_{t_h} x)(u)]$.



(a) LoG (b) LoP

(c) DoG (d) DoP

Figure 4.4: Spectral responses for representative parameters. Since heat-based filters can become negative for $\lambda > 1$, e. g., $\mathbf{LoG}_1$ and $\mathbf{DoG}_{\{3,8\}}$, we facilitate presentation by showing their absolute spectral response.

The spectral response of DoG

$$r_{\mathrm{DoG}}(\lambda;t_l,t_h) = (1-\lambda)^{t_l} - (1-\lambda)^{t_h} \tag{4.9}$$

is shown in Figure 4.4c for representative parameter values.

The main benefits of DoG are its flexibility and high resilience to noise. As we discern from its spectral response, DoG not only controls the smaller and larger sizes of detectable events independently (flexibility), but is also more aggressive in attenuating signal components (resilience). For this reason DoG generally outperforms LoG. On the flip side, choosing two parameters is more challenging than one. If the parameters are incorrectly chosen, DoG may perform worse than LoG.

4.4.2. Potential-based Filters

LoG and DoG are efficient algorithms but they depend on synchronous rounds and do not tolerate dynamics (they assume *stationarity*). This is because the heat kernel is not an attractor (i. e., not a steady state) of difference equation (4.5), but a transient state. Asynchrony and dynamics introduce disturbances that alter the system's trajectory through the state space. The disturbed and original trajectories are not guaranteed to share transient states. The synchronous constraint is not unique to LoG and DoG, but it is shared by all heat-based filters, which renders them of little use in most dynamic wireless networks. To overcome this major drawback, we propose a novel set of asynchronous bandpass filters based on the potential kernel [82]. Since the potential kernel is an attractor and not a transient state, potential-based filters are not significantly affected by dynamics (e. g., due to mobility or due to the change of the monitored signal) and are asynchronous.

Potential kernel. The potential of a signal x expressed in terms of the eigenvectors ϑ_k of P is

$$\mathbf{P}_\varphi x = \sum_{k=1}^{n} \left(\frac{1-\varphi}{\varphi} \lambda_k + 1 \right)^{-1} \langle x, \vartheta_k \rangle \vartheta_k, \tag{4.10}$$

where $\varphi \in (0, 1]$. Contrary to the heat kernel, the potential kernel converges under very challenging conditions, such as (*i*) asynchronous operation, (*ii*) partial information (only using 25% of neighborhood information at each computation round), and (*iii*) node mobility. These characteristics make potential-based filters well suited for event detection in mobile large scale networks.

Laplacian of Potential. The LoP filter computes the Laplacian after the signal has been smoothed by a potential kernel

$$\mathbf{LoP}_\varphi x \triangleq L\mathbf{P}_\varphi x.$$

As shown in Figure 4.4b, LoP has a spectral response

$$r_{\mathrm{LoP}}(\lambda; \varphi) = \frac{\lambda}{\frac{1-\varphi}{\varphi}\lambda + 1}. \tag{4.11}$$

We can gain an initial intuition on how potential-based filters compare to heat-based filters by comparing the impulse response of LoP and LoG. Figure 4.3b depicts the responses for a simple path graph of 100 nodes. Unlike LoG, in Figure 4.3a, the central lobe (convex region close to u) of LoP's impulse response is narrow

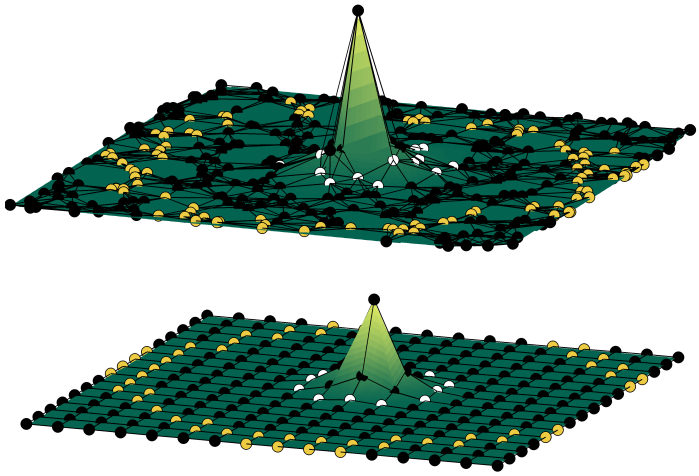


Figure 4.5: Impulse response of the $\mathbf{DoP}_{\{0.06,0.1\}}$ kernel for a grid topology (bottom) versus a random geometric graph (top). The boundaries of the smallest and largest detectable event regions are shown with white and orange markers, respectively.

and always consists of u and it's neighbors. As a consequence, LoP tends to favor abrupt signal transitions and is more sensitive to noise. Beyond the central lobe, the response fades outward with a rate that depends on φ. As φ decreases, the rate also decreases and the response widens. Though we lack an exact characterization of LoP's impulse response, we have numerically found that in the continuous domain the radius of the largest disc LoP can detect (depicted in dashed lines) is close to $14\sqrt{(1-\varphi)/\varphi}$.

As seen by its spectral response in Figure 4.4b, LoP is susceptible to the presence of small regions, which occur around $\lambda = 1$. This helps in capturing fine-grained boundaries (resolution) but renders the filter less resilient to noise. In signals with little noise, LoP outperforms LoG and exhibits a similar performance to DoG and DoP (explained next).

Difference of Potentials. Our last filter subtracts two potential kernels of different width

$$\mathbf{DoP}_{\{\varphi_l,\varphi_h\}}x \triangleq \mathbf{P}_{\varphi_h}x - \mathbf{P}_{\varphi_l}x, \tag{4.12}$$

where $0 < \varphi_l < \varphi_h \leq 1$ are user defined parameters. A network gets ϵ-close to the steady state after the exchange of $2\varphi^{-1}\log\left((c/\epsilon)\right)|\mathcal{E}|$ messages of format $msg_t(u) = [u, (\mathbf{P}_{\varphi_l}x)(u), (\mathbf{P}_{\varphi_h}x)(u)]$.

We inspect the geometry of the DoP kernel using its impulse response in Figure 4.5. Unlike LoP, DoP is more resilient to noise. Decreasing φ_l widens its central lobe and, as a consequence, the smallest event region that it can detect.

DoP's spectral response is

$$r_{\text{DoP}}(\lambda; \varphi_l, \varphi_h) = \frac{\lambda\,(\varphi_h - \varphi_l)}{(\varphi_l(1-\lambda)+1)\,(\varphi_h(1-\lambda)+1)}. \tag{4.13}$$

Similar to DoG, DoP provides flexibility for parametrization as well as resilience to noise. However unlike the first, DoP also has good resolution. That is because its spectral response does not completely eliminate small events, but only decreases their importance. In Section 4.5 we will see that, in most scenarios, DoP outperforms all other algorithms.

4.4.3. Comparison

First, it is important to mention that all four filters eliminate the flat-region ambiguity problem. By removing the signal component that corresponds to λ_1, filters eliminate a signal's DC-offset. The sign of the resulting signal reveals whether a region is convex or concave. It is also important to note that, our filters work in 3D as well as in 2D topologies. As reported by Chung et al. [25], graph filters can be used for graphs embedded in any dimension.

We next compare the four proposed filters in terms of their resilience to noise and resolution. To have a better understanding of this comparison it is important to recall that: (*i*) eigenvalues are rarely higher than 1.2, and (*ii*) eigenvalues around 1.0 capture high-order events (small regions).

Resilience. This is the ability of a detector to attenuate small events, which can be caused by noise. As we saw in Section 4.3, the expected size of event regions described by the k-th signal component is larger than n/k. Therefore, noise will be mainly decomposed into high order components. A filter is thus resilient to noise if its spectral response decreases sufficiently fast after its peak response ($\lambda_{peak} = \arg\max_\lambda r(\lambda)$). In fact, we can order detectors with respect to their resilience by looking at the magnitude of $\partial r / \partial \lambda_k$ for $\lambda_k > \lambda_{peak}$.

$$\text{DoG} > \text{DoP} > \text{LoG} > \text{LoP}. \tag{resilience}$$

The derivation easily follows from Formulas (4.7), (4.9), (4.11), and (4.13). Summarizing, due to their sharp responses, DoG and DoP are better at removing noise.

Resolution. The resolution of a filter depends on its ability to capture details, which are encoded in high order signal components—but usually not as high as noise. Since in most graphs high order signal components are concentrated around $\lambda = 1$, we quantify the resolution of a detector through the ratio of its response at $\lambda = 1$, normalized to its peak response, $\lim_{\lambda \to 1} r(\lambda)/r(\lambda_{peak})$. The resulting ordering is

$$\text{LoP} > \text{DoP} > \text{DoG} \approx \text{LoG}. \qquad \text{(resolution)}$$

Notice that a filter with high resilience is not necessarily characterized by low resolution. DoP attains both by having a sharp response near λ_{peak} that flattens out for higher λ, but still decreasing monotonically. Heat-based filters on the other hand suffer from low resolution; their response is always zero at $\lambda = 1$. When no noise exists, LoP consistently outperforms the three other algorithms.

4.5. Evaluation

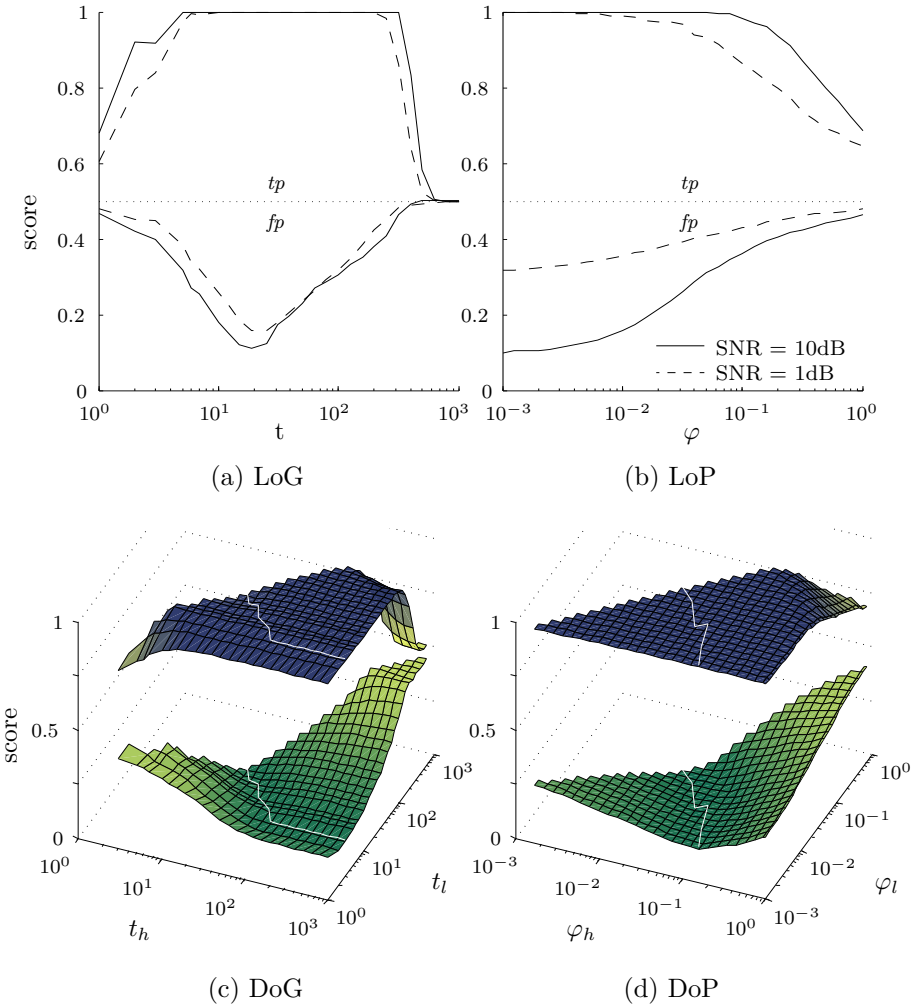
Our evaluation consists of three parts. First, we compare the performance of the proposed algorithms with respect to their resilience, resolution, and sensitivity to parametrization. To achieve a systematic comparison, this part ignores the effect of irregular topologies. We then focus on signal and network dynamics and evaluate their effect on detection. Last, we perform a testbed experiment which demonstrates the feasibility and simplicity of our approach.

Metrics. We quantify detection accuracy in terms of the true positive and false positive scores (tp, fp). The true positive score –also called the detection probability– measures the proportion of event points (nodes) which are correctly identified. The false positive score –also called the false alarm probability– expresses the proportion of non-event points which are incorrectly identified as part of an event region. Ideally, $tp = 1$ and $fp = 0$.

4.5.1. Simulation Results

Before we go on to illustrate the ability of potential-based filters to track dynamics, let us focus on the detection performance of the filters themselves.

Detection performance. Our first experiment considers a grid topology of 400 nodes and a star-shaped event region corrupted with additive Gaussian noise. The star event occupied approximately 25% of the area of the network and it was po-



(a) LoG (b) LoP

(c) DoG (d) DoP

Figure 4.6: Exhaustive comparison of event detectors for a simple grid topology and a star-shaped event region. Each graph depicts the true positive rate (top curve/surface) versus the false positive rate (bottom curve/surface). The white line in the bottom two figures annotates the best performing parameter combinations.

sitioned at its center. Notice that the choice of shape influences the comparison. While filters with high resilience perform better with simple events (such as circles), filters with high resolution favor events with complex boundaries. The star is a good benchmark because it sits in the middle of these extremes. The simulations were performed in MATLAB.

The results –see Figure 4.6– confirm the higher resilience but lower resolution of heat-based filters (as per our analysis). This is particularly noticeable in subfigures 4.6 (a-b). While for low SNR LoG outperforms LoP, the accuracy of the second is higher when SNR=10 dB. For DoG and DoP the trend is slightly different. Subfigures 4.6 (c-d), which have an SNR=1 dB, show that DoP is better than DoG. This is because a noisy star event favors DoP slightly. For coarser-shaped events (not shown due to lack of space), DoG performs better than DoP. *Also observe that potential-based filters are less sensitive to parametrization.* As seen in subfigure (b), LoP's accuracy consistently improves as φ decreases. This is because the size of the smallest regions it detects is independent of φ. DoP's performance on the other hand degrades when the parametrization is suboptimal, but –unlike DoG– it does so gracefully. As long as the size of the largest identifiable region stays larger than the size of the star ($\varphi_l < 0.1$) all of the event points are recognized. False positives occur when the size of the smallest identifiable event region becomes smaller than the size of noise regions ($\varphi_h > 0.1$).

Dynamics. We next examine how our best detector (DoP) performs in more challenging scenarios. We pose two main challenges: node mobility and signal dynamics. In addition, we use complex events. The test events are digitized versions of the Deepwater Horizon oil spill forecast obtained from ten numerical models [77] and contain regions of various sizes and detail. Figure 4.1 shows a snapshot of a simulation run. Due to their variability, the accurate detection of oil spills demands both resolution and resilience over a large range of scales. As is the case in most real-world scenarios, we selected values according to the guidelines of Figure 4.6 and not using exhaustive search. In our simulations, $n = 1000$, the node degree varied between 1 and 78, the network diameter was between 14 and 16, and the SNR = 5 dB[4]. To support mobility, simulations were performed in Lapsang[5], an agent-based discrete-time simulator for mobile wireless networks developed by Neils Browers. All mobility traces were generated using the BonnMotion Mobility Scenario Generation and Analysis Tool [3].

Mobility. To study how the performance degrades when nodes are moving, we simulated node mobility using the random waypoint model and varied node speed between 0 and 20 m/s. Figure 4.7 compares the online performance of DoP to that obtained when solving the problem offline for each network snapshot (offline DoP

[4]A visual illustration of the topology and of the dynamics in question can be found in the video mentioned in the caption of Figure 4.1.

[5]`https://code.google.com/p/lapsang/`

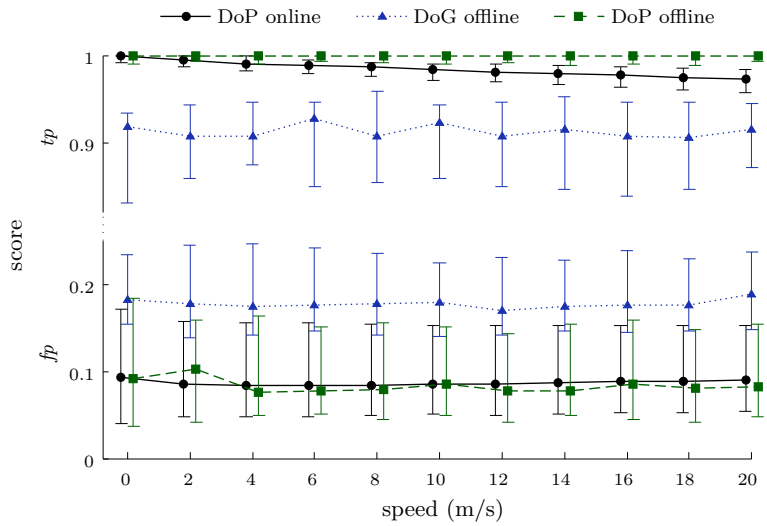


Figure 4.7: Detector accuracy versus node speed in the random waypoint mobility model. Despite the mobility, DoP's online performance is similar to the offline solution. Both are significantly better than DoG. Test signals were generated from the Deepwater Horizon oil spill dataset [77].

and DoG). For each algorithm, we plot the quartiles of *tp* and *fp* scores over 10 runs. *Overall, DoP outperformed DoG.* While DoG exhibited low performance due to its low resolution, DoP detected the oil spills accurately, even for high speeds. *Mobility undoubtedly introduces a transient error that depends on speed.* The phenomenon occurs because nodes do not realize that they enter or exit an event region until they re-converge ϵ-close to the new steady state. *Nevertheless, DoP's accuracy deteriorates slowly as the speed increases.* This is attributed to two main reasons. First, while convergence requires $1/\varphi_l \log(c/\epsilon)$ rounds, ϵ should not necessarily be small. It is sufficient that the sign of $(\mathbf{DoP}_\varphi)(u)$ changes. Second, mobility acts as a spatial low-pass filter that reduces noise. For nodes that remain within the same region (event or non-event), the likelihood that a node is affected by noise is smaller when the node is moving.

Signal dynamics. To evaluate how potential-based filters cope with signal dynamics, we inverted the event and non-event regions ($x = -x$) twice, once at $t = 10$ and once more at $t = 20$ seconds. The inversion stresses the ability of a detector to adapt as it instantaneously cancels the validity of any previous detection. Figure 4.8, shows the min, median, and max scores across time. *We can see that DoP quickly overcomes any dynamics induced by the signal.* Even in the worst case, the

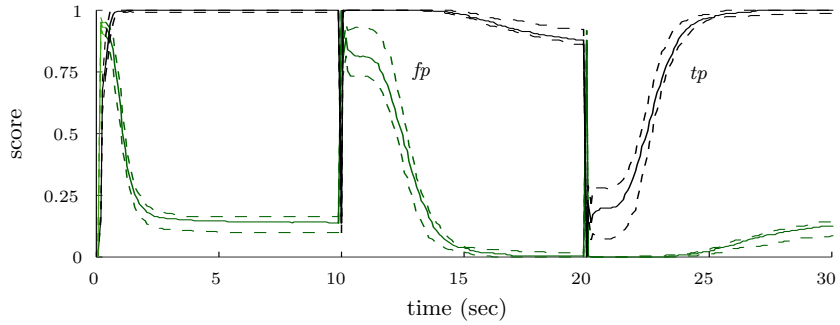


Figure 4.8: Potential-based filters are very robust to signal dynamics. To demonstrate, we invert the signal at $t = 10$ and $t = 20$ seconds. The median (line) as well as the min and max scores (dashed lines) show how DoP quickly overcomes the event region inversion.

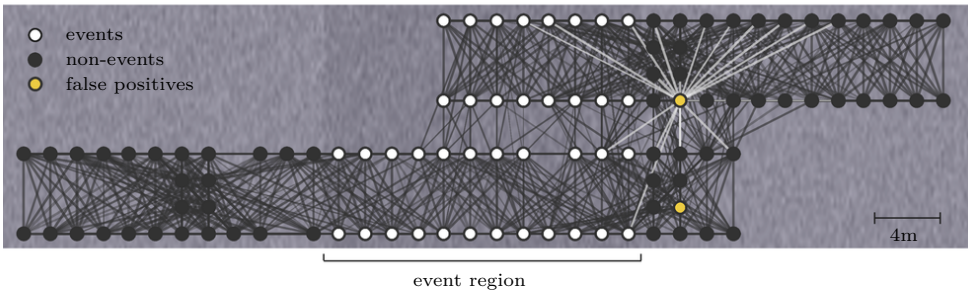


Figure 4.9: Graphical depiction of the testbed's topology, annotated with the results of $\mathbf{DoP}_{\{0.1,0.2\}}$. Black/white nodes are the nodes that correctly identified the event/non-event (dark gray area in the middle). Orange nodes are false positives. In this instance, SNR=5dB and the score is (1, 0.059).

steady state is reached after about 10 seconds. Similar performance was achieved by LoP.

4.5.2. Empirical Results

We evaluated DoP in a testbed of 105 resource constrained devices (MSP430 microcontroller, CC1101 radio), deployed on the ceiling of our building at TU Delft. DoP was implemented in Contiki OS, over a simple asynchronous MAC protocol (NullMAC). For the duration of each computation round (0.5 second), each device transmitted an average of 3.5 packets. We simulated the sampling process, by constructing a virtual signal and then assigning to each node the value underneath it—see Figure 4.9. We repeated the experiment for three parametrizations, two

	$\mathbf{DoP}_{\{0.05,0.15\}}$		$\mathbf{DoP}_{\{0.1,0.2\}}$		$\mathbf{DoP}_{\{0.15,0.2\}}$	
	tp	*fp*	*tp*	*fp*	*tp*	*fp*
SNR=1dB	1	0.14	0.95	0.15	0.92	0.18
SNR=5dB	1	0.08	1	0.08	1	0.09

Table 4.1: The (*tp*, *fp*) scores achieved by DoP in our testbed. Each score is the median over five experiments, each with a different signal.

noise levels, and five different signals. DoP's median (*tp*, *fp*) scores are summarized in Table 4.1.

Observe that the empirical scores are similar to those achieved in simulations. *Our empirical evaluation not only showcases the ease of implementation of potential-based filters (≈ 25 lines of code), but also their robustness to the vagaries of wireless communication.* Low-power wireless links are known to have high temporal and spatial variability, to be asymmetric and to have clustering effects. These combined effects lead to complex and continuously changing topologies. Furthermore, packet losses are commonplace and affect the performance of any protocol requiring one-hop information. We would like to note that most false positives occurred in the ambiguous zone near the boundary. Figure 4.9 depicts such an example. In the figure, both false positives were exterior boundaries with strong links toward the event region. As such, they were "pulled" to higher values by their neighbors within the event region.

4.6. Conclusions

This chapter proposed decentralized filters for detecting event regions and their boundaries on mobile networks. These are the first local, location-free detectors that do not assume signal stationarity. In addition, we provided novel insight on the problem of event detection. We showed that, by examining how an event detector affects the signal decomposition in the spectral domain (i. e., its spectral response), we decouple its effect from the network topology. Using this idea, we characterized the performance of our novel filters (LoP and DoP) and compared them to established image processing filters (LoG and DoG). Our algorithms exhibit good resolution and resilience. They are also efficient, simple, and tolerant to mobility and signal dynamics.

The fact that our filters are parametric, poses an important question: "*What are the most suitable parameters for a given deployment?*". This chapter provided general guidelines, but not an exact methodology for choosing parameters; the question needs to be investigated further.

5

Scale-Space Theory on Graphs

Identification of Peaks and Pits

"You live and learn. At any rate, you live."

—Douglas Adams,
Mostly Harmless

GRAPH filters are a recent and powerful tool to process information in graphs. Yet despite their advantages, graph filters are limited. The limitation is exposed in a filtering task that is common but not fully solved in sensor networks: the identification of a signal's peaks and pits. Choosing the correct filter necessitates *a-priori* information about the signal and the network topology. Furthermore, in sparse and irregular networks graph filters introduce distortion, effectively rendering identification inaccurate even when signal-specific information is available. Inspired by research in computer vision, we derive a family of scale-space kernels for graphs

Parts of this chapter have been published in IPSN'15, April 2015, Seattle, US [79].

that gather information across all possible *scales of observation*: from fine to coarse. The gathered information is then used to distributedly identify the signal's peaks and pits. Our scale-space approach diminishes the need for *a-priori* knowledge, and reduces the effects caused by sparse and irregular topologies exhibiting: (*i*) superior accuracy to the state-of-the-art in the presence of noise, and (*ii*) at least 20% higher precision than the best graph filter, when evaluated on our testbed.

5.1. Introduction

Recently, there has been a surge of research focusing on the processing of graph data. One of the breakthroughs of the community has been the design of *graph filters*, distributed algorithms with applications to sensor, transportation, social and biological networks [100, 109]. Similar to how classical filters operate on time signals and images, graph filters operate on *graph signals*, i. e., signals defined on the nodes of irregular graphs [110]. One of the benefits of graph filters is that they allow one to observe graph data at different scales. For example, Figure 5.1 shows that a signal filtered with a low scale parameter ($s = 0$), exposes fine details, while coarse signal trends are observed at higher scales ($s = 14$). Based on the scale parameter, a low-pass graph filter controls the size of observable signal structures, attenuating structures of small size, such as noise. Beyond noise removal (cf. Chapter 3), graph filters are useful for revealing communities (low-pass) [113], identifying event-regions (band-pass) (Chapter 4), and detecting anomalies (high-pass) [100].

Yet, despite their theoretical guarantees and distributed computational efficiency, graph filters are also limited. A common task that exposes their shortcomings is the identification of the *peaks and pits* of a graph signal. Beyond giving us insights about the signal itself, the peaks and pits of a signal appear recurrently in a wide range of applications in sensor networks. Peaks and pits are implicitly used by event and target tracking algorithms [10, 12, 39, 57] and form the basis of topological methods for signal mapping and compression, such as surface networks [58], iso-contour maps [102], and Morse-Smale complexes [133]. Furthermore, peaks play a central role in gradient-based routing and navigation [41, 82], where a discovered path is only useful if it leads to a true peak.

On the surface, identifying the peaks and pits of a signal appears deceptively simple: a node is at the summit of a peak if its value is the largest amongst its neighbors (local maximum). Equivalently, a node is at the bottom of a pit if its value is the smallest amongst its neighbors (local minimum). In practice however, the accurate identification of peaks and pits is challenging [57, 58]. The challenge

arises due to two key problems. First, extrema are inherently tied to the local signal derivative and thus notoriously sensitive to noise. Second, extrema are affected by how the network is connected, and they occur more often in sparse irregular networks. For these two reasons, graph signals often contain *false extrema*—maxima and minima that do not correspond to the real peaks and pits of the physical signal.

Though eliminating false extrema is a filtering problem, graph filters exhibit several drawbacks:

The correct scale. First, the filtering efficiency depends on the correct choice of scale. For Figure 5.1, this drawback maps to the following question: what scale gives the most truthful representation of the underlying signal? To choose the scale of observation correctly, one must have *a-priori* information about the observed phenomenon, as well as of the instrument of observation—in our case, the network topology; information which is rarely available and often changes over time.

Loss of information. Second, even the correct choice of scale results in loss of information. Every scale conveys useful information about a signal: coarse scales describe large structures, whereas fine scales reveal details. Enforcing a single scale of observation can lead us to ignore valuable information.

Phantom effects. Third, this chapter shows that filtering over *irregular* graphs, such as those found in real wireless networks, can cause *phantom extrema*, i. e., extrema that are not present on the signal, but are an artifact of the filtering process. These phantom extrema severely hamper identification in practice, even when the scale is chosen correctly.

5.1.1. Related Work

In classical signal processing, one of the standard approaches to overcoming the limitations of filtering is the *scale-space approach* [122]. According to scale-space theory, we can only capture the full set of features present in a signal if we examine it across all possible scales of observation. Scale-space approaches have been proven invaluable in image processing. They are widely used for extracting image features, such as extrema, saddles, and corner pixels, as well as for image smoothing and edge detection [74, 128]. The central question of scale-space theory is identifying the scale-space kernel (or filter) that abides to a set of conditions, commonly referred to as scale-space axioms [75]. We currently know that, in both continuous and discrete settings, the only kernel that abides to the scale-space axioms is the heat

kernel. And even though there are indications that the same kernel also applies to graphs [65, 128], a rigorous examination of scale-space theory on graphs is -to this point- missing. We prove that contrary to classical scale-space theory, in graphs the heat kernel is not the only kernel that abides to the scale-space axioms; others do too. We also propose an efficient way to compute them in a distributed network, which is especially relevant in wireless sensor networks, due to the limited time and resources available to communicate and process sensor data.

We have to note that, though prevalent in image processing, scale-space is not the only approach to identify peaks and pits in sensor networks. The first to recognize that false extrema exist (also referred to as weak peaks/pits) were Jeong et al. [57, 58]. The authors showed that inferring the properties of a physical signal from its discretization is hard, regardless of the discretization density. Moreover, they proposed a method for identifying signal peaks and pits. Our evaluation shows that their algorithm does improve identification accuracy, but only to a limited extent. When the network is not well-connected (small average degree, irregular links) or the signal is noisy, the identification is inaccurate.

5.1.2. Contributions

We provide three main contributions:

Contribution 1. *Extending scale-space theory to graphs—Section 5.3.* We address two fundamental questions: *What are the scale-space kernels that are appropriate for graphs and how efficiently can we compute them distributedly?* To answer this question, we first identify the properties that graph scale-space kernels must have, and then, evaluate the pros and cons of three candidate kernels. We also show that in practice, *synchronous* implementations of graph scale-space kernels seem to be the only viable option in terms of time and message-size complexity. Our analysis suggests that synchrony is a fundamental requirement of scale-space kernels for graphs—in the sense that currently known asynchronous algorithms exhibit higher complexity. Last, we show that scale-space kernels are essentially graph filters. Our insight allows us to draw interesting connections between scale-space theory and signal processing on graphs.

Contribution 2. *Using the scale-space approach to identify signals' peaks and pits—Sections 5.4-5.5.* Similar to images, instead of selecting a single scale, we observe a signal at every possible scale and use the combined information to identify the peaks and pits [75]. Intuitively, this method follows a *survival of the fittest*

approach, where the longer a peak (or pit) survives across scale, the higher the likelihood of being identified as a 'true' feature. Overall, the scale-space approach entails three steps: (*i*) Using a scale-space kernel, we progressively simplify the sensed signal (see Figure 5.1). The sequence of scaled signals, each simpler than the previous, gives rise to the signal's *scale-space*. (*ii*) All the while, we track how the simplification changes the signal extrema. The captured information, that is referred to as the signal's *deep structure* [67], contains the location and lifetime of each extremum across scale. (*iii*) We use the deep structure to discern whether an extremum is true, false or phantom, essentially identifying the peaks and pits of the underlying signal. All three steps are computed locally within the network, require no location information, and incur a computational overhead similar to that of a graph filter.

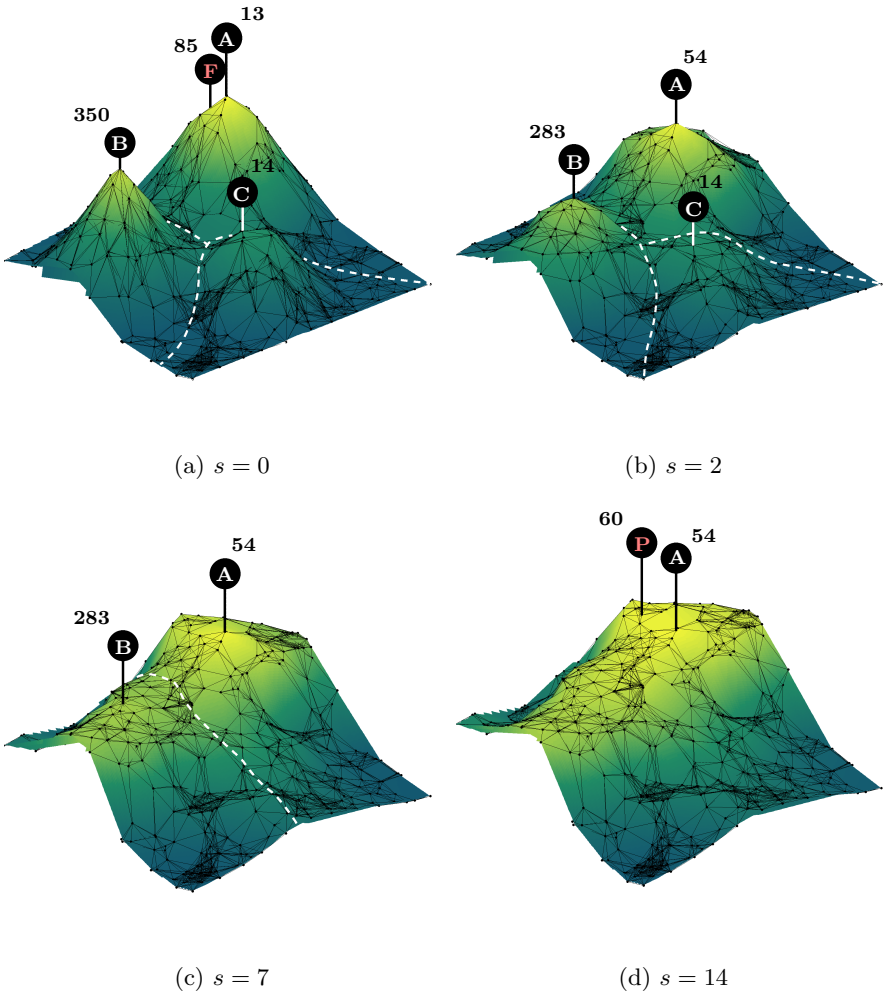
Contribution 3. *Defining the challenges of peak identification in real-world sensor networks—Section 5.6.* We implemented both state-of-the-art and scale-space algorithms in a testbed consisting of 100 wireless sensor nodes, and used the gathered information to benchmark the accuracy of each method. To the best of our knowledge, we are the first to evaluate such mechanisms in real-word scenarios, thus including radio-specific effects like irregular coverage, asymmetric links, packet loss and temporal link variability. We discovered that these phenomena have a drastic impact on the accuracy of peak/pit identification and should be considered in the evaluation of future mechanisms. Moreover, we show that the superior resilience of scale-space methods to radio effects makes our approach far more precise than the state-of-the-art and at least 20% as accurate as the most accurate graph filter.

5.2. Preliminaries

Before delving into scale-space theory, we start by describing the problem of peak and pit identification in sensor networks and discussing its main challenges. Section 5.2.2 then presents an overview of our approach.

5.2.1. Peak and Pit Identification

Consider a sensor network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of n nodes and m links which is monitoring its environment. Each node $u \in \mathcal{V}$ is situated in some (possibly unknown) physical location, and samples a physical signal present in a Euclidean space and imbued with noise of unknown characteristics, such as mean, variance, and type. The sensed



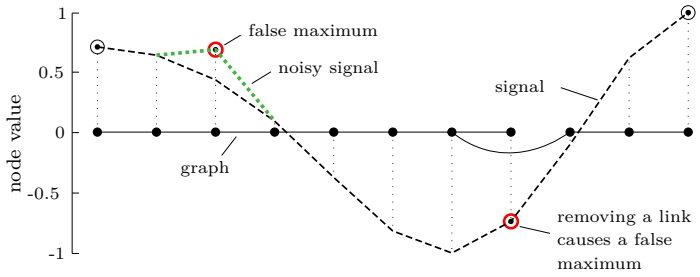
(a) $s = 0$ (b) $s = 2$

(c) $s = 7$ (d) $s = 14$

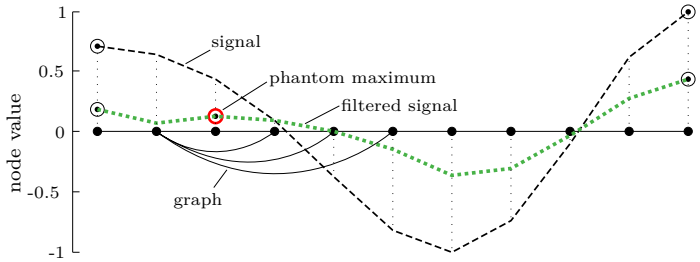
Figure 5.1: The maxima of a graph signal across four different scales s. A number indicates the id of each maximum. The three peaks are indicated with letters A-C. F and P indicate a false and a phantom extremum, respectively. The sequence of signals across all scales gives rise to the signal's scale-space.

information is captured by a time-invariant[1] graph signal $x : \mathcal{V} \to \mathbb{R}$, which assumes a real value $x(u)$ on each node u. Our objective is to identify the peaks and the pits of the underlying physical signal.

[1]The time-invariance assumption is made for ease of presentation; as long as the signal changes slow enough with respect to the sampling rate and communication bandwidth, our results also apply.



(a) False extrema due to noise (left) and sparsity (right).



(b) Phantom extrema are artifacts of filtering.

Figure 5.2: Even slight changes to the connectivity of a path graph (back solid lines) and to the values of the physical signal (black dotted line) can cause false and phantom extrema to appear. On the top, the noisy sensed signal (green dotted line) contains two false maxima, caused by noise and irregular connectivity. On the bottom, the filtered signal (green dotted line) contains a phantom maximum.

Challenges. Identifying the extrema of the sensed graph signal x is easy: a node u is a *maximum* if $x(u) > x(v)$ for all nodes $v \sim u$ in its communication vicinity. Correspondingly, u is a *minimum* if $x(u) < x(v)$ for all $v \sim u$. Nevertheless, the maxima and minima of x do not necessarily correspond to the peaks and the pits of the underlying physical signal. In Figure 5.1(a) for example, there are four extrema but only three peaks (A,B,C). These mismatches occur for three main reasons, the first two reasons are well known in the community, while the last is an insight obtained through our work:

Noise. Whether because of spatio-temporal perturbations or sensor imprecision, signals can fluctuate significantly around their original value, leading to false positives and false negatives (cf. Figure 5.2(a)).

Graph irregularities. It is commonly assumed that whenever two nodes are placed within a given radius, the nodes are joined by a valid link. In practice, links exhibit high spatial and temporal variability, leading to false positive extrema: a low-valued node connected to a high-valued node would not select itself as an extremum, but if the link disappears, it will (cf. Figure 5.2(a)). This phenomena is pervasive in real networks and it is particularly acute in sparse graphs. Proposition 5.1 below shows that the expected number of extrema arising from volatile links grows with an exponential trend as the graph becomes sparser[2]. This trend is later observed in our evaluation.

Proposition 5.1. *The expected number of maxima (minima) X appearing if k out of m edges of a random geometric graph are deleted at random is* $\mathrm{E}[X] > n\,(e^{k/m} - e^{-m/n} - \epsilon)$*, where ϵ becomes negligibly small when $k > m/n$.*

Proof. The event that a node u becomes an extremum after deleting k edges is described by an indicator random variable $X_u = \{0, 1\}$. X_u depends on the number Y_u of neighbors $v \sim u$ with $x(v) \geq x(u)$ (correspondingly $x(v) \leq x(u)$). Because u becomes a maximum (minimum) if all edges to neighbors with larger (smaller) values are deleted, we have that

$$\mathrm{E}[X_u = 1 | Y_u = y] = \mathrm{P}(X_u = 1 | Y_u = y) = \binom{k}{y}\binom{m}{y}^{-1}.$$

Moreover, from independence, the expected number of appearing maxima (minima) X is

$$\mathrm{E}[X] = \sum_{u \in \mathcal{V}} \mathrm{E}[X_u] = n \sum_{y=1}^{k} \mathrm{P}(X_u = 1 | Y_u = y)\mathrm{P}(Y_u = y).$$

Since Y_u is always smaller than the node degree, we obtain a lower bound by substituting Y_u with the degree random variable D. In particular, for a random geometric graph with average degree m/n,

$$\begin{aligned}\mathrm{E}[X] &> n \sum_{i=1}^{k} \binom{k}{i}\binom{m}{i}^{-1} \frac{(m/n)^i e^{-m/n}}{i!} \\ &= n\,(e^{k/m} - e^{-m/n} - \epsilon),\end{aligned}$$

where $\epsilon = 1 - \Gamma(k, m/n)/\Gamma(k, 0)$ is negligibly small for $k > m/n$ and Γ is the incomplete gamma function. ■

[2]Note that the number of maxima (minima) cannot grow infinitely large: it is at most equal to the size of the graph's maximal independent set.

Phantom effects. Broadly speaking, graph filters alter a signal by diffusing it locally. The filtering process is thus determined by the underlying connectivity. If a regular lattice is used –as is common in classical signal processing– no bias is introduced by the topology in the filtering process. In real deployments however, the node density varies across the network and highly dense areas tend to 'accumulate' a higher share of the diffused mass. This effect creates phantom extrema: extrema that are neither present on the underlying physical signal nor in the sensed signal, but they are an artifact of filtering over an irregular topology (cf. Figure 5.2(b)).

5.2.2. The Scale-Space Approach

In this chapter, we use scale-space theory to identify the peaks and pits of a sensed signal. Our approach boils down to three steps, each computed distributedly within the network:

Step 1. *Computing a signal's scale-space.* The central idea of scale-space is that we can learn more about a signal by systematically examining it across different scales of observation: coarse scales give us the big picture (Figure 5.1(d)), whereas in fine scales details prevail (Figure 5.1(a)). More concretely, the scale-space of a graph signal x consists of a sequence $\{y_s\}$ of scaled signals, each simpler than the previous. The scale-space is constructed by filtering x with a family of $n \times n$ kernels $\mathbf{K}_s$ as

$$y_s = \mathbf{K}_s x, \tag{5.1}$$

where the *scale parameter* s denotes (roughly) that information is diffused within the s-hop neighborhood. Scale determines the maximum size of structures that can be observed. In Section 5.3, we identify appropriate scale-space kernels $\mathbf{K}_s$ and give local algorithms for their computation.

Step 2. *Tracking extrema across scale.* As a signal is progressively simplified, its extrema evolve. Peak B for instance, is born on node 350 in Figure 5.1(a), moves to node 283 in Figure 5.1(b), and dies in Figure 5.1(d). This process is compactly captured by the signal's *deep structure*, which visualizes the scale-trajectory of each extremum e (see Figure 5.3). The distributed computation and analysis of the deep structure is given in Section 5.4.

Step 3. *Extremum selection.* In the last step, we use the trajectory of each maximum (minimum) to infer the nature and importance of the corresponding peak

(pit). We focus on two aspects: First, the extremum's *lifetime*, which is defined as the length of the scale-period over which e exists. Second, we focus on *birth events*. As explained before, in contrast to images, graph filters distort signals in a way that is unique to the graph topology, leading to phantom extrema. In Section 5.5.2 we will see that phantom extrema are easy to spot from the deep structure because they are always born in large scales.

5.3. Scale-Space Theory on Graphs

This section sets the necessary theoretical basis of scale-space analysis in graphs. Though the ideas are rooted upon the scale-space theory for continuous signals and images [72, 75], our analysis deviates from the original. In contrast to the classic setting, graphs have irregular connectivity. Additionally, in networks each node can only directly exchange information with its neighbors. This motivates us to ask: *What type of scale-space kernels are appropriate for graphs and how efficiently can we compute them distributedly?*

We address this question in three steps: *In Section 5.3.1, we focus on the existence of graph scale-space kernels.* We derive necessary and sufficient conditions for a matrix to be a scale-space kernel and we present three candidates kernels. *In Section 5.3.2, we focus on computation.* We show that scale-space kernels are locally computable if synchrony is assumed, but that currently known practical asynchronous algorithms are not local. *Last, in Section 5.3.3 we expose the connection between graph scale-space kernels and graph filters.* We show that each of the candidate graph scale-space kernels forms in fact a one-parameter family of graph filters. The observation gives insight into the operation of kernels and to their relation to the graph spectrum.

5.3.1. Axiomatic Scale-Space Theory

The objective of scale-space theory is to provide a scale-invariant observation of a signal x. This is achieved by diffusing the signal with a family of kernels $\mathbf{K}_s$, where the *scale parameter* s determines the size of structures observed. But which kernel should one use? One of the breakthroughs of the scale-space community has been the axiomatization of the theory [75]. A scale-space kernel is then the one that satisfies the three main scale-space axioms[3]:

[3] Though different combinations of axioms have been used in the literature [75], the three mentioned here suffice for the characterization of graph scale-space kernels.

Axiom 5.1 (Linearity)**.** *Kernels $\boldsymbol{K}_s$ are linear operators.*

Axiom 5.2 (Semi-group property)**.** *The family of scale-space kernels forms a* semi-group*:* $\mathbf{K}_{s_1}\mathbf{K}_{s_2} = \mathbf{K}_{s_1+s_2}$.

Axiom 5.3 (Non-enhancement)**.** *The absolute value of any extremum in* $\mathbf{K}_s x$ *must always decrease in* $\mathbf{K}_{s+1}x$.

The first axiom basically asserts that a scale-space kernel is not signal specific, in the sense that it abides to the superposition principle: for any two signals x_1 and x_2, $\mathbf{K}_s(x_1 + x_2) = \mathbf{K}_s x_1 + \mathbf{K}_s x_2$. Additionally, a scale-space kernel should act in the same way at all scales. If the semi-group property is not met, diffusion deforms the signal in a scale-specific way, which is undesirable. Last, according to the non-enhancement axiom, a kernel must always simplify existing signal extrema. Note that guaranteeing the simplification of *existing* extrema at higher scales (Axiom 5.3) is not the same as guaranteeing that no new extrema appear. Signals often contain hidden structures, which diffusion reveals. For example, consider two peaks joined by a thin bridge. If the lowest point of the bridge is slightly taller than the lowest peak, then the signal has only one maximum. The second maximum is however revealed when, due to diffusion, the bridge collapses. Thus, non-enhancement represents the weaker alternative of guaranteeing that existing extrema must be always simplified.

Even though the fundamental question of scale-space theory –*which scale-space kernel satisfies the scale-space axioms?*– has been answered for the continuous and discrete settings [72, 75], the answer does not directly apply to graphs. Graphs exhibit irregular connectivity and are not, in general, a metric space. This poses an analytical challenge as, unlike images, differentiation is impossible. These differences motivate us to address scale-space theory from a graph-theoretic perspective. In the following, we show that a graph scale-space kernel $\mathbf{K}_s$ that satisfies the scale-space axioms exists and that it is connected to the graph spectrum.

Central to our discussion is the notion of an h-local matrix. Intuitively, an h-local matrix is an operation that uses signal information at most h hops away from each node.

Definition 5.1 (h-local matrix)**.** *An arbitrary matrix M is h-local if, for all u_i and u_j in $\mathcal{V}$ with shortest-path distance $d(u_i, u_j) > h$, $M_{ij} = 0$.*

We follow with our main result.

Theorem 5.1. *A kernel $\mathbf{K}_s$ satisfies the scale-space axioms if and only if $\mathbf{K}_s = S^s$, where the scale-space matrix S is 1-local and non-negative.*

Proof. We start by noticing that, for any kernel satisfying Axioms 5.1 and 5.2, it must be that $\mathbf{K}_s = S\mathbf{K}_{s-1}$, where $S = \mathbf{K}_1$. Applying this recursively, we obtain $\mathbf{K}_s = S^s$. The conditions that S is 1-local and non-negative are imposed by Axiom 5.3 and are given in Lemmas 5.1 and 5.2, respectively. ■

Lemma 5.1. *Kernel $\mathbf{K}_s = S^s$ satisfies the non-enhancement axiom only if S is a 1-local matrix.*

Proof. We show by method of contradiction that, kernel $\mathbf{K}_s = S^s$ satisfies Axiom 5.3 only if S is 1-local. Any matrix S is always h-local for h equal to the network diameter; the question is whether $h = 1$. For sake of contradiction assume that $h > 1$. We show that, if this is true, a signal x always exists (independently of $\mathcal{G}$) that violates the non-enhancement axiom. Therefore the assumption is incorrect and $h = 1$. The construction is as follows: Choose two nodes u_i, u_j with $d(u_i, u_j) = h > 1$ such that $S_{ij} \neq 0$. Assign values $x(u_i) = 1$ and $x(v) = 0$ otherwise. Clearly, u_i is a maximum and its value should decrease in in the next iteration $(Sx)(u_i)$. Nevertheless, a real number $\beta \gg 1$ always exists for which, if $x(u_j) = \beta \operatorname{sign}(S_{ij})$, then $(Sx)(u_i) > 1$, which is a contradiction. ■

Lemma 5.2. *Kernel $\mathbf{K}_s = S^s$ satisfies the scale-space axioms if and only if S is a 1-local and non-negative matrix.*

Proof. We begin by establishing that, if the two conditions hold, the scale-space axioms are satisfied. Let u_i be a maximum of x. By definition, its neighbors u_j have strictly smaller values. It is easy to see that, if the non-negative condition $S_{ij} \geq 0$ holds, then $(Sx)(u_i) \leq x(u_i)$, which is exactly the non-enhancement axiom. As we can see, when S is 1-local and non-negative it always satisfies the three axioms. The two proposed conditions are therefore sufficient. As we show next, since the opposite is not always true, the two conditions are also necessary. The proof is done by contradiction: if we assume that $S_{ij} < 0$, we can always construct a signal that breaks the non-enhancement axiom. That is simply by assigning $x(u_i) = 1$, and for each neighbor u_j with $S_{ij} < 0$, $x(u_j) = -\beta$, where $\beta \gg 1$. The non-negativity of S's diagonal guarantees that extrema do not oscillate between being maxima and minima as scale increases. ■

kernel	name	S	symmetric	column-st.	row-st.
$\mathbf{H}_t$	heat	$I - \mathcal{L}$	yes	no	no
$\mathbf{T}_t$	random-walk	T	no	yes	no
$\mathbf{P}_t$	consensus	P	no	no	yes

Table 5.1: Candidate graph scale-space kernels, along with the properties of their scale-space matrices S. Row/column-st refers to row- and column-stochasticity, respectively.

Remark. Though it is not requested by the scale-space axioms, it also useful to impose that $\|\mathbf{K}_s x\|$ never grows infinitely large. For this reason, in the following we only consider scale-space matrices S with spectral radius smaller or equal to one.

Candidate kernels. Though Theorem 5.1 establishes the properties of graph scale-space kernels, it does not provide an explicit form. In other words, many possible kernels may exist that satisfy the scale-space axioms. We have identified three such candidate kernels (cf. Table 5.1): The first is the *heat kernel* $\mathbf{H}_s = (I - \mathcal{L})^s$, where $\mathcal{L}$ is Chung's normalized Laplacian matrix [26]. Additionally to satisfying the conditions of Theorem 5.1, the heat kernel is symmetric, which suits applications working over undirected graphs. In wireless networks however this property is not satisfied.

The next two kernels under consideration are asymmetric. But to validate their suitability for peak identification, we need to evaluate two important properties: *column-* and *row-stochasticity*: (*i*) Column-stochasticity warrants that the mass of x remains constant as it is being diffused, i. e., $\mathbb{1}^\top \mathbf{K}_s x = \mathbb{1}^\top x$ for all s, where $\mathbb{1}$ is the all ones vector. The random walk kernel $\mathbf{T}_s = T^s = (AD^{-1})^s$, where D is the diagonal degree matrix and A the adjacency matrix, is column stochastic. For our purposes however column-stochasticity is not a desirable property, because the distribution of mass is strongly biased towards well connected nodes. Considering the high irregularity of wireless networks, column-stochasticity would exacerbate the phantom extrema effect. On the other hand, this kernel is suitable for purely graph-based signals, such as web page centrality, and it has also been used for graph partitioning by Chung [22]. (*ii*) Row-stochasticity governs the behavior of a signal at very large scales. Consider the *consensus kernel* $\mathbf{P}_s = P^s = (D^{-1}A)^s$, our third scale-space kernel. Being row-stochastic, the consensus kernel flattens signals completely as $s \to \infty$. This property is particularly useful for filtering physical signals, such as the measurements of a sensor network, because it progressively eliminates (smooths) structures based on their size. Given that the consensus kernel

$\mathbf{P}_s$ has the required properties for peak and pit identification, we used it as the default kernel for the rest of this chapter.

5.3.2. Distributed Computation

Having established that a graph scale-space kernel exists, we proceed to examine how it can be computed efficiently in a distributed network. We show that the kernel is computed in constant time and message size if all nodes are synchronized. In the asynchronous case, currently known practical algorithms are non-local.

Computational models. For convenience, we will assume that the computational proceeds in rounds t, during which nodes exchange *at least one* message with each of their neighbors. Based on whether rounds of neighboring nodes overlap or not, we distinguish two versions: the asynchronous and the synchronous model. The main assumption posed, i. e., that at least one message is exchanged with each neighbor, can be implemented in either of two ways: deterministically by using a local schedule and probabilistically by random beaconing. We quantify the computation cost in terms of the algorithm's *time complexity* (total number of rounds) and *message-size complexity* (number of bits transmitted per message).

Synchronous algorithms. From Theorem 5.1, we can immediately see that, in the synchronous model, $\mathbf{K}_s$ is computable in constant time and message size by the well known recursion:

$$\underbrace{y^{(t+1)}(u) \leftarrow \sum_{v \sim u} [S]_{uv} y^{(t)}(v)}_{\text{repeat every round}} \quad \text{and} \quad \underbrace{y^{(0)}(u) \leftarrow x(u)}_{\text{initialization}} \tag{5.2}$$

The output $y^{(t)}(u)$ of node u after s rounds is exactly $(\mathbf{K}_s x)(u)$. Furthermore, each exchanged message contains exactly one scalar $(y^{(t)}(v))$. Since the computation does not depend on any measure of network size, we conclude that $\mathbf{K}_s$ is locally computable in the synchronous model.

Asynchronous algorithms. How efficiently can scale-space kernels $\mathbf{K}_s$ be computed by a network without round-level synchronization? To answer this question, we examine the (only) three algorithms known to compute graph kernels asynchronously. We then show that these known algorithms cannot compute scale-space kernels with simultaneous $O(s)$ time complexity and $O(1)$ message-size (as in the synchronous model).

The *first algorithm* is straightforward: one can 'simulate' the synchronous recursion (5.2) in an asynchronous network by algorithmically enforcing synchronous initialization, execution, and termination. Synchronization however incurs a recurring overhead of Ω(diameter) rounds [17]. The first algorithm therefore exhibits increased time-complexity and is not local. The *second algorithm* trade's off time for message-size complexity: according to Theorem 5.1, $\mathbf{K}_s$ is s-local. This means that each node u needs only the values and connectivity of all of its neighbors in its s-hop vicinity to compute $(\mathbf{K}_s x)(u)$. Communicating once with each neighbor in the s-hop vicinity is possible in s rounds. However, the message size can grow exponentially with the number of hops, i. e., $\Omega(\delta^s)$, where δ is the minimum node degree. Its increased message-size complexity renders the second algorithm impractical. The *third known algorithm* takes a different approach: instead of computing a graph kernel using power iteration (as in (5.2)), this type of algorithms uses an alternative recursion that -converges- to the output in $O(s)$ without being affected by asynchrony-induced perturbations. Though, as is shown in Chapter 3, this method holds great promise for graph kernels in general, Theorem 5.2 shows that it cannot be applied in our case.

Theorem 5.2. *No 1-st order recursion converges to* $\mathbf{K}_s$ *in the asynchronous model.*

Proof. In the most general sense, 1-st order recursions are given by

$$y^{(t+1)} = A^{(t+1)} y^{(t)} + B^{(t+1)} x \quad \text{and} \quad y^{(0)} = x, \tag{5.3}$$

where, because $\{A^{(t)}\}$ and $\{B^{(t)}\}$ are sequences of 1-local matrices, at each round t nodes only directly communicate with their neighbors.

Though inherently synchronous, under sufficient stability conditions such recursions also converge in the asynchronous model. By standard arguments, (5.3) is stable if, for all $A^{(t)}$, the largest absolute eigenvalue $|\lambda_{\max}(A^{(t)})| < 1$. In the special case that $B^{(t)} = 0_{n\times n}$, stability is also obtained as long as $|\lambda_{\max}(A^{(t)})| \leq 1$. To ensure feasibility in the asynchronous model we have to additionally impose that no τ exists for which $A^{(t)} = 0_{n\times n}$ for all $t > \tau$. Indeed, if such τ exists system (5.3) degenerates to recursion 5.2, which is synchronous.

The steady state is of the system is

$$\mathbf{K}' x = \begin{cases} \lim\limits_{t\to\infty} \Phi(1, t+1)x & \text{if } B^{(t)} = 0_{n\times n} \\ \lim\limits_{t\to\infty} \sum\limits_{k=1}^{t+1} \Phi(k+1, t+1) B^{(k)} x & \text{otherwise,} \end{cases} \tag{5.4}$$

where $\Phi(t_1, t_2) = A^{(t_2)} A^{(t_2-1)} \dots A^{(t_1)}$.

To deduce that $\mathbf{K}' \neq \mathbf{K}_s$, we will consider the amount of signal information each kernel depends on. We will show that, whereas $\mathbf{K}_s$ only considers the signal in an s-hop vicinity around each node, $\mathbf{K}'$ depends on the entire signal. The two matrices therefore cannot be equal.

From Theorem 5.1, we know that $\mathbf{K}_s$ is exactly s-local. But do matrix sequences $\{A^{(t)}\}$ and $\{B^{(t)}\}$ exist such that $\mathbf{K}'$ is s-local? Being a product of $(t_2 - t_1 + 1)$ 1-local matrices, $\Phi(t_1, t_2)$ is $(t_2 - t_1 + 1)$-local. Furthermore, it is non-zero: if $A^{(\tau)} \neq 0$ for every $\tau \in [k+1, t+1]$,

$$\|\Phi(k+1, t+1)\| \geq \prod_{\tau=k+1}^{t+1} \sigma_{min}(A^{(\tau)}) > 0,$$

5

where σ_{min} is the minimum singular value [83]. Terms $\Phi(k+1, t+1)B^{(k)}x$ thus violate the locality requirement whenever $t \geq s + k - 2$. We conclude that $\mathbf{K}'$ can only be s-local if $B^{(t)} \neq 0_{n \times n}$ and the coefficients of successive terms cancel out. This is however impossible as, from Theorem 5.1, all coefficients are positive. ■

The intuition of the proof is that, whereas $(\mathbf{K}_s x)(u)$ is truncated, i. e., it depends on -at most- the values in an s-hop neighborhood of u, any kernel computed by the third algorithm decays asymptotically with the number of hops. Though more tedious, the same argument applies for showing that no recursion of *any* order converges to $\mathbf{K}_s$ in the asynchronous model. Notice that, our results do not suffice to prove that $\mathbf{K}_s$ are not locally computable in the asynchronous model with bounded message size, but rather that no currently known such algorithm exists.

5.3.3. Connection to Graph Filters

We give an alternative interpretation to graph scale-space theory by noticing that each of the three candidate kernels forms a family of low-pass graph filters [110]. This brings forth two main insights: *First, scale acts as the parameter of a low-pass graph filter.* Therefore, the way that a scale-space kernel attenuates peaks and pits is affected by the spectral graph properties. *Second, scale-space theory reveals the design characteristics of a -signal-agnostic- low-pass graph filter*, i. e., the only appropriate filter when no information about the signal and graph is known.

Showing that $\mathbf{H}_s$ is a graph filter is straightforward—any power series of a generalized Laplacian matrix is a valid graph filter. The same also holds for kernels $\mathbf{T}_s$

and $\mathbf{P}_s$ (up to a normalization factor). To see this notice that,

$$\begin{aligned}\mathbf{P}_s = P^s &= \left(D^{-1/2}(I-\mathcal{L})D^{1/2}\right)^s \\ &= D^{-1/2}(I-\mathcal{L})^s D^{1/2}\end{aligned} \tag{5.5}$$

and equivalently,

$$\mathbf{T}_s = T^s = D^{1/2}(I-\mathcal{L})^s D^{-1/2}. \tag{5.6}$$

Both $\mathbf{T}_s$ and $\mathbf{P}_s$ are therefore power series of the normalized Laplacian $\mathcal{L}$, normalized by (diagonal) matrices $D^{\pm 1/2}$. Furthermore, the spectral response of all three candidate kernels is given by $r(\lambda_k; s) = (1-\lambda_k)^s$, where λ_k is the k-th eigenvalue of $\mathcal{L}$.

Insights. Our first insight concerns the interpretation of scale s. Because all three filters attenuate phenomena of high variation (their spectral response is a monotonically decreasing function in terms of λ_k) they are low-pass. From this perspective, the scale of observation is a parameter that determines the extent to which phenomena of high variation (corresponding to high λ_k) are filtered. The larger the scale is, the simpler the graph signal becomes. Therefore, the way that a kernel alters a signal is inherently tied to the spectral graph properties. Roughly, peaks lying on sparse subgraphs, and containing many bottleneck links, are attenuated first. A more exact characterization of the influence of the graph spectrum can be found in the analysis of graph filters—see Chapters 2-to-4.

Our analysis also has a second consequence. It answers an open question on signal processing on graphs: *what is the spectral response of a signal-agnostic low-pass graph filter?* Since scale-space matrices S form a superset of generalized Laplacians, any filter which abides to the scale-space axioms (i. e., it does not make any assumptions about the signal) must have a spectral response of $r(\lambda_k; s) = (1-\lambda_k)^s$, for λ_k the eigenvalue of some generalized Laplacian. It is therefore possible to design signal-agnostic filters using any arbitrary Laplacian (i. e., any notion of graph variation).

5.4. Tracking Extrema Across Scale

In this section we identify and track the extrema of graph signals across scale. The computed trajectory –called the signal's *deep structure*– is a tree-like abstraction that reveals information about the relation and relative importance of peaks and pits.

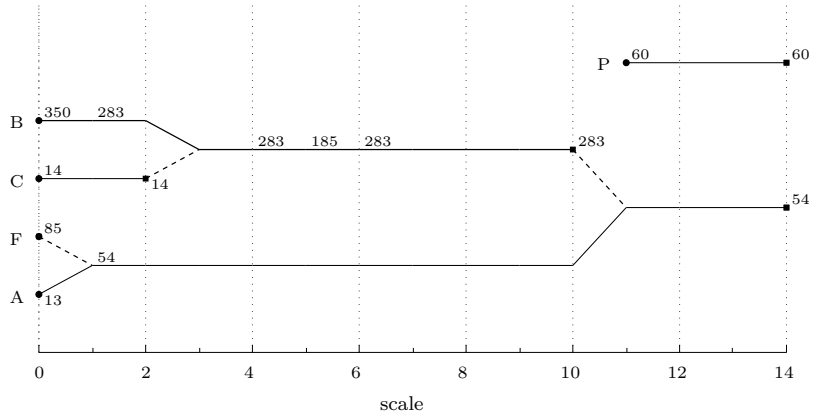


Figure 5.3: The deep structure (of Figure 5.1) describes the trajectory of extremal points (horizontal black lines) across scale. Dotted lines correspond to collapses. For each extremum we give the ids of the nodes in its trajectory.

5.4.1. Deep Structure

As a signal is diffused, its extrema evolve. But what does this reveal about the signal's peaks and pits? We distinguish two cases:

Drift. When a peak/pit changes shape, the corresponding extremum drifts. In Figures 5.1(a) and 5.1(b) for example, as peak B is flattened, the maximum drifts from node 350 to 283. In this case, we say that extremum 350 is *unstable*. The movement of an unstable extremum e is captured in its trajectory $(u_b, \ldots, u_s, \ldots u_d)$, where u_s is the node e resides on at scale s, b is the scale over which e is born and d over which it dies. Notice that, even though it is possible that $u_s \neq u_{s+1}$, both correspond to a single signal structure (i. e., peak or pit).

Collapse. When a peak/pit collapses, the corresponding extremum dies. A collapse thus entails the destruction of a signal structure. Consider for example peak C in Figure 5.1(b). By $s = 7$, the peak has collapsed and extremum 14 dies. It is also easy to see that, after peak C collapses, it is drawn into peak B: a gradient ascent on y_7 starting on node 14 ends up on node 283.

Diffusion results in a tree-hierarchy of collapses, each corresponding to a structure merging with a larger nearby structure. This process is compactly captured by the signal's deep structure. As shown in Figure 5.3, the deep structure visualizes the trajectory $(u_b, \ldots, u_d)$ of each extremum e using a black horizontal line. The

trajectory spans the scales over which e lives: it starts at scale b and ends at d. Collapses on the other hand are depicted using dotted lines. A collapse line connects the rightmost endpoint of e's trajectory with the extremum it merges into.

5.4.2. Tracking Algorithm

We next present an algorithm for the computation of the signal's deep structure. In our approach, the knowledge of the deep structure is distributed amongst the signal extrema. For this reason, our algorithm's time complexity $O(d_x)$ does not depend on the network diameter d (no node knows the complete deep structure), but on the diameter d_x of the largest peak/pit in the signal's scale-space[4]. If a global view is necessary, the data can be disseminated using traditional techniques [49].

Due to drift or collapse, the extrema of a signal evolve. We track their trajectory by forwarding information greedily along the diffused signal's gradient. The algorithms are better understood from an extremum's perspective: Consider an extremum e which at scale s resides on node u_s. W.l.o.g., assume that e is a maximum[5]. If u_s is also a maximum at scale $s+1$, then no change occurred and $u_{s+1} = u_s$. On the other hand, if drift or collapse occurs, a neighbor v will have a larger value than u_s. While v may not be necessary a maximum, the maximum u_{s+1} is easily found by following the gradient ascent path, i. e., by iteratively forwarding a tracking query to the neighbor with the largest value.

Corollary 5.1. *If a node u_s has a locally maximal value at scale s, but not at scale $s+1$, the gradient ascent path from u_s to the new local maximum u_{s+1} can have arbitrary length.*

Proof. We construct an example in which u_s and u_{s+1} are k hops away. Because of the semi-group axiom, it is sufficient to give an example for $s = 0$, $y_s = x$ and $y_{s+1} = y_1$. Let $\mathcal{G}$ be a line graph and x a signal with strictly monotonically increasing values $y_s(u_{i+1}) = y_s(u_i) + B$ for all $u_i \in \mathcal{V}$. Create a maximum at u_i by setting $y_s(u_i) = y_s(u_{i+1}) + \beta$ for some small positive β. We will show that for any scale space kernel, $y_1(u_{i-1}) < y_1(u_i) < y_1(u_{i+1})$ if $B > \beta/3$. The maximum will therefore move until it reaches node u_n ($k = n - i$ hops) at the end of the line graph. The value of a node u_i at scale 1 is $\alpha_l x(u_{i-1}) + \alpha_c x(u_i) + \alpha_r x(u_{i+1})$, where α are kernel dependent positive coefficients for the left (α_l), center (α_c), and right (α_r) neighbors.

[4]Though usually $d_x \ll d$, it is possible to construct an example in which $d_x = d$.

[5]For ease of presentation, we focus our discussion on maxima. Our results however also hold for minima. Tracking minima is equivalent to tracking the maxima of $-x$.

Furthermore, $\alpha_l = \alpha_r$ as the nodes in a line-graph have no way of distinguishing between their left and right neighbors. Inequality $y_1(u_{i-1}) < y_1(u_i)$ holds if: (i) $\alpha_l x(u_{i-2}) < \alpha_r x(u_{i+1})$ and (ii) $(\alpha_c - \alpha_l)x(u_{i-1}) < (\alpha_c - \alpha_r)x(u_{i-2})$. Due to our construction, both inequalities are satisfied. Following the same reasoning, inequality $y_1(u_i) < y_1(u_{i+1})$ holds if $3B = x(u_{i+2}) - x(u_{i-1}) > x(u_i) - x(u_{i+1}) = \beta$. ■

Even though the tracking algorithm is the same whether the extremum drifts or collapses, it is easy to distinguish the two cases locally: in a collapse, two distinct extrema merge into one. Thus whenever a maximum receives more than one gradient ascent (tracking) query, a collapse has occurred.

But if two extrema merge, which of the two collapsed? Suppose that

$$e_u = (\ldots, u_s, w_{s+1}, \ldots, w_d) \text{ and } e_v = (\ldots, v_s, w_{s+1}, \ldots, w_d)$$

are the trajectories of two maxima, which at scale $s+1$ merge at node w_{s+1}. To figure out which maximum collapsed we use the following procedure: From Corollary 5.2, we know that the absolute value of an *unstable* extremum always decreases. Thus, if the value of maximum e_u increased between scales s and $s+1$, and the value of e_v decreased i. e., if

$$y_s(v_s) \geq y_{s+1}(w_{s+1}) > y_s(u_s), \tag{5.7}$$

then e_u collapsed and e_v was unstable. But what if the value of both e_u and e_v decreased from scale s to $s+1$, i. e.,

$$y_s(u_s) > y_{s+1}(w_{s+1}) \text{ and } y_s(v_s) > y_{s+1}(w_{s+1})? \tag{5.8}$$

This phenomenon can occur in noisy signals, when a peak is 'hidden' under nearby extrema (caused by noise): as the signal is diffused and the extrema merge, their values jointly decrease. In this case, there is no reliable way to distinguish which extremum collapsed[6]. Thus the scale-space method can discern that the merged extrema jointly form a peak, but can not trace back the extremum to the most relevant node at scale zero. In Figure 5.3, this case would be depicted as two dotted lines merging into a new extremum.

Corollary 5.2. *The absolute value of an extremum always decreases, i. e.,* $|y_s(u_s)| \geq |y_{s+1}(u_{s+1})|$, *where* u_s *is the node the extremum resides on at scale* s.

[6] In fact, one could argue that all extrema collapsed.

Proof. Let $u_s \neq u_{s+1}$ the nodes over which an unstable extremum e resides on at scales s and $s+1$, respectively. W.l.o.g., the extremum is a maximum and the claim is that $y_s(u_s) \geq y_{s+1}(u_{s+1})$. By Theorem 5.1, the value of a node in scale $s+1$ is bounded by the maximum value at most 1-hop away from it in scale s (this includes the node itself). Thus, the only way that the inequality does not hold is that there exists node v at most 1-hop away from u_{s+1} with $y_s(v) > y_s(u_s)$. This implies that a second extremum e' exists near u_{s+1} which is reachable by gradient ascent. Moreover, e collapses to e' at $s+1$, which is a contradiction. ■

5.5. Extremum Selection

We now describe how we use the signal's deep structure to identify peaks and pits. We consider two criteria for extremum selection: The *lifetime criterion* aims at filtering out false extrema, while the *birth criterion* aims at discarding phantom extrema.

5.5.1. The Lifetime Criterion

One of the fundamental results of scale-space theory is that the lifetime of extrema is a measure of the importance of the corresponding peaks and pits [73]. Though a number of definitions exist, the most natural defines the lifetime l_e of an extremum e as the length of scale-period over which e exists: $l_e = d_e - b_e$. Here, b_e and d_e denote the scale over which e is born and dies, respectively. The lifetime criterion is useful in separating small structures (false extrema) –caused by small signal perturbations– from inherent signal trends (true extrema): an extremum is true if $l_e \geq l$, where l is a user-defined lifetime threshold, and false otherwise. As we will see in the following, by setting l we can (roughly) retain peaks/pits that are bigger than l hops.

According to classic scale-space theory, the lifetime criterion is successful because it captures two key properties of the signal's peaks and pits: *volume* and *span*. Volume is a property that mainly depends on the signal, while span is also affected by the graph topology. Important structures in a signal have large volume. In Figure 5.1(a) for example, peak A is the largest because it is tall and decreases slowly at each direction. The volume in an s-hop neighborhood around e is captured by $y_s(e) = (\mathbf{K}_s x)(e)$ and is a weighted average of all the information residing at most s hops away from e. Additionally, a peak is important if it spans multiple hops. In Figure 5.1(a), peak A is more significant than B because A's width covers more

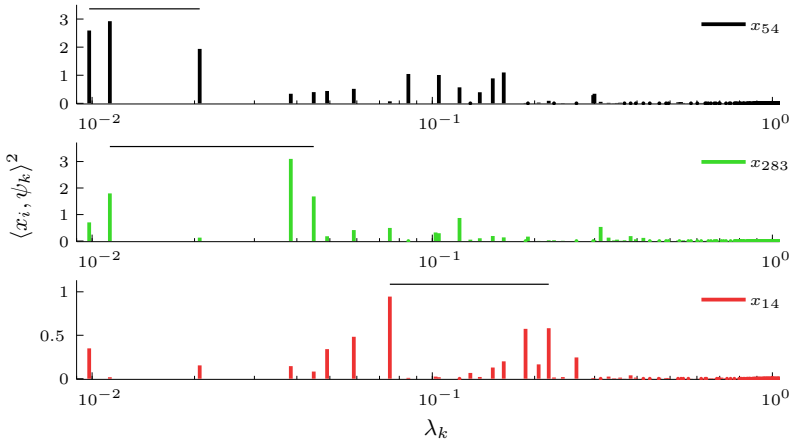


Figure 5.4: The larger a peak (pit) is, the more its curvature distribution is concentrated around small-order projections. Here, we show the distribution of each of the three peaks illustrated in Figure 5.1(a). The horizontal bar spans the top three curvatures of each peak.

hops. Intuitively, lifetime gives the size of the largest s-hop neighborhood centered around e.

The influence of network topology. We next investigate the relation of extremum lifetime to the graph spectrum. Our results suggest that lifetime assigns higher importance to structures residing on dense subgraphs.

As we saw in Section 2.2, any signal x can be written as $x = \sum_{k=1}^{n} \hat{x}_k \phi_k(S)$, where $\hat{x}_k = \langle x, \phi_k(S) \rangle$ is the k-th spectral coefficient of x. The major benefit of the spectral decomposition is that it induces a notion of size. In a nutshell, slow changing signals –intuitively signals containing large structures– decompose into small order terms and their spectral coefficients are biased towards smaller k.

We use the same method to examine structures individually. To clarify this consider the example shown in Figure 5.1(b). If we split x to sub-signals $x = x_{54} + x_{283} + x_{14}$, corresponding to the peaks around extrema 54, 283, and 14, respectively, we have that

$$x = \sum_{k=1}^{n} \langle x_{54}, \phi_k(S) \rangle \phi_k(S) + \sum_{k=1}^{n} \langle x_{283}, \phi_k(S) \rangle \phi_k(S) + \sum_{k=1}^{n} \langle x_{14}, \phi_k(S) \rangle \phi_k(S). \quad (5.9)$$

The size of structure i is therefore captured by the distribution of spectral coefficients $\{\langle x_i, \phi_k(S) \rangle^2, \text{ for } k \in [1, n]\}$. Figure 5.4 summarizes all three distributions by plotting a bar for each $\langle x_i, \vartheta_k \rangle^2$ at μ_k, i. e., for the consensus scale-space kernel. The horizontal line spans the top three coefficients of each structure. As shown in the

figure, the distribution of larger structures (top) is concentrated around projections to eigenvectors with lower order k and vice versa. Comparing to the signal's deep structure (shown in Figure 5.3), we also observe that longer a structure lives, the larger its coefficients are.

But how does lifetime relate to spectral decomposition? Even though the exact relation remains elusive, we conjecture that lifetime is inversely proportional to the eigenvalue gap between the distributions of different structures. This suggests that lifetime is a relative measure; it captures the importance of a structure relative to nearby structures. Furthermore, it suggests that lifetime assigns higher importance to structures over dense subgraphs (such structures tend to be decomposed into lower order eigenvectors). In the following we give evidence that the conjecture holds by deriving a closed-form expression for the simpler case in which the signal contains two structures with singleton distributions (each structure decomposes into exactly one eigenvector).

Proposition 5.2. *Let λ, ϕ be the eigenvalues and eigenvectors of a scale-space matrix S. Consider a signal $x = \alpha_i \phi_i + \alpha_j \phi_j$, with $i < j$ and constants $\alpha_i > \alpha_j > 0$ chosen such that at least one maximum of each eigenvector exists in x and w.l.o.g., let u_i, u_j denote two non-degenerate maxima of ϕ_i, ϕ_j. Their lifetime –respectively l_i and l_j– when filtered with a scale-space kernel $\mathbf{K}_s^* = e^{Ss}$ is*

$$l_i = \infty \quad \text{and} \quad l_j = \frac{\log\left(\frac{\lambda_j \alpha_j \phi_j(u_j)}{\lambda_i \alpha_i \phi_i(u_i)}\right)}{\lambda_j - \lambda_i}. \tag{5.10}$$

This proposition essentially computes the lifetime of the signal's structures, i. e., the peaks and pits of the two eigenvectors. Notice that, while eigenvectors have multiple extrema, from a spectral point-of-view, all extrema are equivalent. Maxima u_i and u_j are thus good representatives of the signal structures. Upon closer inspection we also see that the proposition uses kernel $\mathbf{K}_s^*$ instead of $\mathbf{K}_s = S^s$ (see Section 5.3). $\mathbf{K}_s^*$ is simply the continuous counterpart of $\mathbf{K}_s$ and allows us to identify lifetime with higher precision.

Proof. Since maxima u_i and u_j both exist in x, to compute their lifetime l_i and l_j it is sufficient to find the scales d_i and d_j over which the maxima die. For this we will use the non-enhancement axiom. Note that, since the axiom gives sufficient but not necessary conditions, this result only holds for non-degenerate extrema (their

value increases after dying and no drift occurs). The scale derivative of y_s is

$$\begin{aligned}\frac{\partial y_s}{\partial s} &= \frac{\partial \mathbf{K}_s^*(\alpha_i\phi_i + \alpha_j\phi_j)}{\partial s} \\ &= \sum_{k=1}^{n} \frac{\partial e^{-\lambda_k s}}{\partial s}\langle \alpha_i\phi_i + \alpha_j\phi_j, \phi_k\rangle\phi_k \\ &= -\alpha_i\lambda_i e^{-\lambda_i s}\phi_i - \alpha_j\lambda_j e^{-\lambda_j s}\phi_j,\end{aligned}$$

where in the last step we exploited that eigenvectors are bi-orthonormal. Function $y_s(u)$ has an inflection point at

$$\frac{\partial y_s(u)}{\partial s} = 0 \Rightarrow s^* = \frac{\log\left(\frac{\alpha_j\lambda_j\phi_j(u)}{\alpha_i\lambda_i\phi_i(u)}\right) + 2\mathrm{Im}(1)\pi k}{\lambda_j - \lambda_i}.$$

Here, Im(1) is an imaginary number of unit length and $k \in \mathbb{Z}$. For $k = 0$, s^* is a positive real number, as $\alpha_j > \alpha_i$, $\lambda_j > \lambda_i$, and $\phi_j(u_j) > \phi_i(u_i)$. Since $y_{s^*}(u)$ is convex, s^* is a minimum and corresponds to the collapse of u_j, i. e., $l_j = s^*$. Because no other minimum exists for $s \in \mathbb{R}^+$, $l_i = d_i = \infty$. ■

Proposition 5.2 is the first step towards understanding the relationship between the extremum lifetime and the graph spectrum. Our main insight is that, the eigenvalue gap $\lambda_j - \lambda_i$ is the decisive factor in determining lifetime. From spectral graph theory we know that eigenvalue λ_i quantifies how easy is it to partition a graph in (at most) i sub-graphs. For instance, the smaller λ_2 is, the harder it is to partition the graph in two. As such, *structures with large lifetime are cohesive, well connected, do not contain bottleneck links, and span a significant portion of the network*. The height of peaks on the other hand –given by amplitude ratio $\alpha_j\phi_j(u_j)/\alpha_i\phi_i(u_i)$– is of secondary importance.

5.5.2. The Birth Criterion

As depicted in Figure 5.2(b), a scale-space kernel introduces distortion when the underlying graph is unevenly connected. We call any effects that are present on the filtered signal, but not on the underlying physical and sensed signal, *phantom*. Phantom effects are particularly harmful as they obscure real data (cf. Section 5.6). Nevertheless, in the particular case of phantom extrema, there is an easy way to recognize them from the signal's deep structure. As we show in the following, phantom effects occur only in large scales. As a consequence, a phantom extremum e is always born at large scales and $b_e > 0$. This is the birth criterion.

We examine phantom effects analytically by comparing filtering of the same signal in two different topological spaces: $\mathcal{G}$ models the network topology (e. g., an irregular and sparse graph), whereas $\overline{\mathcal{G}}$ models the space over which the signal is defined (e. g., an ideal unbiased lattice). To demonstrate the dependency between phantom effects, topology and scale, we examine how the absolute quadratic error

$$e_s = \left| \frac{x^\top (\overline{\mathbf{K}}_s - \mathbf{K}_s) x}{x^\top x} \right| \tag{5.11}$$

of the corresponding scale-space kernels $\overline{\mathbf{K}}_s = \overline{S}^s$ and $\mathbf{K}_s = S^s$ changes as the signal is filtered. We are interested in how the error behaves when scale s increases. Using simple operations we can see that $0 = e_0 \leq e_\infty$, which indicates that the error has an increasing trend. The following result bounds the error between the extremes of $s = 0$ and $s \to \infty$, when S is symmetric (all candidate kernels have symmetric scale matrices).

Proposition 5.3. *For any two graphs $\overline{\mathcal{G}} = (\mathcal{V}, \overline{\mathcal{E}})$ and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, signal $x \in \mathbb{R}^n$, and scale $s \in \mathbb{N}$,*

$$\max(0,\, e_\infty - 2\lambda_2^s) \leq e_s \leq 1 - \lambda_{min}^s + \epsilon_s$$

where $\epsilon_s = (2s^s)/(s+1)^{s+1}$ when s is odd and zero otherwise, $-1 < \lambda_{min} < 0$ is the smallest amongst the eigenvalues of $\overline{S}$ and S, and $0 < \lambda_2 < 1$ is the smallest of the second eigenvalues of $\overline{S}$ and S.

Proof. We split the proof in two parts: the upper bound is given in Lemma 5.3 and the lower in Lemma 5.4. ■

Obviously, both the lower and upper bounds increase with s. Moreover, for all $s \leq \log_{\lambda_{min}}(1 - e_\infty)$, $e_s < e_\infty$. This suggests that phantom effects become more likely as scale increases. We have to note that, it is possible, though unlikely, that an extremum e born at a scale larger than zero is not phantom. As discussed in Section 5.3, we can construct a signal that contains hidden peaks; i. e., peaks revealed by filtering. In that sense, the birth criterion is necessary but not sufficient.

Lemma 5.3. *For any two graphs $\overline{\mathcal{G}} = (\mathcal{V}, \overline{\mathcal{E}})$ and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, signal $x \in \mathbb{R}^n$, and scale $s \in \mathbb{N}$, $e_s \leq 1 - \lambda_{min}^s + \epsilon_s$, where $\epsilon_s = (2s^s)/(s+1)^{s+1}$ when s is odd and zero otherwise. Furthermore, $-1 < \lambda_{min} < 1$ is the smallest amongst the eigenvalues of $\overline{S}$ and S.*

Proof. Denote by $\lambda_{\max}(S)$ and $\lambda_{\min}(S)$ the largest and smallest eigenvalue of a matrix S. It is well known that for any symmetric matrix S, $\lambda_{\min}(S)x^\top x \leq x^\top Sx \leq \lambda_{\max}(S)x^\top x$ [16]. Furthermore, as a consequence of Theorem 5.1, $\lambda_{\max}(S) = 1$ and $-1 < \lambda_{\min}(S) < 1$. Combining the inequalities for matrices $\overline{S}^s$ and S^s we have that

$$\lambda^s_{\min}(\overline{S}) - 1 \leq \frac{x^\top(\overline{S}^s - S^s)x}{x^\top x} \leq 1 - \lambda^s_{\min}(S).$$

Setting $\lambda_{min} = \min(\lambda_{\min}(S), \lambda_{\min}(\overline{S}))$, we get a first upper bound $e_s \leq 1 - \lambda^s_{min}$. When s is odd, $1 - \lambda^s_{min} > 1$, which is loose. To improve upon this, we will use the triangle inequality:

$$\begin{aligned} e_{s+1} &= \left| \frac{x^\top\left(S^{s+1} - S^s + \overline{S}^s - \overline{S}^{s+1} + S^s - \overline{S}^s\right)x}{x^\top x} \right| \\ &\leq \left| \frac{x^\top(S^{s+1} - S^s)x}{x^\top x} \right| + \left| \frac{x^\top(\overline{S}^{s+1} - \overline{S}^s)x}{x^\top x} \right| + e_s \end{aligned} \tag{5.12}$$

Term $x^\top(S^{s+1} - S^s)x$ captures the distance traveled in one step by each of the two dynamical systems and is

$$\begin{aligned} \left| \frac{x^\top(S^{s+1} - S^s)x}{x^\top x} \right| &= \left| \frac{\sum_{k=1}^n \left(\lambda_k^{s+1} - \lambda_k^s\right)(\psi_k^\top x)^2}{x^\top x} \right| \\ &\leq \frac{\left| \sum_{k=1}^n \left(\left(\frac{s}{s+1}\right)^{s+1} - \left(\frac{s}{s+1}\right)^s \right)(\psi_k^\top x)^2 \right|}{x^\top x} \\ &= \frac{s^s}{(s+1)^{s+1}}. \end{aligned} \tag{5.13}$$

Above, ψ_k is the k-th eigenvector of S and, in the second step, we bounded the convex function $\lambda_k^{s+1} - \lambda_k^s$ using its minimum value. Substituting (5.13) into (5.12), we obtain the desired bound. ■

Lemma 5.4. *For any two graphs $\mathcal{G}$ and $\overline{\mathcal{G}}$, signal $x \in \mathbb{R}^n$, and scale $s \in \mathbb{N}$, $e_{s+1} \geq \max(0, e_\infty - 2\lambda_2^s)$ where $0 < \lambda_2 = \min(|\lambda_2(S)|, |\lambda_2(\overline{S})|) < 1$ is the smallest of the second eigenvalues of matrices S and $\overline{S}$.*

Proof. We will express $S^s - \overline{S}^s$ in terms of the e_∞. Because scale-space kernels are marginally stable, e_∞ is bounded. Furthermore, it is easily computable when

scale-space matrices S and $\overline{S}$ are known.

$$e_s = \left| \frac{x^\top \left(S^\infty - \overline{S}^\infty + \overline{S}^\infty - \overline{S}^s - S^\infty + S^s \right) x}{x^\top x} \right|$$
$$\geq e_\infty - \left| \frac{x^\top (\overline{S}^\infty - \overline{S}^s) x}{x^\top x} \right| - \left| \frac{x^\top (S^\infty - S^s) x}{x^\top x} \right| \tag{5.14}$$

Note that above we abuse notation and refer to $\lim_{s\to\infty} S^s$ as S^∞. For both the second and third terms in (5.14) the following simple upper bound holds:

$$\left| \frac{x^\top (S^\infty - S^s) x}{x^\top x} \right| = \left| \frac{\sum_{s=2}^n \lambda_i^s(S)(\psi_i^\top x)^2}{x^\top x} \right|$$
$$\leq |\lambda_2^s(S)| \tag{5.15}$$

Substituting (5.15) into (5.14), we obtain the lower bound

$$e_s \geq e_\infty - \lambda_2(S)^s - \lambda_2(\overline{S})^s \geq e_\infty - 2\lambda_2^s \tag{5.16}$$

which holds as long as $s \geq \log_{\lambda_2}(e_\infty)$. Otherwise, we use the trivial lower bound $e_s \geq 0$. ■

5.6. Evaluation

We split our evaluation in two parts. Section 5.6.1 quantifies the identification accuracy of our method and compares it with the state-of-the-art via simulations. Section 5.6.2 evaluates a proof-of-concept implementation in a large-scale wireless testbed.

5.6.1. Simulations Results

We conducted Matlab simulations using a test set consisting of ten synthetic physical signals. Each signal was a mixture of seven multi-variate Gaussian distributions with random mean and covariance. The signal was sampled by 1024 nodes deployed at random. The objective of the simulations was to quantify the efficiency of identifying the seven Gaussian peaks. We measured identification accuracy using the standard metrics of *precision* and *recall*. Precision measures the fraction of true peaks present among all retrieved peaks, and recall the fraction of true peaks with respect to the ground truth. The ideal method therefore retrieves all true peaks (recall = 1) and discards peaks caused by false/phantom extrema (precision = 1).

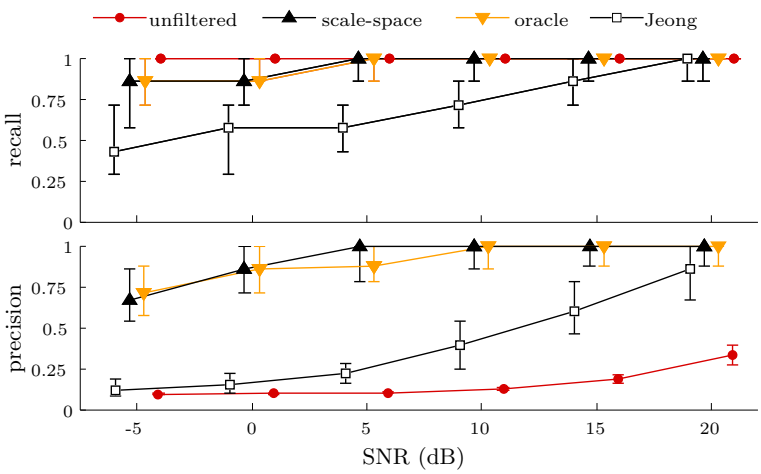


Figure 5.5: Precision/recall median and standard deviation for different methods and SNRs. Both oracle and scale-space filters achieved significantly better recall than state-of-the-art. Scale-space did so with a fixed lifetime thresholds ($l = 7$). To diminish overlap, we perturbed the horizontal positioning of data points.

To evaluate our algorithms, we compared the scale-space approach (*scale-space*) to the state-of-art method for identifying peaks and pits (*Jeong*) [58]. Additionally, we compared our method to the 'raw' unfiltered signal (*unfiltered*) and to the single-scale graph filter with the best possible performance (*oracle*). The latter was computed offline by exhaustively searching, for each experiment, the scaled signal y_s that maximized the minimum value of the metrics: $\text{maxmin}_s\{\textit{precision}(y_s), \textit{recall}(y_s)\}$. While it is infeasible to compute the oracle filter online, it serves as a benchmark for judging the benefit of the scale-space approach (multi-scale) over the best possible graph filter (vs. best single scale).

Because graph filters (and the scale-space approach) identify the peaks and pits of a signal by varying the scale of observation, an extremum of a scaled signal is not necessarily located at the same node as in the physical signal. To verify that an extremum in a large scale corresponded to a peak, we checked whether the node that the extremum resided on at the sensed signal was close to a true extremum. When extrema merged without collapsing, we checked whether one of them was close to a true extremum. Otherwise, the identification was deemed a false positive. Analogously, peaks/pits missed were deemed false negatives.

The effect of noise. To evaluate the influence of noise, we perturbed the sensed signal with random Gaussian noise of zero mean and progressively larger variance.

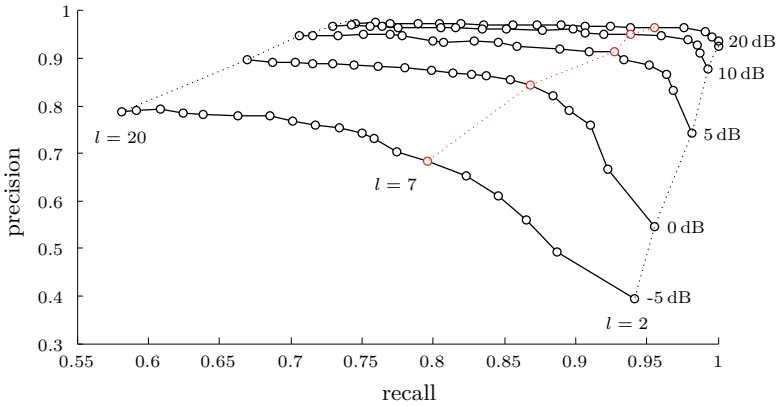


Figure 5.6: Precision/recall trade-off of scale-space for different lifetime thresholds l and SNRs. The points annotated in red correspond to the lifetime threshold used in our evaluation ($l = 7$).

This experiment focused on well-connected topologies (average degree between 13.7 and 14.3) and was devoid of sparsity effects. The average diameter of the graph was 25.4. Figure 5.5 summarizes our results over ten different random topologies, ten signals, and six signal-to-noise ratios (SNR).

We have two main observations: *First, the method by Jeong et al. can be very inaccurate for small SNR.* Even though the method improved over the unfiltered signal, it exhibited a median precision of only 0.12 for SNR $= -5$ dB. *Second, the accuracy of the scale-space approach, in both precision and recall, is on par with the oracle filter.* It should not be a surprise that a hand-tuned graph filter is exceptionally efficient in removing noise. It is however noteworthy that the scale-space approach could achieve low error over so diverse noise-levels with a single parametrization, a lifetime threshold of $l = 7$. Extremum lifetime is little affected by small signal perturbations. In fact, to demonstrate the robustness of our approach to incorrect parametrization, we used the same threshold ($l = 7$) in all simulations and experiments.

But how does the lifetime threshold l affect the identification accuracy in noisy signals? In Figure 5.6 we depict five precision-recall (PR) curves of the same experiment—each curve corresponding to one SNR. The value of l ranges from 2 to 20. The maximum value of 20 is way larger than required to identify any of the seven peaks, but it was used to be exhaustive in our exploration. As usual, a trade-off exists between being too selective (high l) and not selective enough (small l): a threshold that achieves high precision suffers in recall, and vice versa. *Nevertheless, we found that our method improved upon state-of-the-art over all reasonable param-*

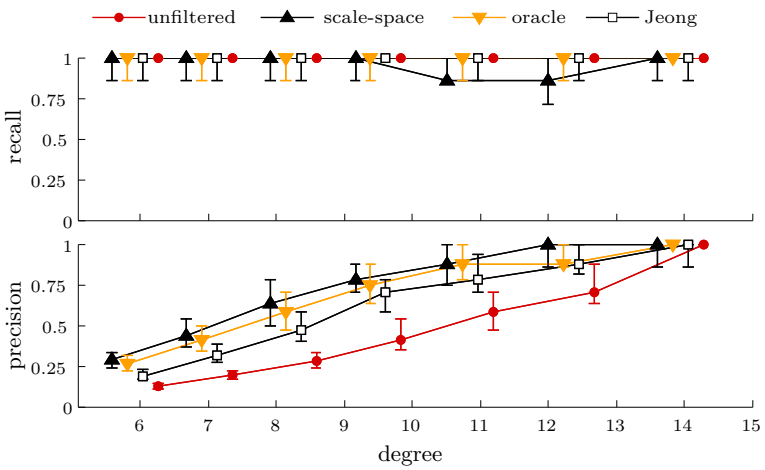


Figure 5.7: Precision/recall median and standard deviation for different methods and average node degrees. To diminish overlap, we perturbed the horizontal positioning of data-points.

eterizations. The scale-space approach outperformed the method by Jeong et al. in precision for all tested l, and in recall for $2 \leq l \leq 10$. Note that, setting $l > 10$ would imply a peak diameter close to 20, which is significantly larger than in our experiments.

The effect of sparsity. As shown by Proposition 5.1, sparsity presents a major challenge by introducing false extrema, i. e., extrema not present in the physical signal. We evaluated the impact of sparsity by progressively decreasing the transmission radius of nodes (no noise was introduced). This resulted in increasingly sparser networks with average degree varying from 5.61 to 14.3. Figure 5.7 summarizes our results over 700 runs (7 radii $\times$ 10 physical signals $\times$ 10 runs).

To begin with, the results confirm that sparsity severely affects precision. We found that the number of false extrema increases exponentially as the average node degree decreases. The exponential trend has a devastating effect on identification accuracy. A case in point is that, for the smallest transmission radius (leftmost errorbars in the figure) the unfiltered signal contained ≈ 50 maxima (average) compared to the seven peaks of the physical signal. Hence, even in the absence of noise, some filtering is necessary. *We can also see that the scale-space approach achieves a slightly higher precision than the state-of-the-art, while not sacrificing recall.* Given the overwhelming number of false extrema (false positives) present in sparse graphs, optimizing for precision is very desirable. To give a sense of perspective, when the

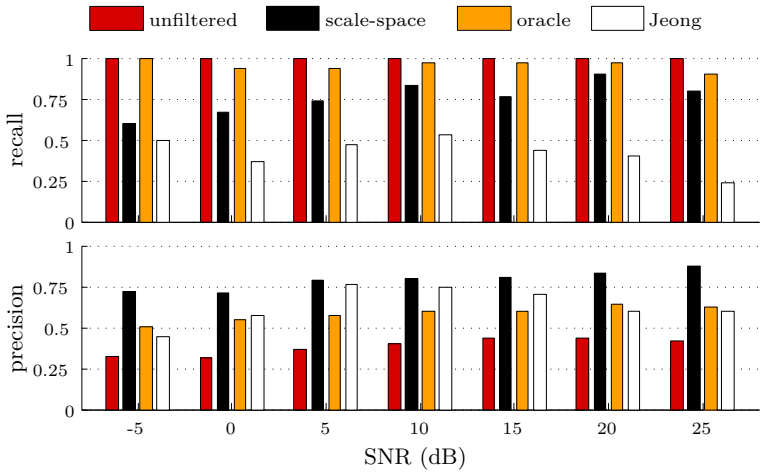


Figure 5.8: Average precision/recall for each noise level in our testbed. The precision of the oracle filter is significantly hampered by the poor correlation of wireless links with the underlying physical space.

average degree was 8.25 the scale-space approach had zero false negatives and four false positives, whereas the method by Jeong et al. had no false negatives but eight false positives (in median).

It is important to remark the lack of accuracy of the oracle filter. If the scale of the oracle filter is hand-picked to maximize its precision/recall, why does it behave so poorly? Furthermore, how does the scale-space approach improve upon it (even slightly)? Sparsity affects a graph filter in two distinct ways. Whereas in small scales a signal is overwhelmed by false extrema, in larger scales filtering causes phantom extrema. This is a challenge that a simple filter cannot overcome: no single scale depicts the extrema of the physical signal clearly. On the contrary, the scale-space approach filters out false extrema at low scales while discarding phantom extrema at higher scales.

5.6.2. Empirical Results

Contrary to common assumption, the connectivity of wireless networks is highly irregular. Especially indoors, wireless links exhibit high spatial and temporal variability, phenomena which significantly affect peak and pit identification. To evaluate the accuracy of identification methods in real-world wireless networks, we implemented scale-space kernels in Contiki and ran an extensive set of experiments in a 100-node wireless testbed deployed above the ceiling tiles of our office building.

The scale-space computation was built on top of a simple CSMA MAC protocol with round-level synchronization (as discussed in Section 5.3.2, synchrony is a fundamental requirement of scale-space analysis). Experiments were conducted using 10 synthetic signals over 7 noise levels (from -5 dB to 25 dB), resulting in 70 distinct runs. Due to the limited size of our testbed, we inserted in each synthetic signal three well-separated Gaussian distributions with random mean and variance (as compared to seven in our simulations). We used the extracted data (the signals' scale-space $\{y_s\}$ and per-round connectivity) to evaluate and compare the aforementioned methods offline. The number of nodes varied from 73 to 99, resulting in a network diameter between 9 and 10 hops, and an average degree between 7.17 and 9.5 neighbors. Figure 5.8 summarizes our results.

We provide three main observations: *First, the identification precision does not significantly increase when the SNR increases.* For the unfiltered signal, at 25 dB the average precision was only 0.1 higher than at -5 dB. The fact that the precision was low even at high SNR is due to the testbed being not only (relatively) sparse, but also differently connected from the random geometric graphs used in our simulations. We found that a significant number of false extrema were caused by graph irregularities, as depicted in Figure 5.2(a). Nodes spatially adjacent to a true extremum were not always in wireless proximity, thus wrongly identifying themselves as extrema. At low SNR, the number of false positives remained bounded (up to 15), which is not surprising because the number of false extrema (either maxima or minima) is at most equal to the size of the graph's maximal independent set. *Second, the identification recall of Jeong's method was much smaller than in simulations.* We found that the close proximity between peaks, as well as link asymmetry, caused false negatives. We argue that, by testing existing methods in a real-world scenario, we were able to observe behaviors (problems) never observed before. We therefore suggest that future mechanisms should include testbed experiments in order to be evaluated in conditions as diverse as possible. *Third, the scale-space approach was significantly more precise than the oracle filter.* Different from our simulation, the irregular patterns of real wireless links can cause lots of phantom extrema (cf. Section 5.5.2). In fact, phantom extrema can be so severe, that noise effects pale in comparison. In our experiments, phantom extrema were abundant at almost all scales, severely hampering the precision of any (single-scale) graph filter, and thus of the oracle filter. In contrast, the scale-space approach was able to distinguish and discard phantom extrema using the birth criterion. In summary, our testbed experiments demonstrated that, in real-word conditions, it is essential for a method to be resilient to phantom and false extrema.

5.7. Conclusions

According to scale-space theory, to understand the structures of a signal with unknown characteristics, one must observe it at all possible scales—no single filter is sufficient. But can we do so when the signal is defined on an irregular graph? By identifying the properties of scale-space kernels appropriate for graphs (cf. Theorem 5.1)) and studying their computation (cf. Theorem 5.2), this chapter in effect extended scale-space theory to graphs. We demonstrated the usefulness of the scale-space approach by applying it to the problem of peak and pit identification in sensor networks. Beyond peak identification, we believe that principles of scale-space analysis can be beneficial for other problems in sensor networks. For instance, the deep structure could be used to extend current topological methods for signal mapping and compression to multiple scales [102, 133], thus making distributed pattern recognition possible. A second potential application is event-region detection. Tracking event-boundaries across scale has the potential to improve the resilience of current algorithms to phantom effects.

This chapter takes a step towards the distributed scale-invariant analysis of graph signals. However, it also opens up new challenges. The first challenge concerns the asynchronous computation of scale-space kernels. We showed that currently known (practical) asynchronous algorithms cannot compute scale-space kernels in $O(s)$. Thou-gh it seems that synchrony is a fundamental requirement of scale-space kernels for graphs, we lack a formal proof. The second challenge entails characterizing the relation between extremum lifetime and the graph spectrum. We conjectured that lifetime is inversely proportional to the eigenvalue gap of the structures' spectral coefficients. An exact characterization remains elusive. We also found that filtering in graphs can cause phantom effects and we gave bounds on the induced error. Yet the process is little understood. For instance how does this error relate to the properties of the underlying graph?

6

Conclusions

"So long, and thanks for all the fish."

—Douglas Adams,
The Hitchhiker's Guide to the Galaxy

6.1. Summary of Contributions

The ideas, algorithms, and experiments presented in this thesis are inspired by a simple, yet powerful observation: *the graph spectrum is an ideal tool for understanding the properties of graph signals.* In an analogous way to using the Fourier transform for studying time and spatial signals in the frequency domain, the spectral transform enables us to reason about the information on a graph based on the eigenvalues and eigenvectors of the graph's Laplacian. The major benefit of this decomposition is that it induces a notion of size inherently connected to the topological properties of the underlying graph. By projecting a signal onto eigenvectors, we distinguish the size of signal structures on a graph-specific fashion—the larger, more coherent and better connected a structure is, the lower the spectral order of the eigenvectors it decomposes into.

Within this context, this thesis provided the following three main contributions:

Contribution 1. *The design of efficient distributed graph filters, suitable for dynamic wireless networks.* Graph filters are the graph analogues of classical filters—that is, they attenuate signal structures according to their (spectral) size. When one possesses global knowledge, filtering a graph signal is straightforward—though expensive. By directly computing the eigendecomposition of the Laplacian and using Definition 2.3, we can filter a signal in $O(n^3)$ arithmetic operations [98]. In distributed wireless networks however, filtering becomes non trivial. Existing distributed graph filters rely on strong assumptions that render them inefficient. They assume that all nodes are synchronized and that the signal and network remain unchanged for the duration of the computation.

Our first contribution is the removal of these assumptions. In Chapters 3 and 4 we introduced the potential kernel and its variants, a novel family of distributed graph filters suitable for wireless networks. All linear graph filters are dynamical systems; what makes our approach unique is the way the proposed dynamical systems are designed. While existing filters output a transient –and unstable– state of the dynamical system, potential-based filters always converge to the steady –and stable– state. This renders them *robust* to any disturbances, such as those introduced by asynchrony, packet loss, and computation errors. Moreover, because their steady state is determined in a unique way by the input state, our filters *track* changes in the signal and graph. They are therefore able to cope with dynamics.

We showcased the efficiency of potential-based graph filters using analysis, simulations and experiments. The filters were shown to converge linearly –$O(1)$ rounds– and to track various types of dynamics, including the addition and deletion of links and nodes (churn), and the change of information over time. Simulations and experiments validated our analysis and demonstrated that our filters can cope with message loss and the unpredictability of wireless links.

Contribution 2. *A large portion of this thesis concerned the spectral analysis of graph filters.* Our most important results were:

1. *We examined how low-pass filters eliminate the extrema of a graph signal.* Our contribution was two-fold: (*i*) In Lemma 3.1, we showed that the number of extrema of each eigenvector is equal to the number of its weak nodal domains. As such, the k-th signal component can have at most k extrema. (*ii*) We provided an impossibility result on the limiting behavior of low-pass filters. Even though unimodality often occurs experimentally, Corollary 3.1 asserts that no low-pass filter exists that guarantees unimodality under all possible

inputs. Our results have important implications for the search scope of greedy search methods. While low-pass filters tend to increase the search scope by eliminating local extrema, rather sadly, no filter can guarantee a global search scope.

2. *We gave guidelines for filter parametrization.* The efficiency of graph filters directly depends on the correct choice of filtering parameters. By controlling the spectral response, parameters essentially determine the size of signal structures in a filtered signal. The exact relation between spectral response and region size is beyond our current understanding. We can however give guidelines for parametrization when the distribution of eigenvalues is known *a-priori*. By deriving bounds on the spectral response of an ideal band-pass filter, Proposition 4.3 allows us to tune an ideal filter to favor regions of bounded size.

3. *We showed how to analyze the behavior of graph filters in a graph-independent manner.* Due to the exponential size of the search space, it is practically impossible to evaluate filters by testing them on all possible topologies. In Chapter 4, we proposed instead that their properties are quantified and compared based on their spectral response. Our method entails mapping filtering objectives, e. g., resilience to noise and detection resolution, to response properties. Using this methodology, we compared analytically the detection properties of various event region detection filters in Section 4.4. Simulation result confirmed the predictions of our analysis.

Contribution 3. *Setting the foundations of distributed scale-space theory on graphs.* Graph filters are/will be invaluable for network analysis and computation. Nevertheless, they are also inherently limited. More often than not, signals contain information over multiple scales: while coarse scales give the big picture, fine scales describe the details (and noise). By enforcing a single scale of observation, filtering inevitable results in loss of information. In the classical setting, these limitations have lead to the development of scale-space theory. According to this theory, to understand a signal we must examine it across all possible scales. Indeed, the scale-space approach is considered a standard in image analysis [75].

In Chapter 5, we set forward the foundations of graph scale-space theory. We addressed two fundamental research challenges: *First, what are the properties of graph scale-space kernels?* Based on analogues of the well known scale-space axioms, Theorem 5.1 establishes existence. Furthermore, the theorem shows that graph

scale-space kernels form a subset of low-pass graph filters. Therefore, they are inherently tied to the spectral properties of the graph. *Second, how efficiently can we compute them in a network?* Even though scale-space kernels are locally computable is a synchronous model of computation, currently known practical asynchronous algorithms are not local.

6.2. Discussion

Undoubtedly significant progress has been made. We have attained a good understanding of how graph filters process information and, moreover, we have in our arsenal a small family of filters that can be computed very efficiently in dynamic wireless networks. Yet to possess a concrete and rigorous theory of graph filters, we still have many challenges to face. To this end, the following text discusses some of the main open problems of the field. We split the discussion in two parts: the first concerns fundamental, theoretical problems, whereas the latter focuses on computation.

6.2.1. Fundamental Challenges

The central challenge we face is quantifying –in an intuitive way– the spectral properties of graph signals and filters. It must be obvious at this point that, most analysis approaches hinge on the understanding of the graph spectrum. The spectra of graphs have been at the center of researchers' attention for many years [14, 26, 114]. Many important questions however remain unanswered, especially relating to the eigenvectors of graphs. Because of the field's fascination with eigenvalues, the study of eigenvectors has been, to a large extent, neglected. Nonetheless, eigenvectors are very important for graph filters: they form the decomposition basis of the spectral transform, and thus bestow their structural properties (e. g., their variation) to the decomposition. Unfortunately, with the exception of few trivial graphs (e. g., path, ring, and lattice graphs), eigenvectors cannot be understood directly. An intuitive interpretation of the transform usually entails studying how the structural properties of eigenvectors vary.

Discrete nodal domains provide a good example. We know that the number of maximal induced subgraphs with a given sign is bounded with the order of the eigenvector (Theorem 2.1), and so are the eigenvector extrema (Lemma 3.1). Though current results ensue a basic understanding of signal components –in terms of their variation, they are insufficient for a more rigorous examination. Most signals are

composed of linear combinations of many signal components. Characterizing their nodal domains therefore entails reasoning about combinations of eigenvectors. The only available result to date that attempts to bound the nodal domains of a linear combination of eigenvectors is the Courant-Herrmann conjecture [28] described in Section 2.3.2. It is a testament to the complexity of the problem that the conjecture was widely thought to be true for almost forty years, before proven wrong [48] and that no alternative formulation for the general case has been proven since.

A second problem is finding ways to identify and overcome distortion. One of the contributions of Chapter 5 was showing that, when the network topology and the signal do not match, diffusion creates phantom effects at large scales of observation. Phantom effects often cause severe degradation in the accuracy of information processing algorithms. Phantom effects, for example, were the prime reason why simply smoothing a signal is not sufficient for identifying its extrema. Experiments confirmed that topology irregularities of our indoor testbed –long links were abundant, whereas short links were often obstructed by walls– caused the appearance of phantom extrema, i. e., extrema not corresponding to inherent structures of the underlying physical signal, and rendered graph filters inefficient. Unfortunately, the consequences of phantom effects extend beyond monitoring physical phenomena and sensor networks. One of the biggest challenges in graph analysis is finding unbiased and complete data-sets. In most cases, researchers are contend to construct their data by sampling real phenomena, or by inferring properties from experimental observations. Because graph filters exploit the graph topology to infer signal properties, using them on biased and incomplete graphs is likely to cause problems.

But what can we do about it? The first step is identifying whether a phantom effect has indeed occurred. According to Section 5.5.2, a simple way of telling phantom and real extrema apart is by looking at their deep structure: extrema born in large scales are good indications of distortion. Sadly the birth criterion provides necessary but not sufficient conditions. Complex signals often contain hidden structures revealed at large scales of observation[1]. Hidden structures can be wrongly identified as phantom if one solely considers the birth criterion. Increasing accuracy in detection and discovering more ways to repair distortion, both remain open issues.

[1] An example of a hidden structure revealed by diffusion is given in Section 5.3.1.

6.2.2. Computational challenges

Let us focus on computation. Within the context of distributed graph filters, we have identified two open challenges:

The first is systematically designing efficient algorithms for distributed filters based on how much they attenuate each signal component, i. e., their spectral response. The major advantage of the spectral response is that it allows us to study filters independently of the graph in consideration (Section 2.3.2). A case in point is Section 4.4.3, where we showed that, by mapping filtering objectives to the shape of filter response, we can compare existing graph filters systematically in terms of their resilience to noise and boundary detection resolution. Still, one may ask: what is the optimal distributed graph filter that satisfies a number of filtering objectives?

One approach to distributed graph filter design is to construct new filters based on existing ones. By combining linearly known distributed graph filters, one can build a set of computable responses. This is analogous to how, in classical filter design, one designs a high-order filter by combining simpler filters of lower order. Yet, the expressiveness of a constructive approach is unknown—especially if we limit our search to specific models of computation. What is more, even if we are able to construct filters with arbitrary spectral responses, it is unclear which one to choose.

The last challenge entails approximating synchronous graph kernels in the asynchronous model. The challenge is especially relevant for graph scale-space theory. As was shown in Chapter 5, graph scale-space kernels exist. However, even though such kernels are locally computable in the synchronous model, no local recursion exists that computes them exactly (Theorem 5.2). This raises a natural question: how well can we approximate them in an asynchronous model? Even further, can we find algorithms that converge to an approximate solution? The existence of such algorithms would enable us to lift the requirement of synchronization from the network engineer and to perform scale-space analysis on dynamic networks and signals.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications magazine*, 40(8):102–114, 2002.

[2] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[3] N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn. Bonnmotion – a mobility scenario generation and analysis tool. In *International Conference on Simulation Tools and Techniques*, SIMUTools. ICST, Apr 2010.

[4] B. Awerbuch. Complexity of network synchronization. *Journal of the ACM*, 32(4):804–823, 1985.

[5] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione. Broadcast gossip algorithms for consensus. *Transactions on Signal Processing*, 57(7):2748–2761, 2009.

[6] O. Babaoglu, G. Canright, A. Deutsch, G. A. D. Caro, F. Ducatelle, L. M. Gambardella, N. Ganguly, M. Jelasity, R. Montemanni, A. Montresor, et al. Design patterns from biology for distributed computing. *Transactions on Autonomous and Adaptive Systems*, 1(1):26–66, 2006.

[7] R. Band, I. Oren, and U. Smilansky. Nodal domains on graphs–How to count them and why? *arXiv preprint arXiv:0711.3416*, 2007.

[8] P. Barbosa, N. White, and N. Harris. Wireless sensor network for localized maritime monitoring. In *International Conference on Advanced Information Networking and Applications*, pages 681–686, 2008.

[9] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange. Sensorscope: Out-of-the-box environmental monitoring. In *International Conference on Information Processing in Sensor Networks*, IPSN, pages 332–343. ACM/IEEE, 2008.

[10] Y. Baryshnikov and R. Ghrist. Target enumeration via integration over planar sensor networks. *Proceedings of Robotics: Science and Systems IV*, 2008.

[11] M. Basu. Gaussian-Based Edge-Detection Methods – A Survey. *Transactions on Systems, Man, and Cybernetics*, 32(3):252–260, 2002.

[12] P. Basu, A. Nadamani, and L. Tong. Extremum tracking in sensor fields with spatio-temporal correlation. In *International Conference on Militaty Communications*, pages 1050–1055. IEEE, 2010.

[13] G. Berkolaiko. A lower bound for nodal count on discrete and metric graphs. *Communications in Mathematical Physics*, 278(3):803–819, 2008.

[14] T. Bıyıkoglu, J. Leydold, and P. F. Stadler. *Laplacian eigenvectors of graphs: Perron-Frobenius and Faber-Krahn Type Theorems*, volume 1915. Springer, 2007.

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *Transactions on Information Theory*, 52(6):2508–2530, June 2006.

[16] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[17] A. Carzaniga, C. Hall, and M. Papalini. Fully decentralized estimation of some global properties of a network. In *International Conference on Computer Communications*, INFOCOM, pages 630–638, Mar 2012.

[18] C. Cattuto, W. Van den Broeck, A. Barrat, V. Colizza, J.-F. Pinton, and A. Vespignani. Dynamics of person-to-person interactions from distributed rfid sensor networks. *PloS one*, 5(7):e11596–e11596, 2010.

[19] Y. Cheng. Mean shift, mode seeking, and clustering. *Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[20] M. Cho and K. M. Lee. Mode-seeking on graphs via random walks. In *International Conference on Computer Vision and Pattern Recognition*, CVPR, pages 606–613. IEEE, 2012.

[21] F. Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005.

[22] F. Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007.

[23] F. Chung, L. Lu, and V. Vu. The Spectra of Random Graphs with Given Expected Degrees. *Internet Mathematics*, 1(3):257–275, Jan. 2004.

[24] F. Chung and M. Radcliffe. On the spectra of general random graphs. *Electronic journal of Combinatorics*, 18(1):215–229, 2011.

[25] F. Chung, A. G. Yan, and S. Yau. Higher eigenvalues and isoperimetric inequalities on Riemannian manifolds and graphs. *Communications on Analysis and Geometry*, 8(6):969–1026, 2000.

[26] F. R. Chung. *Spectral graph theory*, volume 92. American Mathematical Society, 1997.

[27] M. Coates. Distributed particle filters for sensor networks. In *International Conference on Information processing in Sensor networks*, IPSN, pages 99–107. ACM/IEEE, 2004.

[28] R. Courant and D. Hilbert. Methods of mathematical physics. 1, 1953.

[29] J. Dall and M. Christensen. Random geometric graphs. *Physical Review E*, 66(1):016121, 2002.

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

[31] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.

[32] M. Ding and X. Cheng. Robust event boundary detection in sensor networks-a mixture model based approach. In *International Conference of Computer Communications*, INFOCOM. IEEE, 2009.

[33] A. Dogandzic and B. Zhang. Distributed estimation and detection for sensor networks using hidden Markov random field models. *Transactions on Signal Processing*, 54(8):3200–3215, 2006.

[34] M. Duckham, D. Nussbaum, J.-R. Sack, and N. Santoro. Efficient, decentralized computation of the topology of spatial regions. *Transactions on Computers*, 60(8):1100–1113, 2011.

[35] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268, 2006.

[36] D. Easley and J. Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[37] P. Erdős and A. Rényi. On the evolution of random graphs. In *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.

[38] F. Fagnani and S. Zampieri. Randomized consensus algorithms over large scale networks. *IEEE Selected Areas in Communications*, 26(4):634–649, 2008.

[39] Q. Fang, F. Zhao, and L. Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. In *International symposium on Mobile ad hoc networking and Computing*, pages 165–176. ACM, 2003.

[40] H. Farhangi. The path of the smart grid. *Power and Energy Magazine, IEEE*, 8(1):18–28, 2010.

[41] J. Faruque, K. Psounis, and A. Helmy. Analysis of gradient-based routing protocols in sensor networks. In *Distributed Computing in Sensor Systems*, volume 3560 of *DCOSS*, pages 258–275. Springer, 2005.

[42] J. Fourier. *Theorie analytique de la chaleur, par M. Fourier.* Chez Firmin Didot, père et fils, 1822.

[43] Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.

[44] J. Gao, L. Guibas, N. Milosavljevic, and J. Hershberger. Sparse data aggregation in sensor networks. In *International Conference on Information Processing in Sensor Networks*, IPSN, pages 430–439. ACM/IEEE, 2007.

[45] J. Gao, L. J. Guibas, J. Hershberger, and L. Zhang. Fractionally cascaded information in a sensor network. In *International Conference on Information processing in Sensor networks*, IPSN. ACM/IEEE, 2004.

[46] M. R. Garey and D. S. Johnson. The complexity of near-optimal graph coloring. *Journal of the ACM*, 23(1):43–49, 1976.

[47] G. M. Gladwell and H. Zhu. The courant-herrmann conjecture. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 83(4):275–281, 2003.

[48] G. M. L. Gladwell and H. Zhu. Courant's nodal line theorem and its discrete counterparts. *Quarterly Journal of Mechanics and Applied Mathematics*, 55:1–15, 2002.

[49] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *International Conference on Embedded Networked Sensor Systems*, SenSys, pages 1–14. ACM, 2009.

[50] J. Gould, D. Roemmich, S. Wijffels, H. Freeland, M. Ignaszewsky, X. Jianping, S. Pouliquen, Y. Desaubies, U. Send, K. Radhakrishnan, et al. Argo profiling floats bring new era of in situ ocean observations. *Transactions American Geophysical Union*, 85(19):185–191, 2004.

[51] L. K. Grover. Local search and the local structure of NP-complete problems. *Operations Research Letters*, 12(4):235–243, 1992.

[52] H. Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7):943–968, 1993.

[53] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge university press, 2012.

[54] T. Ideker and R. Sharan. Protein networks in disease. *Genome research*, 18(4):644–652, 2008.

[55] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.-F. Pinton, and W. Van den Broeck. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1):166–180, 2011.

[56] M. O. Jackson. *Social and economic networks*. Princeton University Press, 2010.

[57] M.-H. Jeong and M. Duckham. A coordinate-free, decentralized algorithm for monitoring events occurring to peaks in a dynamic scalar field. In *International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 437–442. IEEE, Apr 2013.

[58] M.-H. Jeong, M. Duckham, and A. Kealy. Decentralized and coordinate-free computation of critical points and surface networks in a discretized scalar field. *International Journal of Geographical Information Science*, 28(1), 2014.

[59] J. Jiang and M. Worboys. Detecting Basic Topological Changes in Sensor Networks by Local Aggregation. In *International Conference on Advances in Geographic Information Systems*, SIGSPATIAL, pages 4:1–4:10, 2008.

[60] G. Jin and S. Nittel. NED: An efficient noise-tolerant event and event boundary detection algorithm in wireless sensor networks. In *International Conference on Mobile Data Management*, MDM, page 153, 2006.

[61] S. Jung, M. Kserawi, D. Lee, and J.-K. Rhee. Distributed potential field based routing and autonomous load balancing for wireless mesh networks. *IEEE Communications Letters*, 13(6):429–431, 2009.

[62] J. A. Kelner, J. R. Lee, G. N. Price, and S.-H. Teng. Higher Eigenvalues of Graphs. In *Annual IEEE Symposium on Foundations of Computer Science*, pages 735–744. IEEE, Oct. 2009.

[63] D. Kempe, J. Kleinberg, and A. Demers. Spatial gossip and resource location protocols. *Journal of the ACM*, 51(6):943–967, 2004.

[64] U. A. Khan, S. Kar, and J. M. Moura. Higher dimensional consensus: Learning in large-scale networks. *Transactions on Signal Processing*, 58(5):2836–2849, 2010.

[65] R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *International Conference on Machine Learning*, volume 2, pages 315–322, 2002.

[66] F. Kuhn, T. Moscibroda, and R. Wattenhofer. What cannot be computed locally! In *Symposium on Principles of Distributed Computing*, PODC, pages 300–309. ACM, 2004.

[67] A. Kuijper. *The deep structure of Gaussian scale space images*. PhD thesis, Utrecht University, 2002.

[68] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *International Conference on World Wide Web*, WWW, pages 591–600, New York, NY, USA, 2010. ACM.

[69] O. Landsiedel, E. Ghadimi, S. Duquennoy, and M. Johansson. Low power, low delay: opportunistic routing meets duty cycling. In *International Conference on Information Processing in Sensor Networks*, IPSN, pages 185–196. ACM/IEEE, 2012.

[70] F. Li, J. Luo, C. Zhang, S. Xin, and Y. He. UNFOLD: UNiform Fast On-Line boundary Detection for Dynamic 3D Wireless Sensor Networks. In *International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc, page 1. ACM, May 2011.

[71] H. Lin, M. Lu, N. Milosavljevic, J. Gao, and L. J. Guibas. Composable information gradients in wireless sensor networks. In *International Conference on Information processing in Sensor networks*, IPSN. ACM/IEEE, 2008.

[72] T. Lindeberg. Scale-space for discrete signals. *Transactions on Pattern Analysis and Machine Intelligence*, 12(3):234–254, Mar. 1990.

[73] T. Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11(3):283–318, 1993.

[74] T. Lindeberg. *Scale-space theory in computer vision*. Springer, 1993.

[75] T. Lindeberg. *Generalized Gaussian Scale-Space Axiomatics Comprising Linear Scale-Space, Affine Scale-Space and Spatio-Temporal Scale-Space*, volume 40. Springer, Dec 2010.

[76] B. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley. Mobility improves coverage of sensor networks. In *International Symposium on Mobile ad hoc Networking and Computing*, MobiHoc, pages 300–308. ACM, 2005.

[77] Y. Liu, R. Weisberg, C. Hu, L. Zheng, Eos, and 2011. Tracking the Deepwater Horizon Oil Spill: A Modeling Perspective. *EOS Transactions, American Geophysical Union*, 92(6):45–52, 2011.

[78] M. Loog, J. J. Duistermaat, and L. M. Florack. On the behavior of spatial critical points under gaussian blurring: a folklore theorem and scale-space constraints. *Lecture notes in computer science*, pages 183–192, 2001.

[79] A. Loukas, M. Cattani, M. A. Zúñiga, and J. Gao. Graph scale-space theory for distributed peak and pit identification. In *International Conference on Information Processing in Sensor Networks*, IPSN. ACM/IEEE, Apr 2015.

[80] A. Loukas, M. Woehrle, P. Glatz, and K. G. Langendoen. On distributed computation of information potentials. In *International Workshop on Foundations of Mobile Computing*, FOMC, pages 1–10, Jul 2012.

[81] A. Loukas, M. A. Zúñiga, I. Protonotarios, and J. Gao. How to identify global trends from local decisions? event region detection on mobile networks. In *International Conference on Computer Communications*, INFOCOM, Apr 2014.

[82] A. Loukas, M. A. Zúñiga, M. Woehrle, M. Cattani, and K. Langendoen. Think globally, act locally: On the reshaping of information landscapes. In *International Conference on Information Processing in Sensor Networks*, IPSN, pages 265–276. ACM/IEEE, Apr 2013.

[83] L.-Z. Lu and C. E. M. Pearce. Some new bounds for singular values and eigenvalues of matrix products. *Annals of Operations Research*, 98(1-4):141–148, 2000.

[84] M. Martalo and G. Ferrari. Low-Complexity One-Dimensional Edge Detection in Wireless Sensor Networks. *EURASIP Journal on Wireless Communications and Networking*, 2(11):6:1–6:13, 2010.

[85] C. Martella, M. Dobson, A. van Halteren, and M. van Steen. From proximity sensing to spatio-temporal social graphs. In *International Conference on Pervasive Computing and Communications*, PerCom, pages 78–87. IEEE, 2014.

[86] C. Martella, M. Steen, A. Halteren, C. Conrado, and J. Li. Crowd textures as proximity graphs. *Communications Magazine*, 52(1):114–121, 2014.

[87] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Conference on Embedded Network Sensor Systems*, pages 337–350. ACM, 2008.

[88] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali. Routing without routes: The backpressure collection protocol. In *International Conference on Information Processing in Sensor Networks*, IPSN, pages 279–290. ACM/IEEE, 2010.

[89] J. Molina-Garcia-Pardo, A. Martinez-Sala, M. Bueno-Delgado, E. Egea-Lopez, L. Juan-Llacer, and J. Garcia-Haro. Channel model at 868 mhz for wireless sensor networks in outdoor scenarios. In *International Workshop on Wireless Ad-Hoc Networks*, IWWAN, pages 2–5. IEEE, 2005.

[90] M. E. Newman. Spread of epidemic disease on networks. *Physical review E*, 66(1):016128, 2002.

[91] S. Nittel. A survey of geosensor networks: Advances in dynamic environmental monitoring. *Sensors*, 9(7):5664–5678, 2009.

[92] S. Nittel, A. Labrinidis, and A. Stefanidis. Introduction to advances in geosensor networks. In *GeoSensor Networks*, pages 1–6. Springer, 2008.

[93] R. Nowak, U. Mitra, and R. Willett. Estimating inhomogeneous fields using wireless sensor networks. *Selected Areas in Communications*, 22(6):999–1006, 2004.

[94] P. Ogren, E. Fiorelli, and N. Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *Transactions on Automatic Control*, 2004.

[95] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In *Conference on Decision and Control*, pages 5492–5498. IEEE, 2007.

[96] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *International Conference on Local Computer Networks*, pages 641–648. IEEE, Nov 2006.

[97] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[98] V. Y. Pan and Z. Q. Chen. The complexity of the matrix eigenproblem. In *Annual symposium on Theory of computing*, pages 507–516. ACM, 1999.

[99] D. Roggen, M. Wirz, G. Tröster, and D. Helbing. Recognition of crowd behavior from mobile sensors with pattern analysis and graph clustering methods. *arXiv preprint arXiv:1109.1664*, 2011.

[100] A. Sandryhaila and J. M. Moura. Discrete signal processing on graphs: Frequency analysis. *Transactions on Signal Processing*, 2014.

[101] R. Sarkar, X. Zhu, and J. Gao. Hierarchical spatial gossip for multiresolution representations in sensor networks. *Transactions on Sensor Networks*, 8(1):4, 2012.

[102] R. Sarkar, X. Zhu, J. Gao, L. J. Guibas, and J. S. Mitchell. Iso-contour queries and gradient descent with guaranteed delivery in sensor networks. In *International Conference on Computer Communications*, INFOCOM. IEEE, 2008.

[103] R. Sarkar, X. Zhu, J. Gao, L. J. Guibas, and J. S. B. Mitchell. Iso-contour queries and gradient descent with guaranteed delivery in sensor networks. In *International Conference on Computer Communications*, INFOCOM. IEEE, 2009.

[104] A. D. Sarwate and A. G. Dimakis. The impact of mobility on gossip algorithms. In *International Conference on Computer Communications*, INFOCOM, pages 2088–2096. IEEE, 2009.

[105] E. Seo, P. Mohapatra, and T. Abdelzaher. Identifying rumors and their sources in social networks. In *SPIE Defense, Security, and Sensing*, pages 83891I–83891I. International Society for Optics and Photonics, 2012.

[106] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. In *Pattern Analysis and Machine Intelligence*, pages 888–905, 2000.

[107] K. Shih, S. Wang, H. Chen, and P. Yang. CollECT: Collaborative event detection and tracking in wireless heterogeneous sensor networks. *Computer Communications*, 2008.

[108] D. Shoup. Cruising for parking. *Transport Policy*, 13(6):479–486, 2006.

[109] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

[110] D. I. Shuman, P. Vandergheynst, and P. Frossard. Chebyshev polynomial approximation for distributed signal processing. In *International Conference on Distributed Computing in Sensor Systems and Workshops*, DCOSS, pages 1–8. IEEE, 2011.

[111] A. Skordylis, N. Trigoni, and A. Guitton. A study of approximate data management techniques for sensor networks. In *International Workshop on Intelligent Solutions in Embedded Systems*, pages 1–12. IEEE, 2006.

[112] P. Skraba, Q. Fang, A. Nguyen, and L. Guibas. Sweeps over wireless sensor networks. In *International Conference on Information processing in Sensor networks*, IPSN. ACM/IEEE, 2006.

[113] A. J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning theory and kernel machines*, pages 144–158. Springer, 2003.

[114] P. Van Mieghem. *Graph spectra for complex networks*. Cambridge University Press, 2011.

[115] C. M. Vigorito, D. Ganesan, and A. G. Barto. Adaptive control of duty cycling in energy-harvesting wireless sensor networks. In *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, SECON, pages 21–30. IEEE, 2007.

[116] U. von Luxburg and U. Luxburg. A Tutorial on Spectral Clustering. *Statistics and Computing*, 17(4):395–416, Nov. 2007.

[117] T.-Y. Wang and Q. Cheng. Collaborative Event-Region and Boundary-Region Detections in Wireless Sensor Networks. *Transactions on Signal Processing*, 56(6):2547–2561, 2008.

[118] Y. Wang, J. Gao, and J. Mitchell. Boundary recognition in sensor networks by topological methods. In *International Conference on Mobile Computing and Networking*, MOBICOM, pages 122–133, 2006.

[119] W. Wei and Y. Qi. Information potential fields navigation in wireless ad-hoc sensor networks. *Sensors*, 11(5):4794–4807, 2011.

[120] E. P. Wigner. On the distribution of the roots of certain symmetric matrices. *Annals of Mathematics*, pages 325–327, 1958.

[121] Wikipedia. Laplacian matrix — Wikipedia, the free encyclopedia, 2014. [Online; accessed 29-July-2004].

[122] A. P. Witkin. Scale-space filtering: A new approach to multi-scale description. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 9 of *ICASSP*, pages 150–153. IEEE, 1984.

[123] M. Woehrle, M. Bor, and K. Langendoen. 868 MHz: a noiseless environment, but no free lunch for protocol design. In *International Conference on Networked Sensing Systems*, INSS, pages 1–8, Jun 2012.

[124] H. Xu and S.-T. Yau. Nodal domain and eigenvalue multiplicity of graphs. *Journal of Combinatorics*, 3(4):609–622, 2012.

[125] Y. Yang, Y. Liu, Q. Zhang, and L. Ni. Cooperative boundary detection for spectrum sensing using dedicated wireless sensor networks. In *International Conference of Computer Communications*, INFOCOM. IEEE, 2010.

[126] M. Z. Zamalloa and B. Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *Transactions on Sensor Networks*, 3(2):7, 2007.

[127] K. Zeng, K. Ren, W. Lou, and P. J. Moran. Energy aware efficient geographic routing in lossy wireless sensor networks with environmental energy supply. *Transactions on Wireless Networks*, 15(1):39–51, 2009.

[128] F. Zhang and E. R. Hancock. Image scale-space from the heat kernel. In *Progress In Pattern Recognition, Image Analysis And Applications*, pages 181–192. Springer, 2005.

[129] F. Zhang and E. R. Hancock. Graph spectral image smoothing using the heat kernel. *Pattern Recognition*, 41(11):3328–3342, 2008.

[130] H. Zhou, H. Wu, and M. Jin. A robust boundary detection algorithm based on connectivity only for 3D wireless sensor networks. In *International Conference of the Computer Communications*, INFOCOM, pages 1602–1610. IEEE, Mar. 2012.

[131] X. Zhu, J. Lafferty, and R. Rosenfeld. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Language Technologies Institute, School of Computer Science, 2005.

[132] X. Zhu and M. Rabbat. Approximating signals supported on graphs. In *International Conference on Acoustics, Speech and Signal Processing*, ICASSP, pages 3921–3924, 2012.

[133] X. Zhu, R. Sarkar, and J. Gao. Topological data processing for distributed sensor networks with morse-smale decomposition. In *International Conference on Computer Communications*, INFOCOM, pages 2911–2915. IEEE, 2009.

Acknowledgments

This thesis would not have been possible without the support of three main people: Matthias Woehrle, Marco Zúñiga, and Koen Langendoen. Matthias helped me take my first awkward steps in research. He had the patience to listen to my confused thoughts and provided solid and unwavering support. Our long discussions encouraged me to overcome the many challenges of the first two years of my PhD. Marco believed in me greatly. His contribution was paramount. He provided clarity and structure to my thinking. He urged me to improve in the best possible way—by never hesitating to share the load when I needed it. His simple, selfless, and unassuming supervision, has been -and continuous to be- a role-model for me. I am also thankful to my promoter, Koen, foremost for his trust. Koen was a fatherly figure during my PhD. He believed in me when I did not, pushed me when I needed it the most, gave me the freedom to pursue my own path, and supported me throughout—for that I am grateful.

A number of other people have also given me invaluable support over the last years. Jie Gao, who hosted me during the winter of 2012, has been a source of inspiration and a fun person to work with. My many colleagues from EWI –especially Marco, Andrei, Ioannis, Niels, Yunus, Venkat, Przemek, Phillip, and Akshay– have been great company and friends; I enjoyed our many discussions greatly! The members of my committee, to whom I am indebted to for their valuable comments.

Lastly, I would like to express my gratitude to my family and friends for being beside me in joy and sorrow. My parents especially, who have given me unconditional love and support. They serve to me as the greatest examples of character, simplicity, and selflessness. My partner Theodora who has imbued my life with purpose. She is a constant source of inspiration and beauty. Whether on the mountains of Epirus, on the canals of Delft, or at the fields of Lancaster, I feel blessed to have embarked with her in the journey of life. My dear friend Marida, who inspired me to love science. My loving grandparents for their humble wisdom. My second-parents Giannis and Litsa, who I deeply miss. My close friends, Andreas (×2), Philipos, Fanis, Arman, Saman, Guiseppe, Andrea, and Riccardo, for accepting me despite my many faults.

Andreas Loukas
Delft, February 2015