Institute of Telecommunications

Master Thesis

Efficient Iterative Reconstruction of Sampled Graph Signals

Sara Abbasi
1329650

Supervisor:

Univ.Prof. Dipl.-Ing. Dr.-Ing. Norbert Görtz

June 2018

Abstract

Graph signal processing is an emerging field of large-scale data analysis that aims to provide useful algebraic signal processing tools to process data defined on graph domains. We use the algebraic approach of graph signal processing to introduce basic concepts, such as graph filters, graph Fourier transform, sampling on graphs and the perfect recovery of sampled graph signals.

In this thesis, we investigate the performance of an iterative recovery algorithm proposed by N. Görtz (2017). The considered method is universally applicable to general graph weight matrices and has an analytic upper error bound that allows to predict the number of iterations as a function of the desired accuracy.

The performance of the iterative algorithm for different weight matrices with entries taken from different random distributions (either uniform, Gaussian or discrete distribution) is compared. Moreover, the impact of multiple factors, such as the signal dimension, the size of the frequency support set, the number of measured signal components, and the sparsity factor of the weight matrices is analyzed for different desired accuracies.

The simulations show that the iterative algorithm performs considerably better in terms of complexity than the classical matrix inversion used in the perfect recovery theorem, when a proper proportion between the signal dimension, frequency support set and number of measured signal components is chosen. Moreover, a significant reduction of the required number of iterations can be achieved, when a lower accuracy of the recovered signal is allowed. Furthermore, the analysis of the results suggested an approach to avoid the computation of the largest and smallest eigenvalue of each realization of the weight matrix, that allows further complexity reduction of the iterative algorithm.

Acknowledgment

Human beings are members of a whole,
In creation of one essence and soul.
If one member is afflicted with pain,
Other members uneasy will remain.
If you've no sympathy for human pain,
The name of human you cannot retain!

Sa'adi
(translated by M. Aryanpoor)

I would like to use this opportunity to express my gratitude to my family and all my friends who made this world a better place for me.

Thanks to my supervisor, Univ.Prof. Dr. Norbert Görtz for all his guidance and patience, who not only encouraged me by his positivity but also inspired me with his work ethics.

I would like to thank my love, supportive colleague and my best friend, Julian, for being near me when I needed him, for embracing my concerns and making my life more cheerful.

Finally, I must express my most profound gratitude to my parents for providing me with tremendous support throughout my life. Thanks to Sina and Soudabeh, whom I owe an important debt for their love, support and generosity. This accomplishment would not have been possible without them. Thank you!

Contents

1 Introduction **9**

2 Graph Signal Processing **11**
2.1 Fundamental Concepts and Definitions 11
2.1.1 Graphs and Directed Graphs 11
2.1.2 Weighted Graphs 12
2.2 Graph Signals 13
2.3 Graph Shift 15
2.4 Graph Filters 16
2.5 Spectral Decomposition and Graph Fourier Transform 20
2.6 Sampling on Graphs 25
2.6.1 Sampling and Interpolation 25
2.6.2 Bandlimited Graph Signals 26
2.6.3 Recovery of Sampled Graph Signals 27

3 Iterative Recovery **31**
3.1 Derivation of Iterative Graph Signal Recovery 31
3.2 Analytic Error-Bound of the Recovery Algorithm 35

4 Numerical Experiments **39**
4.1 Simulation parameters 39
4.2 Simulation outcome 41

5 Conclusion **49**

Appendices **51**

A Detailed Simulation Outcomes–Result Tables **53**

B Detailed Simulation Outcomes–Figures **57**

Bibliography **63**

Chapter 1

Introduction

Nowadays, almost every aspect of human life is being recorded at all levels, which results in rapidly growing volumes of data. Extracting valuable information from the flood of information is always worthy in our life, and when it comes to the digital world, it is also beneficial and requires innovative approaches.

Over the last century, signal processing has been used as one of the most important techniques to analyze and synthesize time signals stemming from various sources. Due to the explosive growth of information and communication, classical signal processing can not solely solve the problems, therefore new approaches need to be developed to represent and process large data sets with complex structure. In order to benefit from the powerful algebraic signal processing tools, graphs are taken into consideration to model the data sets and their relations. Graphs, with their graphical representation, have simplified many problems and by merging traditional signal processing for one-dimensional time signals with graph theory, a new method, which is called graph signal processing, was developed to cope with state of the art challenges.

Therefore, in this work, we start by giving some introduction to graph theory and to some discrete-time signal processing concepts that are merged to define the fundamentals of graph signal processing subsequently. Following the previously published frameworks in this area, we present an optimum solution to recover graphs from sampled graph signals by an efficient iterative algorithm [1]. Several simulations were carried out to examine the performance of the iterative algorithm, and a further possible improvement of the algorithm was tested in the scope of this thesis. In the end, we conclude by a summary of the obtained results and present some suggestions for future work.

Chapter 2

Graph Signal Processing

Signal processing is an enormous and diverse field that has a long and rich history. The emergence of digital computers and microprocessors caused a major shift in signal processing, from analog to digital technologies. Digital technologies were also accompanied by important theoretical developments, such as the fast Fourier transform algorithm (FFT), giving a rise to the field of digital signal processing.

Graphs offer the ability to model different data sets and complex interactions among them. Therefore, one paradigm, that can be considered as a powerful approach to analyze large-scale data, is *Graph Signal Processing*, which extends classical signal processing theory to general graphs. As a result, graph signal processing has laid a strong foothold in various application domains such as sensor networks [2], image processing [3, 4, 5] and machine learning [6]. In the following, we introduce some fundamental concepts of graphs and we extend classical discrete signal processing (DSP) basics to general graphs ($\mathrm{DSP_G}$), i.e., time shift, filters, Fourier transform and sampling.

2.1 Fundamental Concepts and Definitions

Data sets are related by dependency, physical proximity, similarity or by other properties. This "relation" can be expressed through a graph that can take any arbitrary structure. Among a large variety of existing graph structures in literature, we restrict ourselves to a few classes of graphs in this thesis.

2.1.1 Graphs and Directed Graphs

Conceptually, a *graph* connects different nodes with each other. A graph G can be defined as a pair of sets $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a nonempty set of nodes and $\mathcal{E}$ is the set of edges. Fig. 2.1 shows an example for a graph.

Graphs can be classified in *directed* graphs and *undirected* graphs. A directed graph G is formed by nodes connected by directed edges and is a version of graph with ordered pairs of the elements of $\mathcal{E}$. Every arc of a directed graph has an initial and terminal

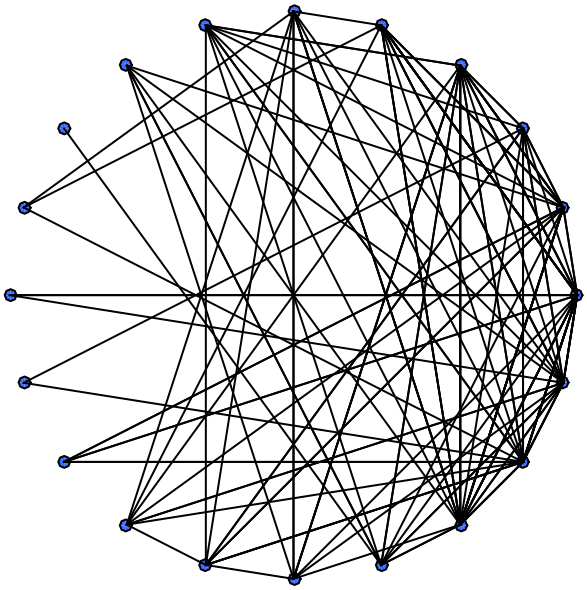


Figure 2.1: Example of a graph

node. The arc from node v_1 to node v_2 is denoted as (v_1, v_2), the opposite direction arc as (v_2,v_1). Fig. 2.2 illustrates these two types of graphs.

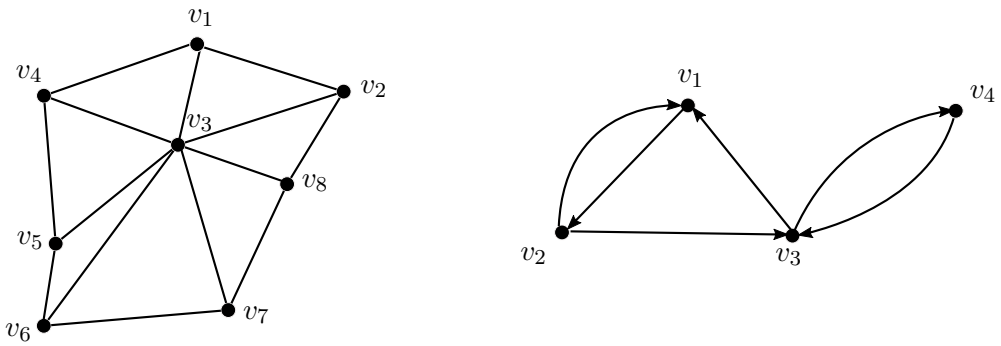


Figure 2.2: Example of (a) an undirected graph, (b) directed graph

2.1.2 Weighted Graphs

In addition to the previous definitions, a *weighted* graph G is defined as a graph, where a weight is associated to each edge. Formally, a weighted graph is then defined as the

triplet $(\mathcal{V}, \mathcal{E}, w)$, where $w : \mathcal{E} \to \mathbb{R}$ associates a real weight to each edge of the graph. Fig. 2.3 shows a weighted graph.

As an example, a common way to associate a weight w_{ij} to each edge of the graph is via a thresholded Gaussian kernel weighting function which is defined as [7]:

$$w_{ij} = \begin{cases} \exp(-\frac{d(i,j)^2}{2\theta^2}), & d(i,j) \leq \varepsilon \\ 0 & \text{otherwise} \end{cases}, \tag{2.1}$$

where $d(i, j)$ represents the distance between two different connected nodes i and j. Then we can redefine the triplet $(\mathcal{V}, \mathcal{E}, w)$ as $(\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathbf{W}$ is the weight matrix.

For a given weight matrix $\mathbf{W}$, a diagonal matrix $\mathbf{D}$ can be defined, such that the value of its ith diagonal element equals the sum of the weights of all entering edges $d_i = \sum_{j=1}^{N} w_{ij}$. The graph Laplacian $\mathbf{L}$ is then defined as

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

Therefore, for an undirected graph, the matrix $\mathbf{L}$ is a real symmetric matrix. We will see that this leads to nice and useful propreties for the spectral decomposition of graphs.

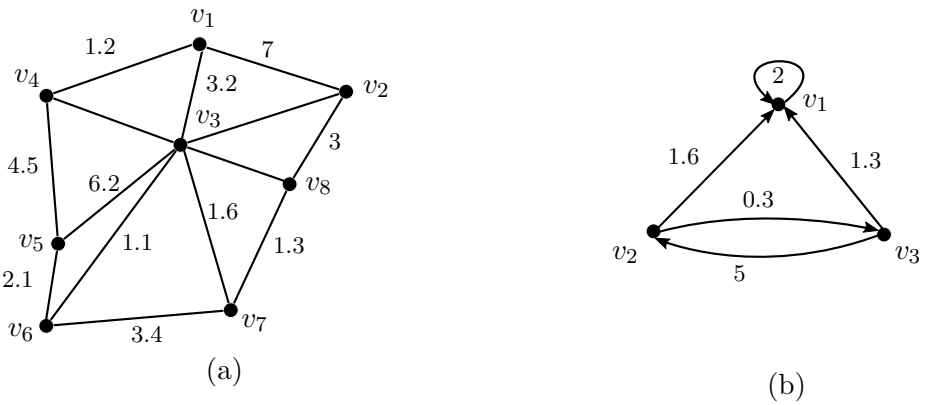


Figure 2.3: Example of (a) a weighted undirected graph, and (b) a weighted directed pseudograph[1]

2.2 Graph Signals

Considering a dataset with N elements, each data element can be seen as a node of a graph and the dataset can be analyzed via graph theory. For instance, the nodes of a graph can represent the values of temperature sensors across a country, colour values of an image pixel, all available goods of an online shopping website, lists of all movies in

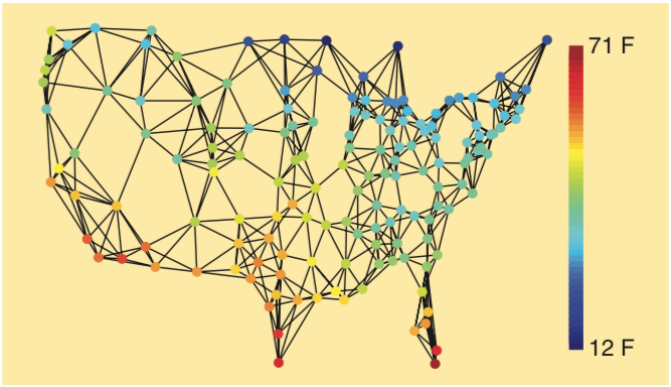
[1] A graph is called directed pseudograph, when parallel arcs and loops are admissible [8].

Netflix, lists of all songs in Spotify or Youtube etc. The graph, that shows the connection of the nodes, can take any arbitrary structure which might change slowly or fast within time.

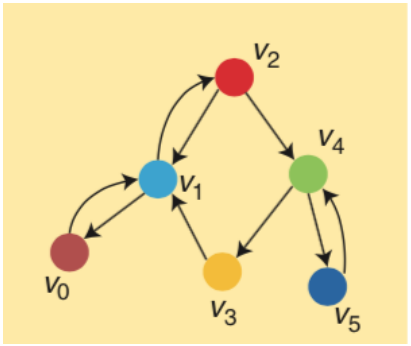
By indexing all N nodes of the graph and associating a real or complex number to each node v_i, we introduce a *graph signal*, which can be treated as an N dimensional vector $\mathbf{x} = (x_1, x_2, ..., x_N)^T$ with real or complex scalars. Each element of a graph signal vector $\mathbf{x}$ is established by a mapping of the node v_i from the set $\mathcal{V}$ into the set of complex numbers $\mathbb{C}$ according to:

$$\mathbf{x} : \mathcal{V} \to \mathbb{C}, \quad v_i \mapsto x_i. \tag{2.2}$$

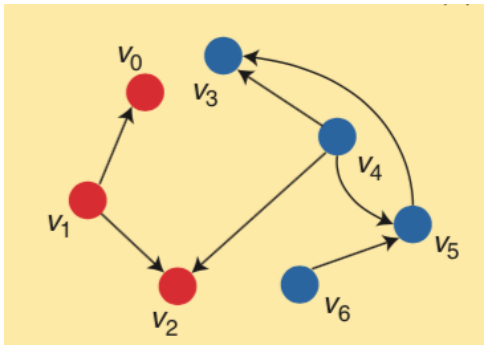
Fig. 2.4 shows three examples of graph signals.



(a)



(b)



(c)

Figure 2.4: Three examples of graph signals: (a) Temperature measurements across the United States reside on the graph that represents the network of weather sensors; (b) Web site topics are encoded as a signal that resides on the graph formed by hyperlinks between the Web sites; (c) The average numbers of tweets for Twitter users are encoded as a signal that resides on the graph representing who follows whom. The figures and the respective interpretation are taken from [9].

2.3 Graph Shift

In DSP, one of the basic nontrivial operations on a signal is delaying the input signal $\mathbf{x}$ by one sample. Therefore, the nth sample of the output can be expressed as $\tilde{x}_n = x_{n-1} \text{mod } N$. The shifted signal can be written as

$$\tilde{\mathbf{x}} = [\tilde{x}_0 \dots \tilde{x}_{N-1}]^T = \mathbf{C}\mathbf{x}, \tag{2.3}$$

where $\mathbf{C}$ is the $N \times N$ cyclic shift matrix (only nonzero entries are shown) [9].

$$\mathbf{C} = \begin{pmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{pmatrix}. \tag{2.4}$$

The corresponding time series graph to the adjacency matrix (2.4) is shown in Fig. 2.5.

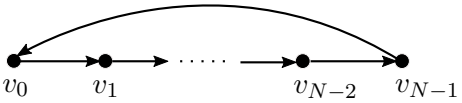


Figure 2.5: Directed line graph representing periodic time series

In DSP_G the notion of *shift* is extended to general graphs by defining the operator *graph shift* that replaces a signal value x_i at node v_i with a weighted linear combination of the signal values at its neighbours

$$\tilde{x}_n = \sum_{m=0}^{N-1} w_{n,m} x_m = \sum_{m \in \mathcal{N}_n} w_{n,m} x_m, \tag{2.5}$$

where $\mathcal{N}_n$ denotes the set of neighbours of x_n as $\mathcal{N}_n = \{m | w_{n,m} \neq 0\}$. Note that as opposed to classical DSP, shifting a finite signal does not need the consideration of boundary conditions, since in DSP_G, the graph $\mathbf{G} = (\mathcal{V}, \mathbf{W})$ captures the boundary conditions.

Therefore, the graph shift or *weighted adjacency* matrix is defined as an $N \times N$ matrix $\mathbf{W}$ that describes the "relation" between the components x_i, $i = 1, \dots, N$ and x_j, $j = 1, \dots, N$.

$$\tilde{\mathbf{x}} = [\tilde{x}_0 \dots \tilde{x}_{N-1}]^T = \mathbf{W}\mathbf{x}. \tag{2.6}$$

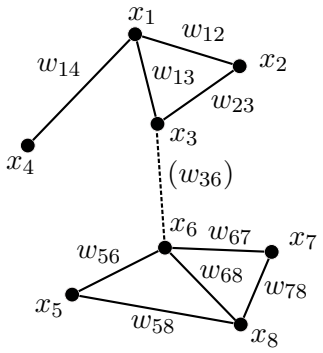
The elements of matrix $\mathbf{W}$ are assumed to be real, $w_{ij} \in \mathbb{R}$, $i, j = 1, 2, \dots, N$ and the neighbourhood relationships between two signal components x_i and x_j are not necessarily symmetric, e.g. in a directed graph, w_{ij} may differ from w_{ji} ($w_{ij} \neq w_{ji}$).

It is commonly assumed that weighted adjacency matrices are symmetric ($w_{ij} = w_{ji}$) (undirected graphs) with real positive weights $w_{ij} > 0$ and that the graph signals are *smooth*. A smooth graph signal is defined as a signal with small values of some *variation metric* $S(x)$ which can be defined as

$$S(x) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(x_i - x_j)^2. \tag{2.7}$$

Therefore, the signal model can be defined implicitly, e.g.: $x \in \mathcal{X} = \{x : S(x) < \varepsilon ||x||_2^2\}$ with $\varepsilon \in \mathbb{R}_+$.

In Fig. (2.6) an undirected graph (symmetric) and its weighted adjacency matrix are depicted. In this graph, there are two disconnected sub-graphs ($w_{36} = 0$) and no self-loop, $w_{jj} = 0$. Note that in general self-loops might exist.



$$\mathbf{W}= \begin{pmatrix} 0 & w_{12} & w_{13} & w_{14} & 0 & 0 & 0 & 0 \\ w_{12} & 0 & w_{23} & 0 & 0 & 0 & 0 & 0 \\ w_{13} & w_{23} & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{56} & 0 & w_{58} \\ 0 & 0 & 0 & 0 & w_{56} & 0 & w_{67} & w_{68} \\ 0 & 0 & 0 & 0 & 0 & w_{67} & 0 & w_{78} \\ 0 & 0 & 0 & 0 & w_{58} & w_{68} & w_{78} & 0 \end{pmatrix}$$

Figure 2.6: Graph shift and corresponding weighted adjacency matrix

The relation (2.5) can be described as a first-order interpolation, regression on graphs, or weighted averaging that are broadly used in telecommunication, distributed consensus, Markov processes and other approaches. Using definition (2.5) leads to a signal processing framework for linear and commutative graph filters and respectively well-defined concepts of the Fourier transform [9].

2.4 Graph Filters

In classical DSP, a filter is defined as a system $\mathbf{H}(\cdot)$ which takes a signal as an input and returns another signal as an output, i.e.,

$$\tilde{\mathbf{x}} = [\tilde{x}_0 \dots \tilde{x}_{N-1}]^T = \mathbf{H}(x). \tag{2.8}$$

Linear shift-invariant (LSI) filters are the most familiar filters used in digital signal processing for time series. A filter is linear, if for any linear combination of input signals, the output signal can be expressed as

$$\mathbf{H}(\alpha x_1 + \beta x_2) = \alpha \mathbf{H}(x_1) + \beta \mathbf{H}(x_2). \tag{2.9}$$

Filters $\mathbf{H_1}(\cdot)$ and $\mathbf{H_2}(\cdot)$ are said to be shift invariant or commutative, if there is no preference in the order of their application to a signal:

$$\mathbf{H}_1(\mathbf{H}_2(x)) = \mathbf{H}_2(\mathbf{H}_1(x)) \tag{2.10}$$

Another way to represent signals and filters in DSP is via the z-transform.A time delay or shift is denoted as z^{-1}, and the LSI filters can be expressed as polynomials in z^{-1} [9]

$$h(z^{-1}) = \sum_{n=0}^{N-1} h_n z^{-n}. \tag{2.11}$$

In a similar manner, finite time signals are given by

$$x(z^{-1}) = \sum_{n=0}^{N-1} x_n z^{-n}. \tag{2.12}$$

Consequently, the output signal is the multiplication of the z-transform of the filter with the input signal z-transform modulo the polynomial $z^{-N} - 1$ [10]

$$\tilde{x}(z^{-1}) = \sum_{n=0}^{N-1} \tilde{x}_n z^{-n} = h(z^{-1})x(z^{-1}) \bmod (z^{-N} - 1). \tag{2.13}$$

Equivalently, the output signal can be written as [11]

$$\tilde{x} = h(\mathbf{C})x, \tag{2.14}$$

with

$$h(\mathbf{C}) = \sum_{n=0}^{N-1} h_n \mathbf{C}^n = \begin{pmatrix} h_0 & h_{N-1} & \dots & h_1 \\ h_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & h_{N-1} \\ h_{N-1} & \dots & h_1 & h_0 \end{pmatrix}, \tag{2.15}$$

where the circulant matrix $h(\mathbf{C})$ in (2.15) is taken by substitution of the cyclic shift matrix (2.4) into the filter z-transform (2.11). In finite-time DSP, this kind of substitution form a surjective mapping froms the space of LSI filters onto the space of $N \times N$ matrices [9].

In $\mathrm{DSP_G}$, the concept of classical DSP filters is extended to graphs. In an analogous manner as the time shift, (2.3) is extended to the graph shift (2.6), filters (2.15) are extended to graph filters. For this purpose, we first define the *minimum polynomial* and *characteristic polynomial* of a matrix.

Definition 2.1:
For a given matrix $\mathbf{A}$ of size $N \times N$, the monic polynomial $m_{\mathbf{A}}(x)$ of smallest degree such that $m_{\mathbf{A}}(\mathbf{A}) = \mathbf{0}_N$ is called the minimal polynomial of the matrix $\mathbf{A}$.

The degree of the minimal polynomial is $\deg(m_{\mathbf{A}}(x)) = N_{\mathbf{A}} \leq N$.

Definition 2.2:
For a given matrix $\mathbf{A}$ of size $N \times N$, its characteristic polynomial is defined as $p_{\mathbf{A}}(x) = \det(x\mathbf{I} - \mathbf{A})$.

By applying the previous definitions, the following theorem proves that any linear, shift-invariant graph filter can be described as a polynomial in the shift $\mathbf{W}$.

Theorem 2.1 (adopted from [12],[13],[14]):
Let $\mathbf{W}$ be the graph adjacency matrix and assume that its characteristics and minimal polynomials are equal, $p_{\mathbf{W}}(x) = m_{\mathbf{W}}(x)$. Then, a graph filter $\mathbf{H}$ is linear and shift invariant if and only if (iff) $\mathbf{H}$ is a *polynomial* in the graph shift $\mathbf{W}$, i.e., iff there exists a polynomial

$$h(x) = h_0 + h_1 x + \cdots + h_L x^L \tag{2.16}$$

with possibly complex coefficient $h_l \in \mathbb{C}$, such that :

$$\mathbf{H} = h(\mathbf{W}) = h_0\mathbf{I} + h_1\mathbf{W} + \cdots + h_L\mathbf{W}^L. \tag{2.17}$$

Proof: Since the shift-invariance condition $\mathbf{W}(\mathbf{H}\mathbf{x}) = \mathbf{H}(\mathbf{W}\mathbf{x})$ holds for all graph signals $\mathbf{x} \in \mathcal{X} = \mathbb{C}^N$, the matrices $\mathbf{W}$ and $\mathbf{H}$ commute: $\mathbf{W}\mathbf{H} = \mathbf{H}\mathbf{W}$. As $p_{\mathbf{W}}(x) = m_{\mathbf{W}}(x)$, all eigenvalues of $\mathbf{W}$ have exactly one eigenvector associated with them. Then, the graph matrix $\mathbf{H}$ commutes with the shift $\mathbf{W}$ iff it is a polynomial in $\mathbf{W}$ [12].

The condition $p_{\mathbf{W}}(x) = m_{\mathbf{W}}(x)$ required in Theorem 2.1 is not always fulfilled. For this purpose, the concept of *equivalent* graph filters is introduced.

Definition 2.3:
Two filters $h(\mathbf{W})$ and $g(\tilde{\mathbf{W}})$ are said to be equivalent, if for any input $x \in \mathcal{X}$, $h(\mathbf{W})\mathbf{x} = g(\tilde{\mathbf{W}})\mathbf{x}$ for any graph shift matrices $\mathbf{W}$ and $\tilde{\mathbf{W}}$.

Therefore, for a graph $G = (\mathcal{V}, \mathbf{W})$, where the condition $p_{\mathbf{W}}(x) = m_{\mathbf{W}}(x)$ does not hold, a graph $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{W}})$, with same nodes $\mathcal{V}$ but possibly different edges can be considered such that $p_{\tilde{\mathbf{W}}}(x) = m_{\tilde{\mathbf{W}}}(x)$. Then, according to following theorem, graph filters applied on G can be formulated as equivalent filter applied on $\tilde{G}$.

Theorem 2.2 [12]:
For any matrix $\mathbf{W}$, there exists a polynomial $r(x)$ and a matrix $\tilde{\mathbf{W}}$ such that $\mathbf{W} = r(\tilde{\mathbf{W}})$ and $p_{\tilde{\mathbf{W}}}(x) = m_{\tilde{\mathbf{W}}}(x)$.

The proof of Theorem 2.2 can be found in [12].

Consequently, the condition $p_{\mathbf{W}}(x) = m_{\mathbf{W}}(x)$ of Theorem 2.1 can be assumed to be held by any graph, since in case a graph does not hold the condition, another graph $\tilde{G} = (\mathcal{V}, \tilde{\mathbf{W}})$ for which the condition is fulfilled can be found and $\tilde{\mathbf{W}}$ can be assigned to $\mathbf{W}$.

The polynomial division of the filter polynomial $h(x)$ by the minimal polynomial $m_{\mathbf{W}}(x)$ can be expressed as

$$h(x) = q(x)m_{\mathbf{W}}(x) + r(x), \tag{2.18}$$

with the remainder $r(x)$ of degree $\deg(r(x)) < \deg(m_{\mathbf{W}}(x)) = N_{\mathbf{W}}$. The graph filter in (2.17) then equals

$$h(\mathbf{W}) = q(\mathbf{W})m_{\mathbf{W}}(\mathbf{W}) + r(\mathbf{W}) = q(\mathbf{W})\mathbf{0}_N + r(\mathbf{W}) = r(\mathbf{W}). \tag{2.19}$$

Therefore, $h(\mathbf{W}) = r(\mathbf{W})$ and $\deg(h(x)) = \deg(r(x)) < \deg(m_{\mathbf{W}}(x)) = N_{\mathbf{W}}$. Having shown that the number of taps in any graph filter is limited to $N_{\mathbf{W}}$, the set of all graph filters on a graph $G = (\mathcal{V}, \mathbf{W})$ can be defined as

$$\mathcal{F} = \left\{ \mathbf{H} : \mathbf{H} = \sum_{l=0}^{N_{\mathbf{W}}-1} h_l \mathbf{W}^l \,\middle|\, h_l \in \mathbb{C} \right\}. \tag{2.20}$$

Another property of the graph filter is its invertibility.

Theorem 2.3 [12]:
A graph filter $\mathbf{H} = h(\mathbf{W}) \in \mathcal{F}$ is invertible if and only if for all distinct eigenvalues of $\mathbf{W}$, $\lambda_0, \ldots, \lambda_{M-1}$, $h(\lambda_M) \neq 0$. Hence, there exists a polynomial $g(x)$ with $\deg(g(x)) < N_{\mathbf{W}}$ which satisfies

$$h(\mathbf{W})^{-1} = g(\mathbf{W}) \in \mathcal{F}. \tag{2.21}$$

The proof of Theorem 2.3 and the way to construct the polynomial $g(x)$ can be found in [12].

Theorem 2.3 suggests that instead of inverting matrix $\mathbf{H}$, we can construct a polynomial $g(x)$ which has at most $N_{\mathbf{W}}$ taps. Finally, following Theorem 2.2 and (2.20), it can be concluded that all graph filters can be completely specified by their taps as

$$h(\mathbf{W}) = \sum_{l=0}^{N_{\mathbf{W}}-1} h_l \mathbf{W}^l. \tag{2.22}$$

The output of the graph filter is given by

$$\tilde{x} = h(\mathbf{W})x. \tag{2.23}$$

Similarly to the computation of output signals by using the z-transform in (2.13), the output signal of graph filters can also be computed by multiplication of the z-transform of the graph filters with the z-transform of the input signal. Assuming a graph signal

$$x(z^{-1}) = \sum_{n=0}^{N-1} x_n b_n(z^{-1}), \tag{2.24}$$

where $b_n(z^{-1})$, $0 \leq n < N$, are linearly independent polynomials of degree at most N (see [12] for details), the obtained output signal is

$$\tilde{x}(z^{-1}) = \sum_{n=0}^{N-1} \tilde{x}_n b_n(z^{-n}) = h(z^{-1})x(z^{-1}) \bmod m_{\mathbf{W}}(z^{-1}). \tag{2.25}$$

2.5 Spectral Decomposition and Graph Fourier Transform

Another fundamental concept in DSP is the concept of the Fourier transform. To define the Fourier transform in $\mathrm{DSP_G}$, we start by introducing spectral decomposition. By spectral decomposition, the signal space $\mathcal{X}$ is decomposed into subspaces $\mathcal{X}_k$ which are invariant to filtering. Therefore, for any signal $\mathbf{x}_k \in \mathcal{X}_k$ and filter $h(\mathbf{W}) \in \mathcal{F}$, the output lies in the same subspace $\mathcal{X}_k$.

$$\mathbf{x}_k \in \mathcal{X}_k \quad \Rightarrow \quad \tilde{\mathbf{x}}_k = h(\mathbf{W})\mathbf{x}_k \in \mathcal{X}_k \quad \forall h(\mathbf{W}) \in \mathcal{F}. \tag{2.26}$$

To decompose uniquely every signal $\mathbf{x} \in \mathcal{X}$ into K signals according to

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_{k-1}, \tag{2.27}$$

each component $\mathbf{x}_k \in \mathcal{X}_k$ must fulfil three conditions [12]:

1) no intersection of the invariant subspaces $\mathcal{X}_k$, i.e., $\mathcal{X}_k \cap \mathcal{X}_m = \{0\}$ for $k \neq m$;

2) $\dim \mathcal{X}_0 + \cdots + \dim \mathcal{X}_{K-1} = \dim \mathcal{X} = N$;

3) each $\mathcal{X}_k$ cannot be decomposed into further invariant subspaces (it is irreducible).

By holding these three conditions, the space $\mathcal{X}$ can be decomposed into

$$\mathcal{X} = \mathcal{X}_1 \oplus \mathcal{X}_2 \oplus \cdots \oplus \mathcal{X}_{k-1}, \tag{2.28}$$

where $\oplus$ is the symmetric difference operator.

Since the graph can take an arbitrary structure, the Adjacency matrix $\mathbf{W}$ might not be diagonalizable. Therefore, we introduce the *Jordan decomposition* of a matrix in order to formulate spectral decomposition of the signal space $\mathcal{X}$ and to express an expansion of the graph signal $\mathbf{x}$.

Definition 2.4:
The Jordan decomposition of a matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$ is defined as

$$\mathbf{A} = \mathbf{V}\mathbf{J}\mathbf{V}^{-1}, \tag{2.29}$$

with the Jordan normal form $\mathbf{J}$ of the matrix $\mathbf{A}$ and the matrix $\mathbf{V}$ that contains the generalized eigenvectors of $\mathbf{A}$ [15].

In general, the matrix $\mathbf{A}$ has $M \leq N$ distinct eigenvalues $\lambda_0, \dots \lambda_{M-1}$. If $M = N$, the Jordan normal form $\mathbf{J}$ is composed of N different 1×1 Jordan blocks $\mathbf{J}_1(\lambda_m)$, $m = 0, 1, \dots, N-1$, that are equal to the eigenvalues of $\mathbf{A}$,

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_1(\lambda_0) & & \\ & \ddots & \\ & & \mathbf{J}_1(\lambda_{M-1}) \end{pmatrix}. \tag{2.30}$$

The matrix $\mathbf{V}$ is then formed by the eigenvectors of $\mathbf{A}$.

Otherwise, if $M < N$, D_m independent eigenvectors $\mathbf{v}_{m,d}$ can be found for each eigenvalue λ_m by solving the linear equation

$$(\mathbf{A} - \lambda_m \mathbf{I})\mathbf{v}_{m,d} = \mathbf{0}, \quad d = 0, \dots, D_m - 1. \tag{2.31}$$

D_m is denoted as geometric multiplicity of λ_m. Each eigenvector $\mathbf{v}_{m,d}$ generates a Jordan chain composed of $R_{m,d} \geq 1$ linear independent generalized eigenvectors $\mathbf{v}_{m,d,r}$.
The generalized eigenvectors can be found recursively [12] according to

$$\mathbf{v}_{m,d,0} = \mathbf{v}_{m,d} \quad \text{and} \tag{2.32}$$

$$(\mathbf{A} - \lambda_m \mathbf{I})\mathbf{v}_{m,d,r} = \mathbf{v}_{m,d,r-1}.$$

For every eigenvector, a Jordan block of dimension $R_{m,d}$ can then be expressed as

$$\mathbf{J}_{R_{m,d}}(\lambda_m) = \begin{pmatrix} \lambda_m & 1 & & & \\ & \lambda_m & 1 & & \\ & & \ddots & \ddots & \\ & & & \lambda_m & 1 \\ & & & & \lambda_m \end{pmatrix}, \tag{2.33}$$

with all other entries set to zero. Furthermore, all generalized eigenvectors $\mathbf{v}_{m,d,r}$, that corresponds to the eigenvector $\mathbf{v}_{m,d}$, form a $N \times R_{m,d}$ matrix

$$\mathbf{V}_{m,d} = (\mathbf{v}_{m,d,0},\ \dots\ , \mathbf{v}_{m,d,R_{m,d}-1}). \tag{2.34}$$

The matrix $\mathbf{V}$ is then obtained by grouping all matrices $\mathbf{V}_{m,d}$

$$\mathbf{V} = (\mathbf{V}_{0,0},\ \dots\ , \mathbf{V}_{M-1,D_{M-1}}). \tag{2.35}$$

Finally, the matrix $\mathbf{J}$ is composed of all Jordan blocks located on its diagonal

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{R_{0,0}}(\lambda_0) & & \\ & \ddots & \\ & & \mathbf{J}_{R_{M-1,D_{M-1}}}(\lambda_{M-1}) \end{pmatrix}. \tag{2.36}$$

By considering the Jordan decomposition of the weight matrix $\mathbf{W}$, the vector subspace $\mathcal{X}_{m,d}$ can be expressed as

$$\mathcal{X}_{m,d} = \text{span}\{\mathbf{v}_{m,d,0}, \dots, \mathbf{v}_{m,d,R_{m,d}-1}\}, \tag{2.37}$$

with the generalized eigenvectors $\mathbf{v}_{m,d,r}$. Since all generalized eigenvectors are linearly independent, every signal $\mathbf{x}_{m,d} \in \mathcal{X}_{m,d}$ can be expanded according to

$$\mathbf{x}_{m,d} = \hat{x}_{m,d,0}\mathbf{v}_{m,d,0} + \ \dots\ + \hat{x}_{m,d,R_{m,d}-1}\mathbf{v}_{m,d,R_{m,d}-1}. \tag{2.38}$$

Furthermore, all subspaces $\mathcal{X}_{m,d}$ have no intersection and their dimensions add up to N, since all generalized eigenvectors of $\mathbf{W}$ are linearly independent. The spectral decomposition of $\mathcal{X}$ can then be expressed as [12]

$$\mathcal{X} = \bigoplus_{m=0}^{M-1} \bigoplus_{d=0}^{D_m-1} \mathcal{X}_{m,d}. \tag{2.39}$$

Graph Fourier Transform

The expansion of a graph signal on the basis of generalized eigenvectors is called the graph Fourier transform. Therefore, a graph signal $\mathbf{x}$ can be represented by its coefficients

$$\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}, \tag{2.40}$$

where $\mathbf{V}$ is the matrix containing the generalized eigenvectors according to (2.34). The Fourier coefficients can then be calculated as

$$\hat{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{x}. \tag{2.41}$$

Therefore, the graph Fourier transform matrix is recognized to be

$$\mathbf{F} = \mathbf{V}^{-1}. \tag{2.42}$$

By assuming a graph with a real and symmetric weight matrix that is diagonalizable and has a complete eigenbasis, the spectral decomposition of the weight matrix $\mathbf{W}$ is given by the eigenvalue decomposition

$$\mathbf{W} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}, \tag{2.43}$$

and it is a simplified version of the Jordan decomposition. Thus, in (2.43) the columns of the $N \times N$ matrix V are formed by the eigenvectors of $\mathbf{W}$, and $\mathbf{\Lambda}$ is the diagonal matrix with corresponding eigenvalues λ_0,..., $\lambda_N - 1$ on its main diagonal; no specific order of the eigenvalues is assumed [16]. To compute the Fourier transform and its inverse in the case of a diagonalizable weight matrix, the matrix $\mathbf{V}$ obtained by (2.43) is substituted in (2.41) and (2.40) respectively.

As mentioned in Section 2.1.2, the Laplacian weight matrix is real and symmetric, thus it has a complete orthonormal eigenbasis and all associated eigenvalues take real values. The corresponding graph Fourier transform is orthogonal: $\mathbf{V}^{-1} = \mathbf{V}^T$. Since in general, the graph adjacency matrix is not restricted to be symmetric, like in the case of a directed graph, we defined the general approach for graph Fourier transform, which is applicable to arbitrary weight matrices, rather than the graph Laplacian approach.

Similarly to the other adopted $\mathrm{DSP_G}$ concepts, the graph Fourier transform can be viewed also as a generalization of the discrete Fourier transform defined in DSP. For example, the mentioned finite discrete periodic time series represented by the graph in Fig. 2.5 has the eigendecomposition

$$\mathbf{C}_N = \frac{1}{N}\,\mathbf{DFT}_N^{-1} \begin{pmatrix} e^{-j\frac{2\pi.0}{N}} & & \\ & \ddots & \\ & & e^{-j\frac{2\pi.(N-1)}{N}} \end{pmatrix} \mathbf{DFT}_N, \tag{2.44}$$

where $\mathbf{DFT}_N$ is the $N \times N$ discrete Fourier transform matrix [9]. Therefore, the graph Fourier transform is defined as: $\mathbf{F} = \mathbf{DFT}_N$.

Moreover, by applying filtering on (2.44), $h(\mathbf{C}_N) = \sum_{l=0}^{N-1} h_l \mathbf{C}_N^l$, the output coefficients can be calculated to be

$$\begin{aligned} \hat{x}_n &= h_n x_0 + \cdots + h_0 x_n + h_{N-1} x_{n+1} + \cdots + h_{n+1} s_{N-1} \\ &= \sum_{K=0}^{N-1} x_k h_{(n-k \bmod N)}, \end{aligned} \tag{2.45}$$

which is the standard circular convolution. Thus, DSP_G is consistent with the traditional DSP theory.

2.6 Sampling on Graphs

In this section, we review briefly the graph sampling theory presented in the literature (see also [17, 18, 19]) and the required conditions to perform a perfect recovery of graph signals. Since graph signals can lie on any irregular, complex structure, the task of graph sampling is not, so far, well understood [18, 19]. For this part, we follow the framework presented in [16, 20].

In graph signals, the notion of frequency corresponds to the numerical values of the eigenvalues and the notion of bandwidth is introduced as the number of non-zero Fourier coefficients which makes graph sampling connected to linear algebra and allow us to benefit from the simple tools of linear algebra for performing sampling on irregular, complex graph structures.

2.6.1 Sampling and Interpolation

In order to perform sampling on a graph signal, we define the sampling operator $\mathbf{\Psi}$ which samples M coefficients of a graph signal and does a linear mapping $\mathbb{C}^N \rightarrow \mathbb{C}^M$ according to

$$\Psi_{ij} = \begin{cases} 1 & \text{if} \quad j = \mathcal{M}_i \qquad i = 1, 2, ...M, \\ & \qquad\qquad\quad j = 1, 2, ..., N \ . \\ 0 & \text{otherwise} \end{cases} \tag{2.46}$$

with the *sampling support set*

$$\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_M\} \quad \text{of size} \quad M = |\mathcal{M}| \leq N \tag{2.47}$$

and

$$\mathcal{M}_i \in \{1, 2, ..., N\} \quad \text{and} \quad \mathcal{M}_i \neq \mathcal{M}_l \quad \text{when} \quad i \neq l, \tag{2.48}$$

which denotes the the set of sampled node indices. Therefore, an M-dimensional sampled graph signal $\mathbf{x}_\mathcal{M}$ is given by

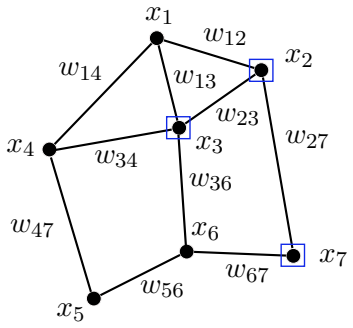
$$\mathbf{x}_\mathcal{M} = \mathbf{\Psi}\mathbf{x} \in \mathbb{C}^M. \tag{2.49}$$

As an example, Fig 2.7 illustrates sampling of a graph signal.

To recover a graph signal from a sampled graph signal, we need to interpolate $\mathbf{x}_\mathcal{M}$, hence, the interpolation operator $\mathbf{\Phi}$ needs to perform mapping $\mathbb{C}^M \rightarrow \mathbb{C}^N$. The interpolated signal equals

$$\mathbf{x}' = \mathbf{\Phi}\mathbf{x}_\mathcal{M} = \mathbf{\Psi}\mathbf{\Phi}\mathbf{x} \in \mathbb{C}^N. \tag{2.50}$$

We can consider two strategies for sampling: *random sampling* or *experimentally designed sampling*. In the case of random sampling, the sample indices are chosen randomly and



The corresponding sampling support set is

$$\mathcal{M} = \{2, 3, 7\},$$

and the sampling matrix equals

$$\mathbf{\Psi} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

$$\mathbf{x}_{\mathcal{M}} = \mathbf{\Psi}\mathbf{x} = (x_2, x_3, x_7)^T.$$

Figure 2.7: Example of graph signal with marked sampled components and the corresponding sampling support set and sampling matrix.

independently, whereas in experimentally designed sampling, samples are selected beforehand.

Obviously, perfect recovery of $\mathbf{x}$ can be performed, when $\mathbf{\Psi\Phi}$ equals the identity matrix. Since the rank$(\mathbf{\Psi\Phi}) \leq M < N$, it is generally not possible to have perfect recovery. To make it possible, the graph signal must have a specific structure; it should be a *bandlimited* graph signal.

2.6.2 Bandlimited Graph Signals

As previously mentioned, the graph frequencies define the bandwidth of the graph signal. In practice, specifying the cut-off graph frequency is not easy and computing of all graph frequencies can be hard and inefficient for large graphs. In the following, we define a band-limitation concept in graph signal proceesing [16].

Definition 2.5:
A graph signal is denoted as a *bandlimited* graph signal, when only a subset of the Fourier-coefficients is non-zero, such that

$$\hat{x}_k = 0 \quad \text{for} \quad k \in \{1, 2, ..., N\} \setminus \mathcal{K}, \tag{2.51}$$

with the *frequency support set* $\mathcal{K} = \{\mathcal{K}_1, \mathcal{K}_2, ..., \mathcal{K}_K\}$ of size $K = |\mathcal{K}| \leq N$ and

$$\mathcal{K}_i \in \{1, 2, ..., N\} \quad \text{and} \quad \mathcal{K}_i \neq \mathcal{K}_j \quad \text{when} \quad i \neq j.$$

A non-bandlimited graph signal is called a *full-band* graph signal. In graph signal processing, bandlimitation can not necessarily be interpreted as low-pass, or smooth. No ordering of the frequencies are considered, therefore, we can reorder the eigenvalues and correspondingly permuting the eigenvector in the graph Fourier transform matrix

to select any band in the Fourier domain [16]. Since the eigenvalues are not sorted in descending manner, the graph is not smooth.

If $\hat{x}_k$ is non-zero only on the known frequency support set $\mathcal{K}$, the non-zero Fourier coefficient can be calculated by

$$\hat{\mathbf{x}}_{\mathcal{K}} = (\mathbf{V}^{-1}\mathbf{x})_{\mathcal{K}} = (\mathbf{V}^{-1})_{(\mathcal{K},:)}\mathbf{x} \tag{2.52}$$

where the notation $\mathbf{C}_{(\mathcal{K},:)}$ means that all columns but only the rows contained in the index subset $\mathcal{K}$ of the matrix $\mathbf{C}$ are chosen. The inverse Fourier-transform can then be expressed as

$$\tilde{\mathbf{x}} = \mathbf{V}_{(:,\mathcal{K})}\hat{\mathbf{x}}_{\mathcal{K}} = \mathbf{V}_{(:,\mathcal{K})}(\mathbf{V}^{-1})_{(\mathcal{K},:)}\mathbf{x}. \tag{2.53}$$

2.6.3 Recovery of Sampled Graph Signals

We know from [16] that perfect recovery of an N-dimensional graph signal $\mathbf{x}$ from $\mathbf{x}_{\mathcal{M}}$ for $M < N$ can be performed by a *qualified* sampling matrix $\mathbf{\Psi}$, if the graph signal vector is adequately bandlimited, that means the number of non-zero Fourier-transform components K is not larger than the numbers of samples M, i.e. $K \leq M$.

Theorem 2.4, **Perfect recovery theorem**, adopted from [16, 1]:

Let the sampling matrix $\mathbf{\Psi}$ satisfy

$$\text{rank}(\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})}) = K, \tag{2.54}$$

where $\mathbf{V}_{(\mathcal{K},:)})$ is an $N \times K$ matrix with an index set $\mathcal{K}$ which is addressing K linear independent vectors with dimension N in the columns of $\mathbf{V}$. For the bandlimited graph signal vectors $\mathbf{x}$ with the frequency support set $\mathcal{K}$ and the Fourier-transform $\hat{\mathbf{x}} = (\mathbf{V}^{-1})\mathbf{x}$, perfect recovery

$$\mathbf{x} = \tilde{\mathbf{x}} = \mathbf{\Phi}\mathbf{x}_{\mathcal{M}} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{x} \tag{2.55}$$

is obtained by choosing

$$\mathbf{\Phi} = \mathbf{V}_{(:,\mathcal{K})}\mathbf{U} \tag{2.56}$$

with the $K \times M$ matrix $\mathbf{U}$ such that

$$\mathbf{U}\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})} = \mathbf{I} \tag{2.57}$$

where the $\mathbf{I}$ is a $K \times K$ identity matrix. The proof of Theorem 4 can be found in [21].

Analogously to the classical sampling theory, the rate of sampling for graph signals is also lower bounded, that means, the sample size M should not be smaller than the bandwidth K, because

$$\text{if } M < K \rightarrow \ \text{rank}(\mathbf{U}\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})}) \leq \text{rank}(\mathbf{U}) \leq M < K,$$

therefore, $\mathbf{U}\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})}$ does not lead to an identity matrix. Hence the perfect recovery theorem is applied only for $M \geq K$.

By applying the recovery theorem for the case of $M > K$, the matrix $\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})}$ has the dimension $M \times K$ which is not square. Consequently, the interpolation operator is defined by its pseudo-inverse [16]

$$\mathbf{U} = ((\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})})^H \mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})})^{-1} (\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})})^H. \tag{2.58}$$

For the case of $M = K$, equation (2.58) yields to a simpler solution which is inverting the $K \times K$ matrix $\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})}$. The notation can be modified to $\mathbf{\Psi}\mathbf{V}_{(:,\mathcal{K})} = \mathbf{V}_{(\mathcal{M},\mathcal{K})}$ since the sampling matrix $\mathbf{\Psi}$ picks rows from $\mathbf{V}_{(:,\mathcal{K})}$ which are defined by the sampling set $\mathcal{M}$. The interpolation formula (2.55) can be rewritten as

$$\tilde{\mathbf{x}} = \mathbf{V}_{(:,\mathcal{K})} \big(\underbrace{(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathbf{V}_{(\mathcal{M},\mathcal{K})}}_{=\mathbf{Q}} \big)^{-1} (\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathbf{x}_{\mathcal{M}}. \tag{2.59}$$

Equation (2.59) applies for any adjacency matrix for which a complete set of N linearly independent eigenvectors exist: This might also include non-symmetric matrices and those with negative weights or self loops.

To show that the condition $M \geq K$ alone is not sufficient, we present an example of a graph which contains disconnected sub-graphs.

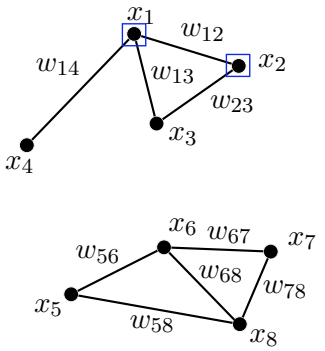
In the following, the graph signal with marked sampled components in Fig. 2.8, together with the sampling support set $\mathcal{M} = \{1, 2\}$ and the sampling matrix

$$\mathbf{\Psi} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

is considered.

As there is no connection between the non-zero components x_5, x_6, x_7, x_8 and any of the sampled components, the recovery of the non-zero components in the lower graph won't be possible. In the case of $x_5 = x_6 = x_7 = x_8 = 0$, the $M = 2$ sampled nodes in the upper graph will allow for perfect recovery, when at most $K = 2$ Fourier coefficients are non-zero but also those two must allow for non-zero components in the upper part of the graph.

Since in Theorem 2.4 the condition (2.54) must be held, we need to know the graph structure to design a qualified sampling operator. Therefore, the adopted sampling strategy will be experimentally designed sampling (see Section V in [16]).



$$\mathbf{W}= \begin{pmatrix} 0 & w_{12} & w_{13} & w_{14} & 0 & 0 & 0 & 0 \\ w_{12} & 0 & w_{23} & 0 & 0 & 0 & 0 & 0 \\ w_{13} & w_{23} & 0 & 0 & 0 & 0 & 0 & 0 \\ w_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{56} & 0 & w_{58} \\ 0 & 0 & 0 & 0 & w_{56} & 0 & w_{67} & w_{68} \\ 0 & 0 & 0 & 0 & 0 & w_{67} & 0 & w_{78} \\ 0 & 0 & 0 & 0 & w_{58} & w_{68} & w_{78} & 0 \end{pmatrix}$$

Figure 2.8: Example of graph containing disconnected sub-graphs and corresponding weighted adjacency matrix.

Considering the eigenvector matrix $\mathbf{V}$ of the graph shift in Fig. 2.8

$$\mathbf{V} = \begin{pmatrix} v_{11} & \boldsymbol{v_{12}} & v_{13} & \boldsymbol{v_{14}} & 0 & 0 & 0 & 0 \\ v_{21} & \boldsymbol{v_{22}} & v_{23} & \boldsymbol{v_{24}} & 0 & 0 & 0 & 0 \\ v_{31} & \boldsymbol{v_{23}} & v_{33} & \boldsymbol{v_{34}} & 0 & 0 & 0 & 0 \\ v_{41} & \boldsymbol{v_{24}} & v_{43} & \boldsymbol{v_{44}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_{55} & v_{56} & v_{57} & v_{58} \\ 0 & 0 & 0 & 0 & v_{65} & v_{66} & v_{67} & v_{68} \\ 0 & 0 & 0 & 0 & v_{75} & v_{76} & v_{77} & v_{78} \\ 0 & 0 & 0 & 0 & v_{85} & v_{86} & v_{87} & v_{88} \end{pmatrix}, \tag{2.60}$$

the condition $\text{rank}(\mathbf{\Psi V}_{(:,\mathcal{K})}) = K$ makes sure that the sampling matrix $\mathbf{\Psi}$ has to pick rows from $\mathbf{V}_{(:,\mathcal{K})}$ such that the resulting matrix $\mathbf{\Psi V}_{(:,\mathcal{K})}$ has rank K. For instance, the frequency support set $\mathcal{K} = \{2, 4\}$ picks the $K = 2$ column vectors in bold font from the first and second row of the matrix $\mathbf{V}$, therefore we have

$$\mathbf{\Psi V}_{(:,\mathcal{K})} = \begin{pmatrix} \boldsymbol{v_{12}} & \boldsymbol{v_{14}} \\ \boldsymbol{v_{22}} & \boldsymbol{v_{24}} \end{pmatrix} \tag{2.61}$$

which has the rank 2. Note that if for instance the frequency support set was chosen as $\mathcal{K} = \{2, 8\}$, $\mathbf{\Psi V}_{(:,\mathcal{K})}$ would only have rank 1 and recovery would not be possible.

Chapter 3

Iterative Recovery

To compute the recovered signal from the sampled signal according to (2.59), the inverse of the matrix $\mathbf{Q}$ needs to be calculated. Since $\mathbf{Q} = (\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathbf{V}_{(\mathcal{M},\mathcal{K})}$, the matrices $\mathbf{Q}$ and $\mathbf{Q}^{-1}$ depend on the sets $\mathcal{M}$ and $\mathcal{K}$. If the sets $\mathcal{M}$ and $\mathcal{K}$ known and fixed, $\mathbf{Q}^{-1}$ can be computed in advance. However, in many applications, the sampling set $\mathcal{M}$ is not fixed, e.g. sensor data is received from many sensor stations via many different unreliable links, thus the matrix $\mathbf{Q}^{-1}$ needs to be recomputed for every measurement. Therefore, an alternative algorithm to the conventional matrix inversion that allows reconstruction of the signal $\mathbf{x}$ with controllable accuracy and low complexity is presented in the following. This approach is motivated by the fact, that for many applications, very high precision of the recovered signal is not needed as long as the maximum error between the original and the recovered signal is known.

3.1 Derivation of Iterative Graph Signal Recovery

In order to avoid the matrix inversion $\mathbf{Q}^{-1}$, an iterative scheme based on concepts from [22, 23] is derived in the following. The approach is based on Neuman series, which are the geometric series for matrices and the algorithm is adopted from [1].

The matrix $\mathbf{Q}$ can be expressed as

$$\mathbf{Q} = (\mathbf{I} - \mathbf{T}_\gamma)/\gamma \quad \text{or equivalently} \quad \mathbf{T}_\gamma = \mathbf{I} - \gamma\mathbf{Q}, \quad \gamma \in \mathbb{R}. \tag{3.1}$$

If a real factor $\gamma > 0$ can be found, such that

$$\|\mathbf{T}_\gamma\|_2 < 1, \tag{3.2}$$

the inverse of $\mathbf{Q}$ (in case it exists) equals to

$$\mathbf{Q}^{-1} = \gamma \sum_{n=0}^{\infty} (\mathbf{T}_\gamma)^n = \gamma \sum_{n=0}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n, \tag{3.3}$$

whereby $||.||_2$ is the spectral matrix norm, induced by the Euclidean norm and defined as the largest eigenvalue of $\mathbf{T}_\gamma$, or the square root of the largest eigenvalue of the positive-semidefinite matrix $\mathbf{T}_\gamma^H \mathbf{T}_\gamma$

$$\|\mathbf{T}_\gamma\|_2 = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{T}_\gamma \mathbf{x}\|_2 = \sqrt{\lambda_{(\mathbf{T}_\gamma^H \mathbf{T}_\gamma),\max}}. \tag{3.4}$$

In order to apply (3.3), the condition $\|\mathbf{T}_\gamma\|_2 < 1$ must hold, that means γ needs to be chosen such that $\|\mathbf{T}_\gamma\|_2 < 1$, therefore the eigenvalues of $\mathbf{T}_{\boldsymbol{\gamma}}^H \mathbf{T}_{\boldsymbol{\gamma}}$ need to be investigated. For this, we start by discussing the eigenvalue decomposition of the matrix $\mathbf{Q}$ and derive the eigenvalues of $\mathbf{T}_{\boldsymbol{\gamma}}$ in a second step.

Since the matrix $\mathbf{Q}$ is a Gramian matrix, $\mathbf{Q}$ is positive semi-definite, so all eigenvalues of $\mathbf{Q}$ are real and non-negative. According to Theorem 2.4, we know that $\mathbf{Q}$ is invertible for $M \geq K$, thus no eigenvalue can be zero. Therefore all eigenvalues of $\mathbf{Q}$ have to be real and positive. Furthermore, since $\mathbf{Q}$ is Hermitian, its eigenvalue decomposition can be simplified to

$$\mathbf{Q} = \mathbf{V}_Q \boldsymbol{\Lambda}_Q \mathbf{V}_Q^{-1} = \mathbf{V}_Q \boldsymbol{\Lambda}_Q \mathbf{V}_Q^H \qquad \text{with} \qquad \mathbf{V}_Q \mathbf{V}_Q^H = \mathbf{I}. \tag{3.5}$$

Therefore, the matrix $\mathbf{T}_\gamma$ from (3.1) can be obtained by

$$\mathbf{T}_\gamma = \mathbf{V}_Q (\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q) \mathbf{V}_Q^H, \tag{3.6}$$

and it can be observed, that the factor γ scales the eigenvalues $\lambda_j^{(Q)}$ of $\mathbf{Q}$, which are located on the diagonal of the matrix $\boldsymbol{\Lambda}_\mathrm{Q}$.

Moreover, the largest eigenvalue of the matrix $\mathbf{T}_\gamma^H \mathbf{T}_\gamma$ needs to be identified to check the convergence condition (3.2) of (3.3). By using equation (3.6),

$$\mathbf{T}_\gamma^H \mathbf{T}_\gamma = \mathbf{V}_Q (\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q)^H \mathbf{V}_Q^H \mathbf{V}_Q (\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q) \mathbf{V}_Q^H = \mathbf{V}_Q (\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q)^2 \mathbf{V}_Q^H. \tag{3.7}$$

The diagonal matrix $\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q$ can be written as

$$\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q = \begin{pmatrix} 1 - \gamma \lambda_1^{(Q)} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 - \gamma \lambda_K^{(Q)} \end{pmatrix}. \tag{3.8}$$

Since the square of a diagonal matrix is again a diagonal matrix with the squared diagonal elements of the initial matrix, the largest eigenvalues $\lambda_{(\mathbf{T}_\gamma^H \mathbf{T}_\gamma),\max}$ is one of the diagonal elements of $(\mathbf{I} - \gamma \boldsymbol{\Lambda}_Q)^2$. Therefore, (3.2) leads to

$$\|\mathbf{T}_\gamma\|_2 = |1 - \gamma \lambda_j^{(Q)}| < 1, \qquad \forall j = 1, 2, ..., K. \tag{3.9}$$

$\gamma > 0$ must be chosen such that the convergence condition is ensured for all j. Since it is known that all eigenvalues $\lambda_j^{(Q)} > 0$, very small eigenvalues might lead to results

close to the convergence bound $\|\mathbf{T}_\gamma\|_2 = 1$ and very large eigenvalues might violate the convergence bound, $\gamma\lambda_{\max}^{(Q)} > 2$. Therefore, we choose γ by solving

$$|1 - \gamma\lambda_{\min}^{(Q)}| = |1 - \gamma\lambda_{\max}^{(Q)}|, \tag{3.10}$$

which results in

$$\gamma = 2/(\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)}). \tag{3.11}$$

Hence, the choice of γ only depends on the smallest and largest eigenvalue of $\mathbf{Q}$. Once γ is defined, the interpolation formula can be rewritten as

$$\tilde{\mathrm{x}} = \mathrm{x} = \mathbf{V}_{(:,\mathcal{K})}\big(\gamma \sum_{n=0}^{\infty}(\mathbf{I} - \gamma\mathbf{Q})^n\big)(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}}. \tag{3.12}$$

Limiting the summation index to the value p and defining the vector $\mathbf{y}_p$ as

$$\mathbf{y}_p = \gamma \sum_{n=0}^{p}(\mathbf{I} - \gamma\mathbf{Q})^n(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}}. \tag{3.13}$$

leads to the following recursive formulation:

For $p = 0$, (3.13) equals

$$\mathbf{y}_0 = \gamma(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}}, \tag{3.14}$$

and $\mathbf{y}_{p+1}$ can be expanded as follows

$$\begin{aligned}
\mathbf{y}_{p+1} &= \gamma \sum_{n=0}^{p+1}(\mathbf{I} - \gamma\mathbf{Q})^n(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} \\
&= \gamma\big(\mathbf{I} + \sum_{n=1}^{p+1}(\mathbf{I} - \gamma\mathbf{Q})^n\big)(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} \\
&= \gamma\big(\mathbf{I} + \sum_{n=0}^{p}(\mathbf{I} - \gamma\mathbf{Q})^{n+1}\big)(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} \\
&= \gamma\big(\mathbf{I} + (\mathbf{I} - \gamma\mathbf{Q})\sum_{n=0}^{p}(\mathbf{I} - \gamma\mathbf{Q})^n\big)(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} \\
&= \gamma(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} + (\mathbf{I} - \gamma\mathbf{Q})\ \gamma\ \sum_{n=0}^{p}(\mathbf{I} - \gamma\mathbf{Q})^n(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} \\
&= \gamma(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}} + (\mathbf{I} - \gamma\mathbf{Q})\mathbf{y}_p \ = \ \mathbf{y}_0 + \mathbf{T}_\gamma \mathbf{y}_p.
\end{aligned} \tag{3.15}$$

Summary of the iterative recovery method

The key facts of the presented iterative recovery algorithm for sampled graph signals are briefly summarized in the following:

Inputs of the iterative algorithm:

– eigenbasis $\mathbf{V}$ of graph shift

– sampling support set $\mathcal{M}$

– frequency support set $\mathcal{K}$

– matrix $\mathbf{Q} = (\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \ (\mathbf{V}_{(\mathcal{M},\mathcal{K})})$

– factor $\gamma = 2/(\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)})$

– sampled graph signal $\mathbf{x}_{\mathcal{M}}$.

Iterative recovery algorithm:

– Define linear operator $\mathbf{T}_\gamma = \mathbf{I} - \gamma\mathbf{Q}$

– Initialize: $\mathbf{y}_0 = \gamma(\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_{\mathcal{M}}$

– Iterate for $p = 1,\ 2,\ \ldots\ p_{\max} - 1$: $\mathbf{y}_{p+1} = \mathbf{y}_0 + \mathbf{T}_\gamma \mathbf{y}_p$.

Output of the iterative algorithm:

– Output recovered graph signal $\tilde{\mathrm{x}} = \mathbf{V}_{(:,\mathcal{K})}\mathbf{y}_{p+1}$.

Remarks:

The complexity of the iterative recovery algorithm is dominated by matrix-vector multiplication in (3.15). Since the dimension of the vector $\mathbf{y}_p$ equals K, the complexity order of the recursive iteration is $\mathcal{O}(p_{\max}K^2)$. (An analytical bound of the maximal number of iterations $p_{\max}$ will be derived in the next section.)

The matrix inversion has a complexity order of $\mathcal{O}(K^3)$, therefore the complexity of iterative approach is lower, as long as $p_{\max} < K$.

3.2 Analytic Error-Bound of the Recovery Algorithm

Using the same definitions, Neuman-series and framework as in the previous section, we will derive an analytic error bound of the iterative recovery algorithm in the following. The derivations in this section are adopted from [1].

$$\begin{aligned} \mathbf{y} = \mathbf{y}_\infty = \underbrace{\gamma \sum_{n=0}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n}_{=\mathbf{Q}^{-1}} (\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_\mathcal{M} = \mathbf{Q}^{-1} \frac{1}{\gamma} \mathbf{y}_0 \\ \implies \mathbf{y}_0 = \gamma (\mathbf{V}_{(\mathcal{M},\mathcal{K})})^H \mathrm{x}_\mathcal{M} = \gamma\, \mathbf{Q}\, \mathbf{y}. \end{aligned} \tag{3.16}$$

Consequently, the error can be expressed as

$$\left\| \mathbf{y} - \mathbf{y}_p \right\|_2 = \left\| \sum_{n=p+1}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n \, \gamma\, \mathbf{Q}\mathbf{y} \right\|_2 . \tag{3.17}$$

Using the matrix inequality $\|\mathbf{AB}\|_2 \leq \|\mathbf{A}\|_2 \|\mathbf{B}\|_2$, (3.17) can be written as

$$\left\| \mathbf{y} - \mathbf{y}_p \right\|_2 \leq \left\| \sum_{n=p+1}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n \, \gamma\, \mathbf{Q} \right\|_2 \left\| \mathbf{y} \right\|_2 . \tag{3.18}$$

Applying this inequality again to the term $\left\| \sum_{n=p+1}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n \, \gamma\, \mathbf{Q} \right\|_2$ results in

$$\begin{aligned} \left\| \sum_{n=p+1}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n \, \gamma\, \mathbf{Q} \right\|_2 = \left\| \sum_{n=p+1}^{\infty} (\mathbf{T}_{\boldsymbol{\gamma}})^n \, \gamma\, \mathbf{Q} \right\|_2 \\ &\leq \sum_{n=p+1}^{\infty} \|(\mathbf{T}_{\boldsymbol{\gamma}})^n\|_2 \|\gamma\, \mathbf{Q}\|_2 \\ &\leq \sum_{n=p+1}^{\infty} \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2^n \, \|\gamma\, \mathbf{Q}\|_2 \\ &= \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2^{p+1} \underbrace{\Big(\sum_{n=0}^{\infty} \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2^n\Big)}_{\text{geometric series}} \|\gamma\mathbf{Q}\|_2 . \end{aligned} \tag{3.19}$$

Since in (3.9) the factor γ is chosen such that $\|\mathbf{T}_\gamma\|_2 < 1$, the geometry series will converge. Hence,

$$\left\| \sum_{n=p+1}^{\infty} (\mathbf{I} - \gamma\mathbf{Q})^n \;\; \gamma\, \mathbf{Q} \right\|_2 \leq \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2^{p+1} \Big(\frac{1}{1 - \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2}\Big) \|\gamma\mathbf{Q}\|_2 . \tag{3.20}$$

The factor $\|\gamma\mathbf{Q}\|_2$ can be bounded as

$$\|\gamma\mathbf{Q}\|_2 = \|\mathbf{I} - (\mathbf{I} - \gamma\mathbf{Q})\|_2 \leq 1 + \|\mathbf{T}_\gamma\|_2 < 1. \tag{3.21}$$

Combining (3.18), (3.20) and (3.21), the error can be bounded as

$$\frac{\left\|\mathbf{y} - \mathbf{y}_p\right\|_2}{\|\mathbf{y}\|_2} \leq \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2^{p+1} \frac{1 + \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2}{1 - \|\mathbf{T}_{\boldsymbol{\gamma}}\|_2}. \tag{3.22}$$

With $\gamma = 2/(\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)})$ from (3.11), the norm $\|\mathbf{T}_\gamma\|_2$ equals

$$\|\mathbf{T}_\gamma\|_2 = \max_{j=1,\ldots,K} \left(|1 - \gamma\lambda_j^{(Q)}|\right) = \left|1 - \frac{2}{\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)}} \lambda_{\max}^{(Q)}\right| = \frac{\lambda_{\min}^{(Q)} - \lambda_{\max}^{(Q)}}{\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)}}. \tag{3.23}$$

Setting the target accuracy $\varepsilon = \left\|\mathbf{y} - \mathbf{y}_p\right\|_2 / \|\mathbf{y}\|_2$ and using (3.22) leads to

$$\varepsilon \leq \left(\frac{\lambda_{\max}^{(Q)} - \lambda_{\min}^{(Q)}}{\lambda_{\min}^{(Q)} + \lambda_{\max}^{(Q)}}\right)^{p_{\max}+1} \frac{\lambda_{\max}^{(Q)}}{\lambda_{\min}^{(Q)}} = \left(\frac{\kappa(\mathbf{Q}) - 1}{\kappa(\mathbf{Q}) + 1}\right)^{p_{\max}+1} \kappa(\mathbf{Q}), \tag{3.24}$$

with $\kappa(\mathbf{Q}) = \lambda_{\max}^{(Q)}/\lambda_{\min}^{(Q)}$ being the condition number of the matrix $\mathbf{Q}$.

The theoretical bound for the number of iterations required to reach the relative accuracy ε is given by

$$p_{\max} \geq \frac{\log_{10}(\kappa(\mathbf{Q})/\varepsilon)}{\log_{10}\left(\frac{\kappa(\mathbf{Q})-1}{\kappa(\mathbf{Q})+1}\right)} - 1. \tag{3.25}$$

So far, a result for the accuracy of the recovered non-zero Fourier coefficient $\mathbf{y}_p$ was obtained, but not for the recovered signal $\mathbf{x}$. Using the relation $\mathbf{x}_p = \mathbf{V}_{(:,\mathcal{K})}\mathbf{y}_p$, an estimate of the graph signal $\mathbf{x}$ can be obtained.

Therefore, the error of the graph signal can be computed as

$$\|\mathbf{x} - \mathbf{x}_{\boldsymbol{p}}\|_2 = \left\|\mathbf{V}_{(:,\mathcal{K})}(\mathbf{y} - \mathbf{y}_p)\right\|_2, \tag{3.26}$$

and the relative error is given by

$$\frac{\|\mathbf{x} - \mathbf{x}_{\boldsymbol{p}}\|_2}{\|\mathbf{x}\|_2} = \frac{\left\|\mathbf{V}_{(:,\mathcal{K})}(\mathbf{y} - \mathbf{y}_p)\right\|_2}{\left\|\mathbf{V}_{(:,\mathcal{K})}\mathbf{y}\right\|_2}. \tag{3.27}$$

The error of $\mathbf{x}_p$ can be bounded as follows (details can be found in [1]):

$$\frac{\|\mathbf{x}-\mathbf{x}_{\boldsymbol{p}}\|_2}{\|\mathbf{x}\|_2} \leq \kappa(\mathbf{V}_{(:,\mathcal{K})}) \frac{\left\|(\mathbf{y}-\mathbf{y}_p)\right\|_2}{\|\mathbf{y}\|_2}. \tag{3.28}$$

For a given frequency support set $\mathcal{K}$ the condition number $\kappa(\mathbf{V}_{(:,\mathcal{K})})$ needs to be computed. In the case of a symmetric graph shift, the eigenvalue decomposition leads to orthonormal eigenvectors and the condition number $\kappa(\mathbf{V}_{(:,\mathcal{K})}) = 1$ for any choice of the frequency support set $\mathcal{K}$. Therefore, the relative errors in (3.22) and (3.27) are identical.

Chapter 4

Numerical Experiments

In this chapter, we are going to investigate the performance of the iterative algorithm presented previously in this thesis. For this purpose, multiple simulations were carried out with different random graph shift matrices. Moreover, a modification of the iterative algorithm that allows further reduction of complexity at runtime is presented.

In the first part, parameters of weight matrices chosen for the simulations are introduced. Different dimensions, distributions and sparsity factors are considered. The second part of this chapter presents the detailed simulation outcomes and shows the impact of each considered parameter. Moreover, a solution to the problem of recomputing the factor γ for every realization is presented. The simulations are repeated with an estimated factor γ, in order to conclude with a performance analysis of the iterative algorithm.

4.1 Simulation parameters

The random graph shift matrices $\mathbf{W}$ of size $N \times N$ are chosen to be symmetric and full rank. Three distributions of the elements $w_{i,j}$ of the matrix $\mathbf{W}$ are considered in the simulations:

- Uniform: $w_{i,j} \sim \mathcal{U}(0,1)$ for $i,j = 1,2,\dots N$,
- Gaussian: $w_{i,j} \sim \mathcal{N}(0,1)$ for $i,j = 1,2,\dots N$,
- Discrete: $w_{i,j}$ chosen randomly with equal probability from the set $\{-1,1\}$.

Since in practical situations each node of a graph is only connected to few other nodes, a fixed portion of the entries of the weight matrix needs to be set to zero. To do so, a sparsity factor δ is introduced that defines the portion of elements of $\mathbf{W}$ that is set to zero, whereby the elements set to zero are chosen randomly from all N^2 elements with equal probability. For example, delta-sparsity 0.90 means that 90% of the N^2 matrix entries $w_{i,j}$ are zero. The following values for the sparsity factor are considered (Table 4.1):

$\delta = 0.75$
$\delta = 0.80$
$\delta = 0.85$
$\delta = 0.90$

Table 4.1: Sparsity factors δ considered in the simulations

Furthermore, the graph is assumed not to have any self-loops, thus the entries on the main diagonal of the weight matrix are set to zero.

Finally, the reversed Cuthill–Mckee algorithm is used to permute the sparse matrix $\mathbf{W}$ into a band matrix form and to bring the non-zero entries closer to the main diagonal [24].

As an illustration, three examples of graph shift matrices with $N = 7$ and $\delta = 0.75$ are shown in the following:

$$\mathbf{W}_{\boldsymbol{uniform}} = \begin{pmatrix} 0 & 0.0375 & 0 & 0 & 0 & 0 & 0 \\ 0.0375 & 0 & 0.3296 & 0 & 0 & 0 & 0 \\ 0 & 0.3296 & 0 & 0.4786 & 0.3575 & 0 & 0 \\ 0 & 0 & 0.4786 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.3575 & 0 & 0 & 0 & 0.1364 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.4065 \\ 0 & 0 & 0 & 0 & 0.1364 & 0.4065 & 0 \end{pmatrix},$$

$$\mathbf{W}_{\boldsymbol{gaussian}} = \begin{pmatrix} 0 & -0.4834 & 0 & 0 & 0 & 0 & 0 \\ -0.4834 & 0 & 1.0457 & 0 & 0 & 0 & 0 \\ 0 & 1.0457 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.498 & 0 & 0 \\ 0 & 0 & 0 & 0.498 & 0 & 0.0616 & 0 \\ 0 & 0 & 0 & 0 & 0.616 & 0 & 0.6635 \\ 0 & 0 & 0 & 0 & 0 & 0.6635 & 0 \end{pmatrix},$$

$$\mathbf{W}_{\boldsymbol{discrete}} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Moreover, different proportions between the signal dimension N, the number of measured components M, and the number of non-zero Fourier coefficients K are considered in the simulations. The values of N, M and K are related through the factors α and β according to

$$M = N\alpha \tag{4.1}$$

$$K = N\beta, \tag{4.2}$$

with factors α and β as indicated in Table 4.2.

	pair 1	pair 2	pair 3	pair 4
α	0.5	0.5	0.2	0.1
β	0.4	0.3	0.12	0.75

Table 4.2: Selected values for α and β

Restricting the signal dimension to three possible values $N = 200$, $N = 1000$ and $N = 5000$, leads to the sets of (N, M, K) listed in Table 4.3.

For each set (N,M,K), the simulations were repeated 1000 times with statistically independent realizations of the graph shift matrix, the selected Fourier coefficients and the selected measurement support. Due to the long simulation runtime for $N = 5000$, only the simulations for the special case of uniformly distributed elements of the weight matrix, $\delta = 0.9$ and $\varepsilon = 0.01$ were carried out.

4.2 Simulation outcome

The relative reconstruction accuracy ε is defined according to section 3.2 as

$$\varepsilon = \frac{||\mathbf{x} - \mathbf{x}_{\boldsymbol{p}}||_2}{||\mathbf{x}||_2}. \tag{4.3}$$

In the following, p denotes the number of iterations needed to reach a given accuracy ε and $\bar{p}$ denotes the number of iterations p averaged over 1000 realizations.

	N	M	K
set 1	200	100	80
set 2	200	100	60
set 3	200	40	24
set 4	200	20	15
set 5	1000	500	400
set 6	1000	500	300
set 7	1000	200	120
set 8	1000	100	75
set 9	5000	2500	2000
set 10	5000	2500	1500
set 11	5000	1000	600
set 12	5000	500	375

Table 4.3: Selected sets (N, M, K)

The complexity of the iterative algorithm was shown to be proportional to $\mathcal{O}(\bar{p}K^2)$ (Chapter 3) whereas the complexity of a matrix inversion is proportional to $\mathcal{O}(K^3)$. Therefore, as long as p is considerably smaller than K, the iterative method has significantly lower complexity than the matrix inversion.

Fig. 4.1 shows the average number of iterations $\bar{p}$ needed to reach the relative accuracy $\varepsilon = 0.01$ for different weight matrices (uniform,Gaussian, discrete) and different sparsity factors. The results show, that for larger problem dimensions, i.e. the set $(N = 1000, M = 500, K = 300)$, $\bar{p}$ is considerably smaller than K and smaller standard deviation of p was recorded compared to the case of smaller signal dimensions.

Moreover, it can be observed that the mean number of iterations strongly depends on the dimensions N, M, K and do not considerably change when the modifying sparsity factors or the distribution of the elements.

Another observed result is that scaling proportionally the set (N, M, K) does not have strong impact on the value $\bar{p}$. For example, Fig 4.1 shows the impact of scaling by a factor 5. It can be noticed that the set $(N = 1000, M = 500, K = 300)$, which is the scaled version of the set $(N = 200, M = 100, K = 60)$, shows an very similar value $\bar{p}$. The same behaviour can be observed for the two other sets shown on the right hand side of Fig 4.1.

Since scaling proportionally (N, M, K) does not affect the necessary number of iterations, it can be concluded that special relations between the number of Fourier-coefficients K and the measurement support set M rule the number of iterations $\bar{p}$. It is the relation between α and β that mainly determines $\bar{p}$. Therefore, for a small signal di-

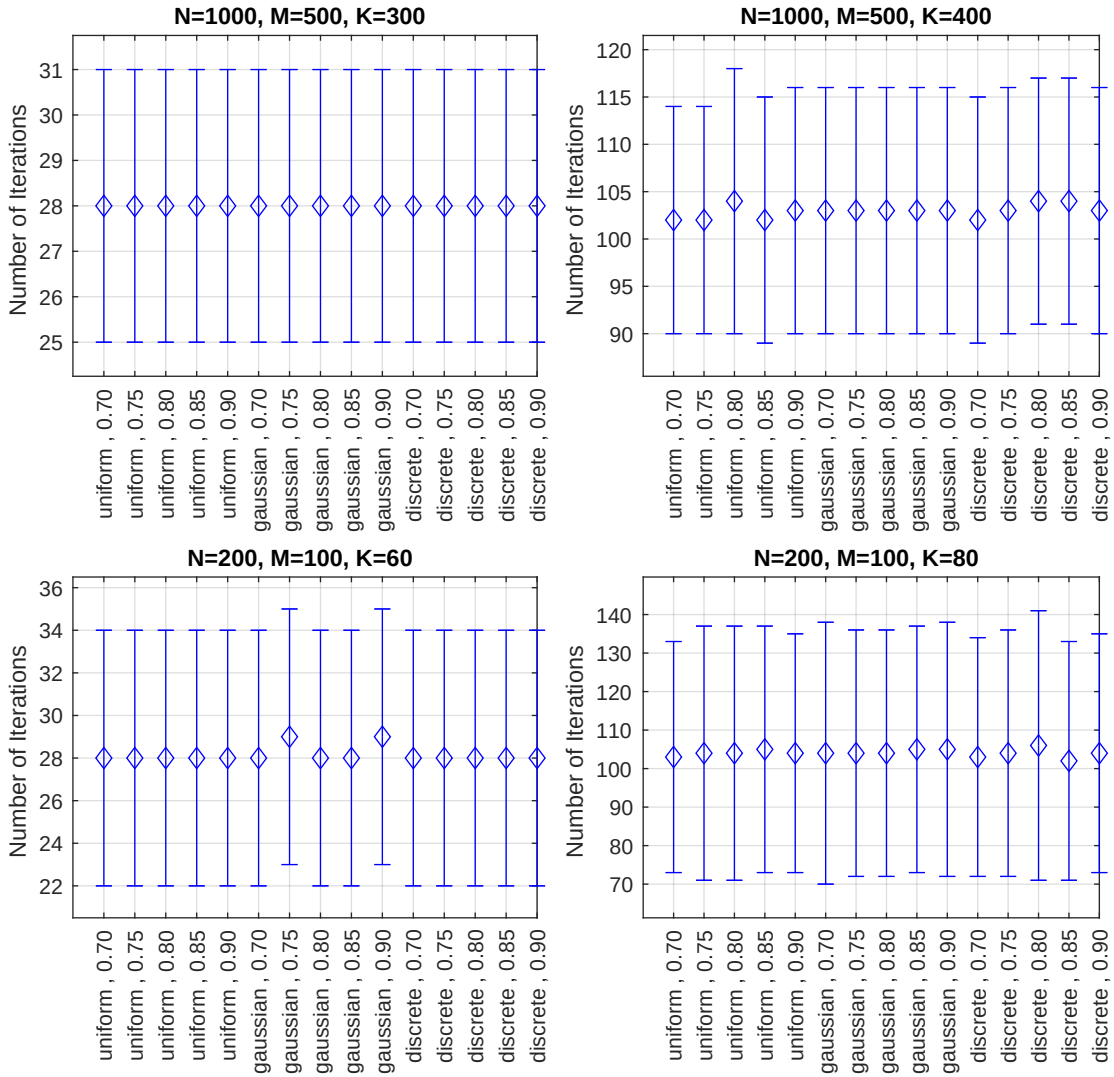


Figure 4.1: Iterative recovery algorithm, number of iterations needed to reach an accuracy of $\varepsilon = 0.01$ (mean indicated by diamond marker, standard deviation by error bar).

mension, i.e. the set $(N = 200, M = 100, K = 80)$, the value of $\bar{p}$ even exceeds K, which means that the complexity of the iterative method becomes larger than the complexity of a matrix inversion.

From an application point of view, a big advantage of the iterative recovery algorithm compared to a matrix inversion is that accepting lower accuracy allows a significant complexity reduction. The number of iterations $\bar{p}$ required to reach different accuracies ($\varepsilon = 0.01$, $\varepsilon = 0.05$ and $\varepsilon = 0.1$) are compared in Table 4.4. Moreover, it can be noted,

that in this example all distributions behave similarly.

	$\varepsilon = 0.01$			$\varepsilon = 0.05$			$\varepsilon = 0.1$		
	$\bar{p}_{max}$	$\bar{p}$	σ_p	$\bar{p}_{max}$	$\bar{p}$	σ_p	$\bar{p}_{max}$	$\bar{p}$	σ_p
uniform	83	28	3	66	14	2	58	9	1
Gaussian	83	28	3	66	14	2	58	9	1
dicrete	83	28	3	66	14	2	58	9	1

Table 4.4: Comparison of number of iterations for different desired accuracies, $\delta = 0.9$ and $N = 1000$, $M = 500$, $K = 300$.

Furthermore, the impact of increasing the signal dimension while maintaining the same values of α, β on the statistical distribution of p is analyzed. The random variable p can only take discrete values[1]. Its probability mass function (pmf) is shown in Fig. 4.2 for uniformly distributed weight matrices. Equivalent figures for other weight matrix distributions (Gaussian, discrete) can be found in Appendix B, Fig. B.1 and Fig. B.2. It can be recognized that for each figure, the location of the peak is almost not affected by increasing the signal dimension. Furthermore, when increasing the signal dimension, the standard deviation of p becomes less, leading to a better predictable number of iterations p needed to reach a desired accuracy.

In principle, the factor γ needs to be recomputed for every realization of the support sets $\mathcal{M}$ and $\mathcal{K}$. That means, for every realization, the largest and smallest eigenvalue of the matrix $\mathbf{Q}$ have to be computed. The statistical distribution of γ is shown in Fig. 4.3 for weight matrices with entries taken from a uniform distribution. Equivalent figures for the two other considered weight matrix distributions can be found in Appendix B, Fig B.3 and Fig B.4. Again, it can be observed that increasing the signal dimension within the same pair α, β does not strongly affects the location of the peak of the pdf of γ, whereas the standard deviation is becoming considerably less when increasing the signal dimension.

The small quantity of standard deviation of γ, especially for large dimension motivated us to reconsider the problem of recomputing the factor γ for every realization. Determining the factor γ without calculation of the largest and smallest eigenvalues of matrix $\mathbf{Q}$ can lead to the reduction of complexity of the iterative method at runtime.

[1] In the following, random variables are denoted by upright characters whereas deterministic variables or realizations of random variables are denoted by italic characters.

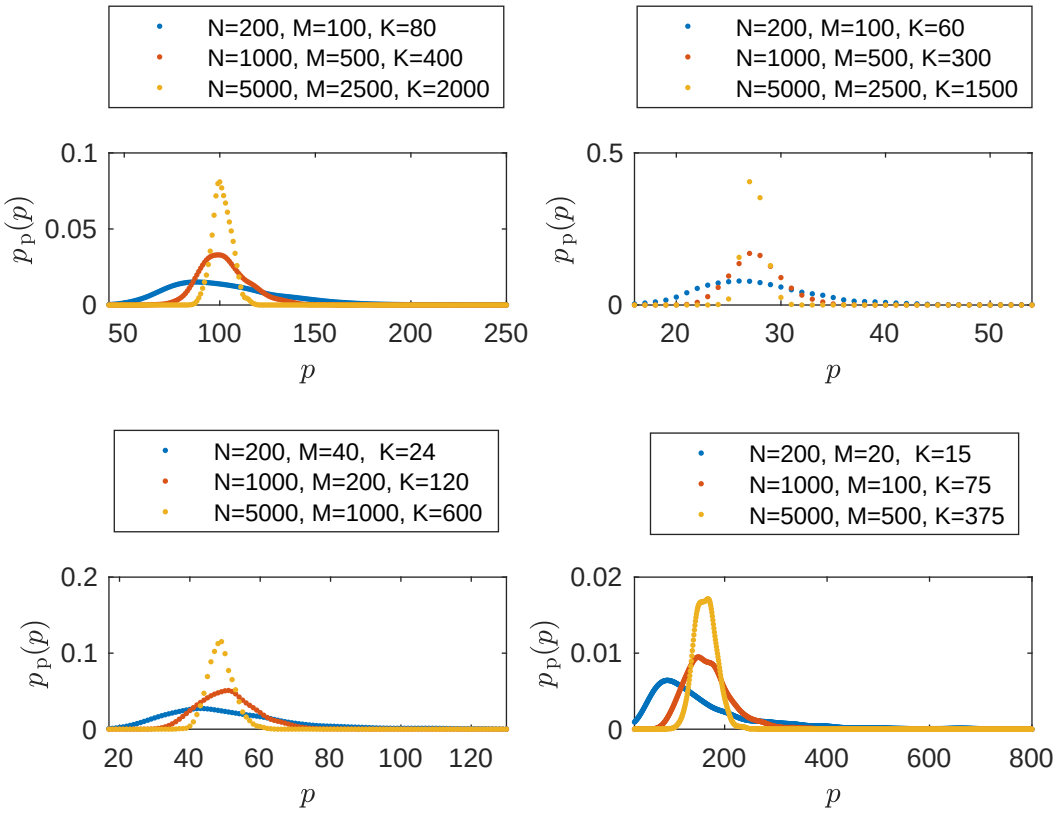


Figure 4.2: pmf of p, $\delta = 0.9$, $\varepsilon = 0.01$, uniform distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Hence, other simulations were carried out with fixed values of the factor γ. The factor $\tilde{\gamma}$ is chosen such, that according to the Tschebycheff inequality, $\tilde{\gamma}$ will be smaller than γ with high probability for every realization,

$$\tilde{\gamma} = \bar{\gamma} - 5\sigma_{\gamma}.$$

In order to show the impact of fixing the factor γ, the pmf of p, obtained by repeating the simulations using $\tilde{\gamma}$ instead of $\bar{\gamma}$, is shown in Fig. 4.4. Similar figures for weight matrices with other distributions (Gaussian, discrete) can be found in Appendix B, Fig. B.5 and Fig. B.6.

Additionally, Table 4.5 provides a comparison of the simulation outcome using the fixed factor $\tilde{\gamma}$ with the previous simulations. The last two columns of Table 4.5 show that for pair 2 ($\alpha = 0.5$ and $\beta = 0.3$), the results are almost identical. Pair 4 ($\alpha = 0.1$ and $\beta = 0.75$) with $N = 200$ performs worst: For some (very few) realizations, the iterative algorithm using fixed $\tilde{\gamma}$ does not converge fast enough ($p > p_{max}$ was observed)

due to the large standard deviation σ_γ. A summary of the simulation outcomes can be found in Appendix A.

Finally, the simulations showed that if the statistical properties of the weighted adjacency matrix of the graph $\mathbf{W}$ are known and M, K are chosen properly, the factor γ can be determined beforehand, without any extra complexity at run-time to compute eigenvalues of the matrix $\mathbf{Q}$.

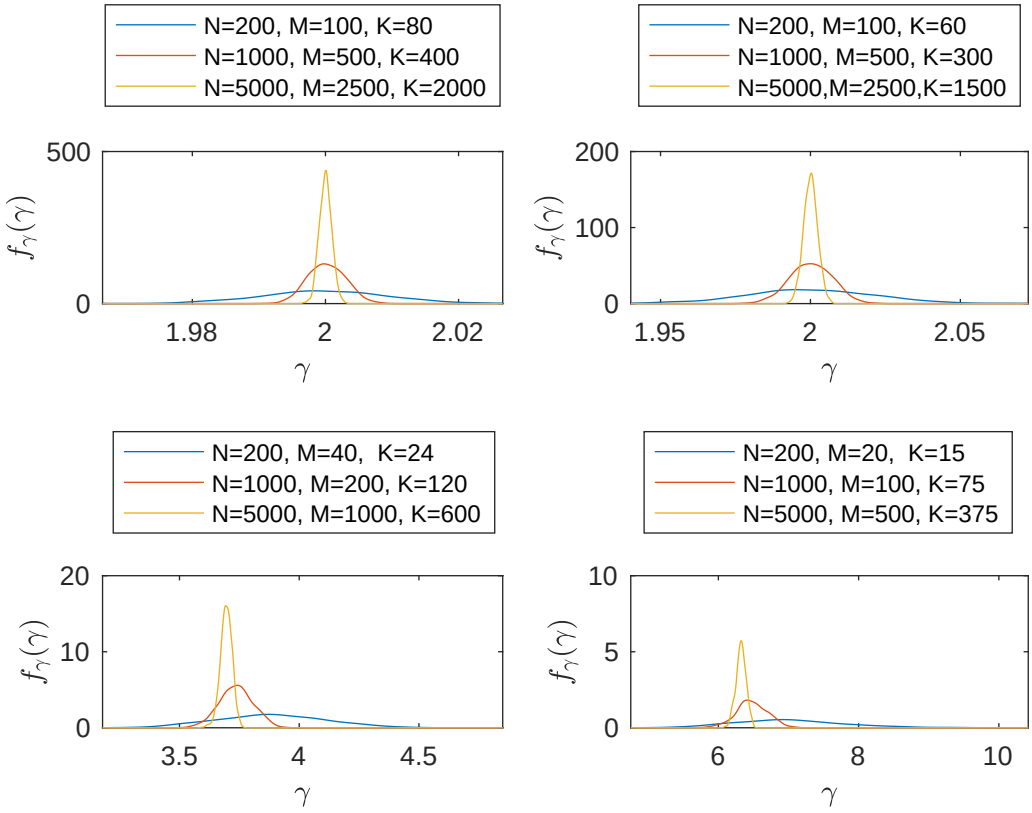


Figure 4.3: pdf of γ, $\delta = 0.9$, $\varepsilon = 0.01$, uniform distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

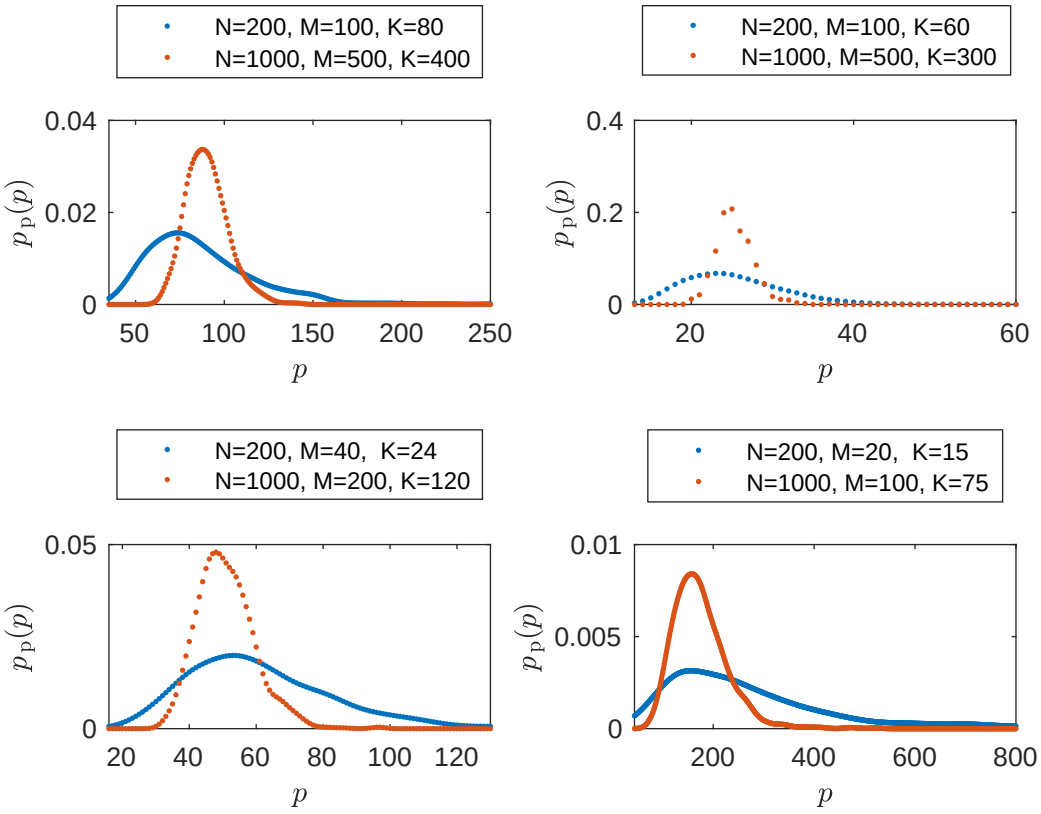


Figure 4.4: pmf of p, fixed $\tilde{\gamma} = \bar{\gamma} - 5\sigma_\gamma$, $\delta = 0.9$, $\varepsilon = 0.01$, uniform distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

set	pair α, β	$\bar{\gamma}$	σ_γ	$\tilde{\gamma}$	$\bar{p}(\gamma)$	$\bar{p}(\tilde{\gamma})$
$N = 200$, $M = 100$, $K = 80$	pair 1	2.0	0.0094	1.953	104	88
$N = 200$, $M = 100$, $K = 60$	pair 2	2.0	0.0215	1.893	28	26
$N = 200$, $M = 40$, $K = 24$	pair 3	3.9	0.2262	2.769	52	62
$N = 200$, $M = 20$, $K = 15$	pair 4	7	0.7951	3.025	163	318
$N = 1000$, $M = 500$, $K = 400$	pair 1	2.0	0.0028	1.986	103	91
$N = 1000$, $M = 500$, $K = 300$	pair 2	2.0	0.0070	1.965	28	26
$N = 1000$, $M = 200$, $K = 120$	pair 3	3.7	0.0699	3.351	51	51
$N = 1000$, $M = 100$, $K = 75$	pair 4	6.5	0.2258	5.371	169	175

Table 4.5: Performance comparison, iterative algorithm, uniformly distributed entries of weight matrix.

Chapter 5

Conclusion

In this thesis, signal processing on graphs, which is a vast field in the analysis of large-scale data is investigated. Graph signal processing is merging the concepts of classical signal processing with graph theory and has laid a strong foothold in various application domains.

We started by introducing some fundamentals of graph theory. Afterwards, we reviewed some concepts of classical signal processing and extended them to graph signal processing, such as time-shift to graph-shift, filters to graph-filters, and Fourier transform to graph Fourier transform. Moreover, we introduced the notion of graph sampling and presented the graph sampling theorem as well as the concepts of random and experimentally designed sampling.

The following part of this thesis dealt with the recovery of sampled graph signals. We introduced the perfect recovery theorem, that, together with Neumann series, served as a basis to derive an iterative recovery algorithm for sampled graph signals. The presented iterative recovery algorithm is universally applicable to general weight matrices and has an analytic upper error bound that allows to predict the number of iterations as a function of the desired reconstruction accuracy.

In order to investigate the performance of the iterative algorithm, multiple simulations with different random graph weight matrices were carried out in the scope of this thesis. The entries of the weight matrices considered in the simulations were taken from one of the following random distributions: uniform, Gaussian or discrete distribution. The simulation results showed that the choice of the random distribution of the entries of the weight matrix does not significantly affect the performance of the iterative algorithm.

Moreover, the impact of the weight matrix dimension, the size of the frequency support set and the size of the sampling support set was investigated. The simulations showed better performance in terms of complexity reduction when a proper proportion between the signal dimension, the size of the frequency support set and the number measured components was chosen.

Another significant reduction of complexity was achieved when accepting lower accuracies. This means applications which require less accuracy can benefit from a higher computation performance since less iterations are needed to reach a given reconstruction accuracy.

Finally, the simulations showed that if the signal dimension N, the size of the frequency support set K and the number measured components M are chosen properly, the factor γ can be estimated beforehand, without any extra complexity at run-time to compute eigenvalues of the matrix $\mathbf{Q}$.

Since the simulations in the scope of this thesis were restricted to three different random distributions (uniform, Gaussian and discrete), further research on this topic could focus on investigating the behaviour of the iterative reconstruction algorithm and predicting the factor γ when using real-world data.

Appendices

Appendix A

		pair 1	pair 1	pair 1	pair 2	pair 2	pair 2	pair 3	pair 3	pair 3	pair 4	pair 4	pair 4
		$N = 200$ $M = 100$ $K = 80$	$N = 1000$ $M = 500$ $K = 400$	$N = 5000$ $M = 2500$ $K = 2000$	$N = 200$ $M = 100$ $K = 60$	$N = 1000$ $M = 500$ $K = 300$	$N = 5000$ $M = 2500$ $K = 1500$	$N = 200$ $M = 40$ $K = 24$	$N = 1000$ $M = 200$ $K = 120$	$N = 5000$ $M = 1000$ $K = 600$	$N = 200$ $M = 20$ $K = 15$	$N = 1000$ $M = 100$ $K = 75$	$N = 5000$ $M = 500$ $K = 375$
Uniform	$\bar{p}$	104	103	102	28	28	28	52	51	49	163	169	163
	$\bar{p}_{max}$	352	411	435	74	83	87	133	162	177	461	598	713
	σ_p	31	13	6	6	3	1	19	8	4	135	47	22
	$\bar{\gamma}$	2	2	2	2	2	2	3,9	3,7	3,7	7	6,5	6,3
	σ_γ	0,0094	0,0028	0,0009	0,0215	0,0070	0,0023	0,2262	0,0699	0,0242	0,7951	0,2258	0,0713
Gaussian	$\bar{p}$	105	103	–	29	28	–	52	52	–	164	171	–
	$\bar{p}_{max}$	354	408	–	76	83	–	133	163	–	460	618	–
	σ_p	33	13	–	6	3	–	19	9	–	147	53	–
	$\bar{\gamma}$	2	2	–	2	2	–	3,9	3,7	–	7	6,5	–
	σ_γ	0 0091	0,0028	–	0,0224	0,0073	–	0,2277	0,0696	–	0,7817	– 0,2175	
Discrete	$\bar{p}$	104	103	–	28	28	–	50	51	–	158	166	–
	$\bar{p}_{max}$	352	410	–	74	83	–	129	162	–	439	600	–
	σ_p	31	13	–	6	3	–	19	9	–	129	47	–
	$\bar{\gamma}$	2	2	–	2	2	–	3,9	3,7	–	7,1	6,5	–
	σ_γ	0,0093	0,0029	–	0,0217	0,007	–	0,2287	0,0692	–	0,7453	0,2159	–

Table A.1: Simulation outcome, 1000 realizations, for different weight matrix distributions and dimensions, $\delta = 0.9$, $\varepsilon = 0.01$
Not all simulations with $N = 5000$ were carried out due to the long simulation runtime.

		pair 1	pair 1	pair 2	pair 2	pair 3	pair 3	pair 4	pair 4
		$N = 200$ $M = 100$ $K = 80$	$N = 1000$ $M = 500$ $K = 400$	$N = 200$ $M = 100$ $K = 60$	$N = 1000$ $M = 500$ $K = 300$	$N = 200$ $M = 40$ $K = 24$	$N = 1000$ $M = 200$ $K = 120$	$N = 200$ $M = 20$ $K = 15$	$N = 1000$ $M = 100$ $K = 75$
Uniform	$\bar{p}$	88	91	26	26	62	51	318(*)	175
	$\bar{p}_{max}$	352	411	74	84	132	162	458	604
	σ_p	33	13	6	3	24	9	301	54
	$\tilde{\gamma}$	$1,953$	$1,985$	$1,892$	$1,965$	$2,751$	$3,387$	$3,037$	$5,347$
Gaussian	$\bar{p}$	89	92	26	26	64	52	323(*)	174
	$\bar{p}_{max}$	350	410	74	83	134	163	494	602
	σ_p	31	13	6	3	25	9	481	56
	$\tilde{\gamma}$	$1,954$	$1,986$	$1,889$	$1,964$	$2,722$	$3,392$	$3,080$	$5,384$
Discrete	$\bar{p}$	89	91	26	26	62	51	283(*)	175
	$\bar{p}_{max}$	354	411	74	83	128	162	452	607
	σ_p	31	12	6	3	25	9	261	54
	$\tilde{\gamma}$	$1,954$	$1,986$	$1,892$	$1,965$	$2,744$	$3,393$	$3,324$	$5,404$

Table A.2: Simulation outcome fixed gamma, 1000 realizations, $\delta = 0.9$, $\varepsilon = 0.01$.
(*) indicates that $p > p_{max}$ at least for one realization.

Appendix B

Gaussian distribution:

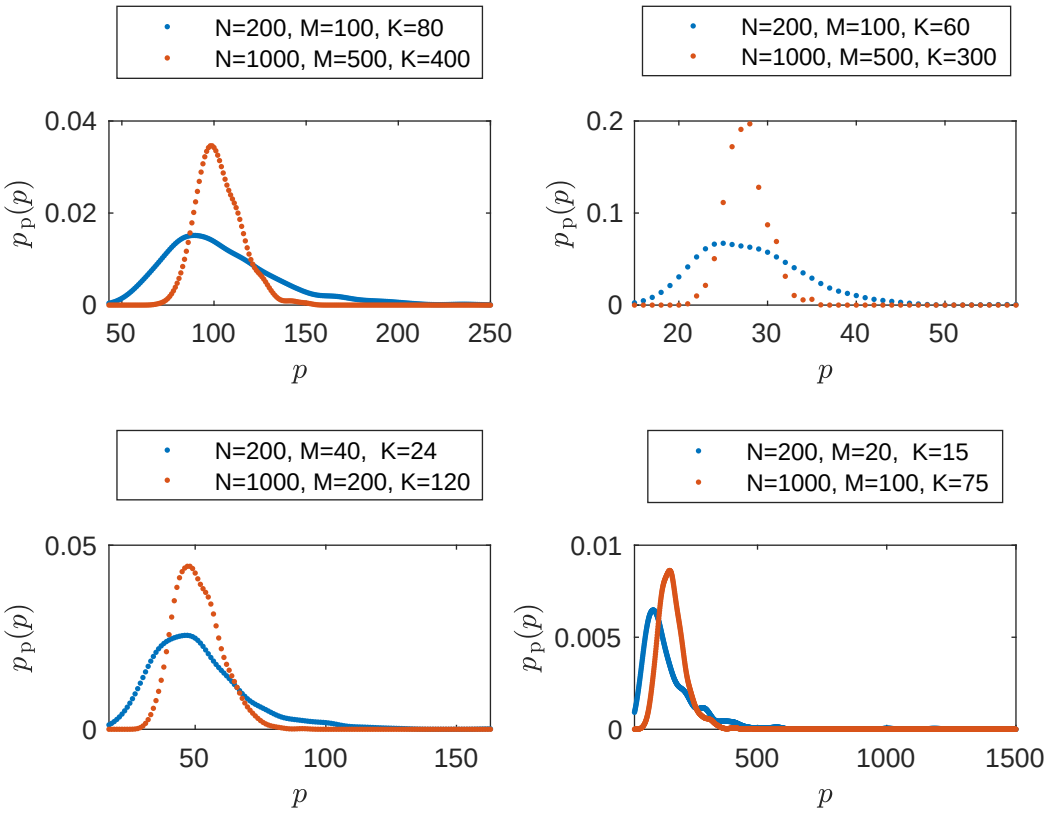


Figure B.1: pmf of p, $\delta = 0.9$, $\varepsilon = 0.01$, gaussian distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Discrete distribution:

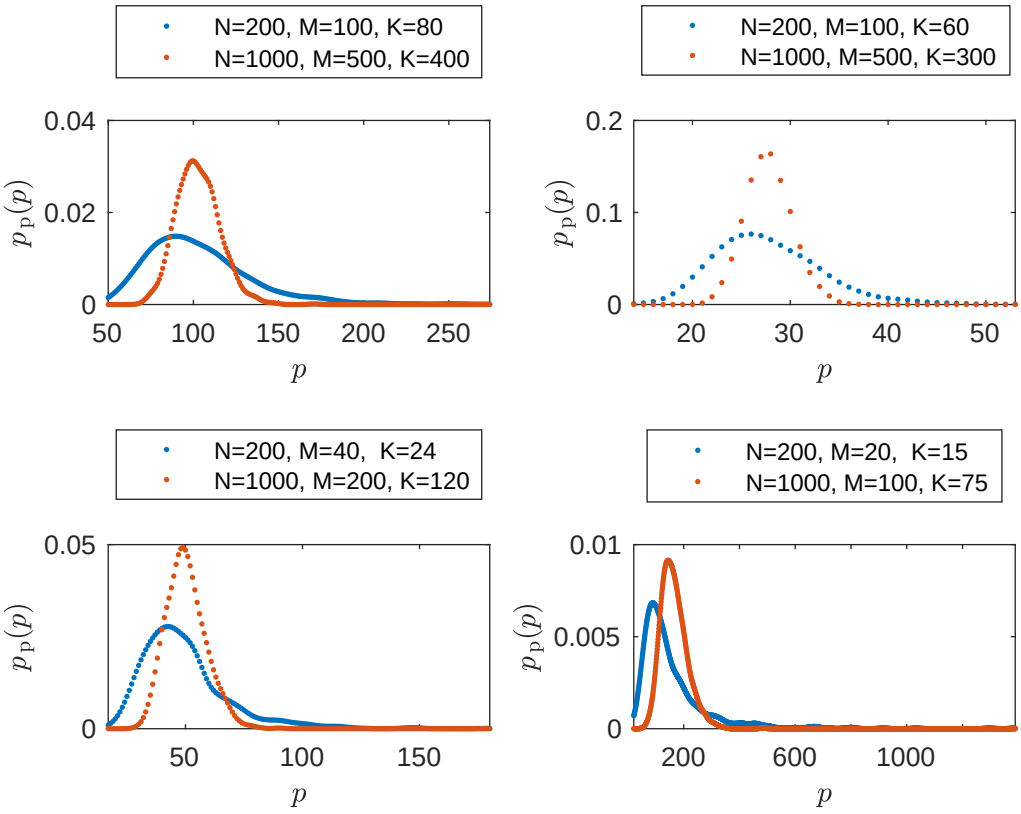


Figure B.2: pmf of p, $\delta = 0.9$, $\varepsilon = 0.01$, discrete distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Gaussian distribution:

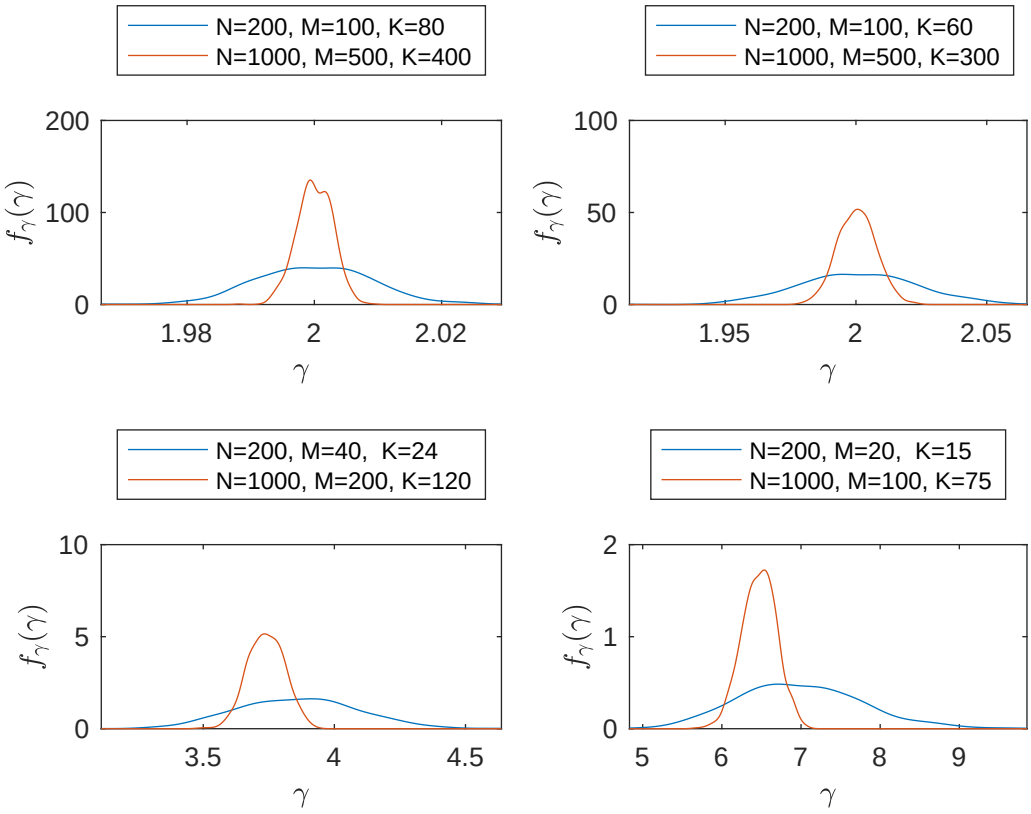


Figure B.3: pdf of γ, $\delta = 0.9$, $\varepsilon = 0.01$, gaussian distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Discrete distribution:

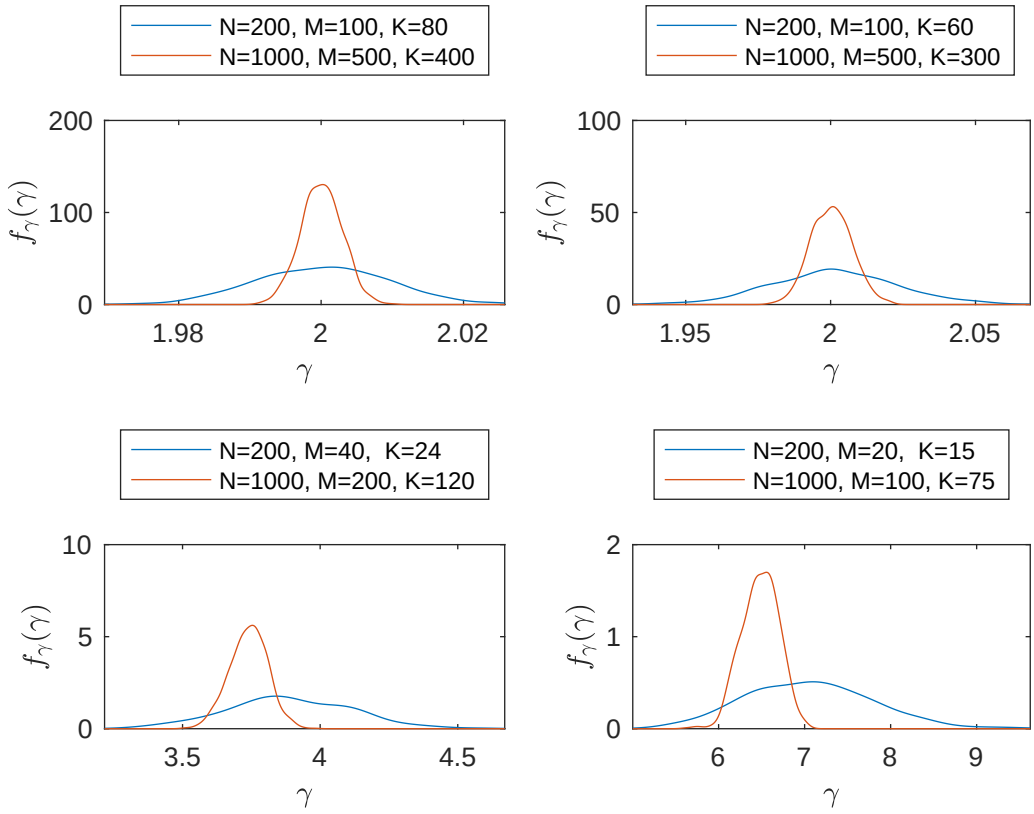


Figure B.4: pdf of γ, $\delta = 0.9$, $\varepsilon = 0.01$, discrete distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Gaussian distribution, fixed factor $\tilde{\gamma}$:

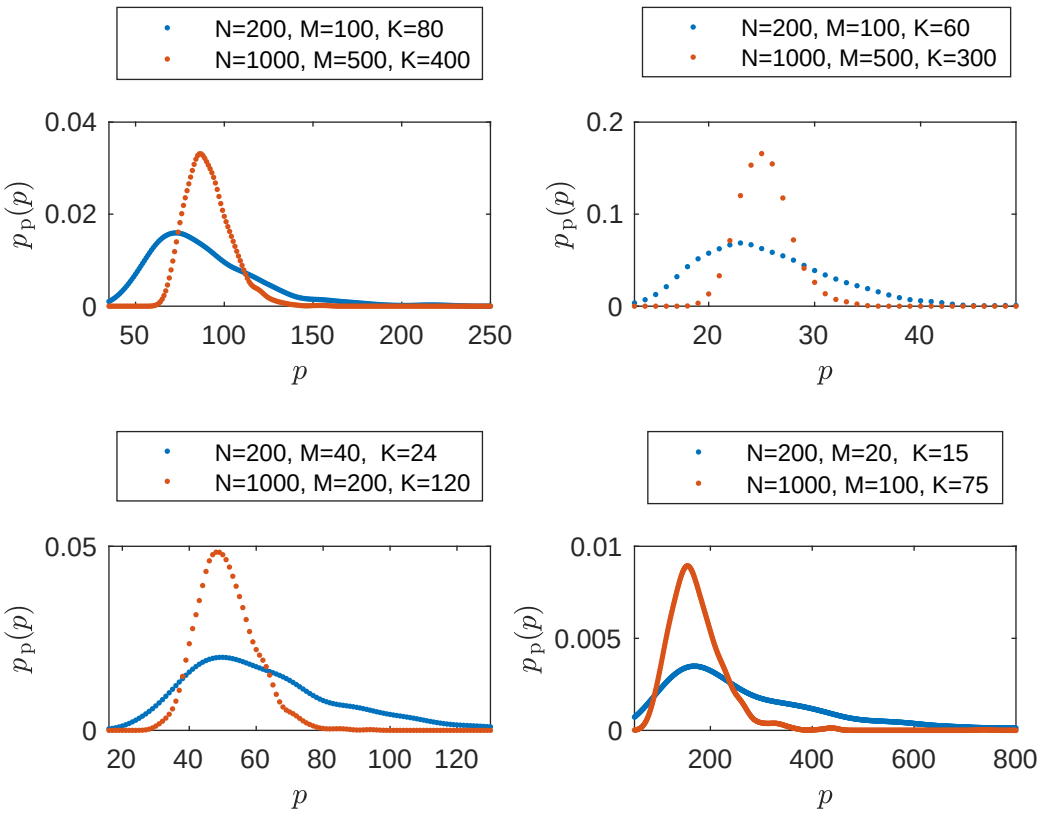


Figure B.5: pmf of p, fixed $\tilde{\gamma} = \bar{\gamma} - 5\sigma_\gamma$, $\delta = 0.9$, $\varepsilon = 0.01$, gaussian distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Discrete distribution, fixed factor $\tilde{\gamma}$:

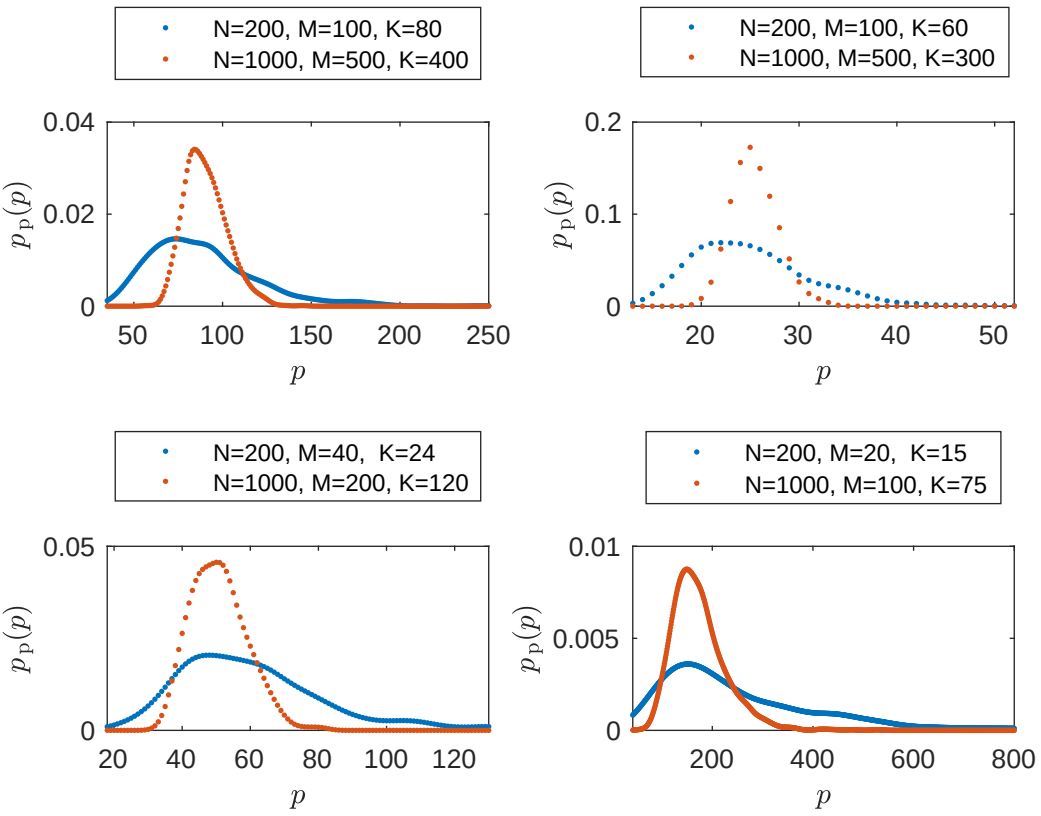


Figure B.6: pmf of p, fixed $\tilde{\gamma} = \bar{\gamma} - 5\sigma_\gamma$, $\delta = 0.9$, $\varepsilon = 0.01$, discrete distribution of entries of weight matrix (one plot for each pair α, β according to Table 4.2).

Bibliography

[1] N. Goertz. "Sampled graph-signals: Iterative recovery with an analytic error bound". In: *IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2017, pp. 1485–1489.

[2] X. Zhu and M. Rabbat. "Graph spectral compressed sensing for sensor networks". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2012, pp. 2865–2868.

[3] J. Shi and J. Malik. "Normalized cuts and image segmentation". In: *IEEE Transactions on pattern analysis and machine intelligence* 22 (2000), pp. 888–905.

[4] F. Zhang and E. R. Hancock. "Pattern Recognition: Graph spectral image smoothing using the heat kernel". In: *Elsevier* 41 (2008), pp. 3328–3342.

[5] S. K. Narang, Y. H. Chao, and A. Ortega. "Graph-wavelet filterbanks for edge-aware image processing". In: *Statistical Signal Processing Workshop (SSP), IEEE*. IEEE. 2012, pp. 141–144.

[6] M. Belkin and P. Niyogi. "Semi-supervised learning on Riemannian manifolds". In: *Machine learning* 56 (2004), pp. 209–239.

[7] D. I. Shuman et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains". In: *IEEE Signal Processing Magazine* 30.3 (2013), pp. 83–98.

[8] J. Bang-Jensen and G. Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

[9] A. Sandryhaila and J. M. F. Moura. "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure". In: *IEEE Signal Processing Magazine* 31.5 (2014), pp. 80–90.

[10] M. Puschel and J. M. F. Moura. "Algebraic signal processing theory: Foundation and 1-D time". In: *IEEE Transactions on Signal Processing* 56.8 (2008), pp. 3572–3585.

[11] A. V. Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[12] A. Sandryhaila and J. M. F. Moura. "Discrete signal processing on graphs". In: *IEEE transactions on signal processing* 61.7 (2013), pp. 1644–1656.

[13] P. Lancaster and M. Tismenetsky. *The Theory of Matrices*. 2nd. Elsevier, 1985.

[14] F. R. Gantmacher. *The Theory of Matrices,Chelsea, New York,* vol. 1. Chelsea Publishing Company, New York, 1959, p. 276.

[15] G. H. Golub and C. F. Van Loan. *Matrix computations*. Vol. 3. JHU Press, 2012.

[16] S. Chen et al. "Discrete Signal Processing on Graphs: Sampling Theory". In: *IEEE transactions on signal processing* 63.24 (2015), pp. 6510–6523.

[17] I. Pesenson. "Sampling in Paley-Wiener spaces on combinatorial graphs". In: *Transactions of the American Mathematical Society* 360 (2008), pp. 5603–5627.

[18] S. K. Narang, A. Gadde, and A. Ortega. "Signal processing techniques for interpolation in graph structured data". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2013, pp. 5445–5449.

[19] A. Anis, A. Gadde, and A. Ortega. "Towards a sampling theorem for signals on arbitrary graphs". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 3864–3868.

[20] M. Vetterli, J. Kovačević, and V. K. Goyal. *Foundations of signal processing*. Cambridge University Press, 2014.

[21] S. Chen, A. Sandryhaila, and J. Kovačević. "Sampling theory for graph signals". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2015, pp. 3392–3396.

[22] H. G. Feichtinger and K. Gröchenig. "Theory and practice of irregular sampling". In: *Wavelets: mathematics and applications* 1994 (1994), pp. 305–363.

[23] K. Gröchenig. "Reconstruction algorithms in irregular sampling". In: *Mathematics of Computation* 59 (1992), pp. 181–194.

[24] E. Cuthill and J. McKee. "Reducing the bandwidth of sparse symmetric matrices". In: *ACM '69 Proceedings of the 1969 24th national conference*. ACM. 1969, pp. 157–172.

Code of Conduct :

Hiermit erkläre ich, dass die vorliegende Arbeit gemäß dem Code of Conduct—Regeln zur Sicherung guter wissenschaftlicher Praxis (in der aktuellen Fassung des jeweiligen Mitteilungsblattes der TU Wien), insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In– noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Vienna, June 2018

Sara Abbasi