Distributed graph topology inference from streaming data

Mircea Moscu

UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES FONDAMENTALES ET APPLIQUÉES

THÈSE DE DOCTORAT

Inférence distribuée de topologie de graphe à partir de flots de données

Mircea MOSCU

Laboratoire J.-L. LAGRANGE

Présentée en vue de l'obtention du grade de docteur en sciences de l'ingénieur
d'Université Côte d'Azur
Dirigée par : Cédric Richard
Soutenue le : 16 décembre 2020

Devant le jury, composé de :
Hichem Snoussi, PU, Université de Technologie de Troyes
Pierre Borgnat, Directeur de Recherche CNRS, ENS de Lyon
André Ferrari, PU, Université Côte d'Azur
Stephen McLaughlin, Professor, Heriot-Watt University
François Septier, PU, Université Bretagne Sud
Jean-Yves Tourneret, PU, Toulouse INP
Rémi Flamary, MCF, HdR, École Polytechnique

Inférence distribuée de topologie de graphe à partir de flots de données

Rapporteurs
Hichem Snoussi, PU, Université de Technologie de Troyes, France
Pierre Borgnat, Directeur de Recherche CNRS, ENS de Lyon, France

Examinateurs
André Ferrari, PU, Université Côte d'Azur, Nice, France
Stephen McLaughlin, Professor, Heriot-Watt University, Edinburgh, United Kingdom
François Septier, PU, Université Bretagne Sud, Morbihan, France
Jean-Yves Tourneret, PU, Toulouse INP, France

Invités
Rémi Flamary, MCF, HdR, École Polytechnique, Palaiseau, France

Encadrant
Cédric Richard, PU, Université Côte d'Azur, Nice, France

Abstract

Distributed graph topology inference from streaming data

The second decade of the current millennium can be summarized in one short phrase: the advent of data. There has been a surge in the number of data sources: from audio-video streaming, social networks and the Internet of Things, to smartwatches, industrial equipment and personal vehicles, just to name a few. More often than not, these sources form networks in order to exchange information. As a direct consequence, the field of Graph Signal Processing has been thriving and evolving. Its aim: process and make sense of all the surrounding data deluge.

In this context, the main goal of this thesis is developing methods and algorithms capable of using data streams, in a distributed fashion, in order to infer the underlying networks that link these streams. Then, these estimated network topologies can be used with tools developed for Graph Signal Processing in order to process and analyze data supported by graphs.

After a brief introduction followed by motivating examples, we first develop and propose an online, distributed and adaptive algorithm for graph topology inference for data streams which are linearly dependent. An analysis of the method ensues, in order to establish relations between performance and the input parameters of the algorithm. We then run a set of experiments in order to validate the analysis, as well as compare its performance with that of another proposed method of the literature.

The next contribution is in the shape of an algorithm endowed with the same online, distributed and adaptive capacities, but adapted to inferring links between data that interact nonlinearly. As such, we propose a simple yet effective additive model which makes use of the reproducing kernel *machinery* in order to model said nonlinearities. The results if its analysis are convincing, while experiments ran on biomedical data yield estimated networks which exhibit behavior predicted by medical literature.

Finally, a third algorithm proposition is made, which aims to improve the nonlinear model by allowing it to escape the constraints induced by additivity. As such, the newly proposed model is as general as possible, and makes use of a natural and intuitive manner of imposing link sparsity, based on the concept of partial derivatives. We analyze this proposed algorithm as well, in order to establish stability conditions and relations between its parameters and its performance. A set of experiments are ran, showcasing how the general model is able to better capture nonlinear links in the data, while the estimated networks behave coherently with previous estimates.

Keywords: network topology, graph signal processing, distributed learning, online graph estimation, linear dependence, nonlinear dependence, reproducing kernels, sparse networks, algorithm analysis

Résumé

Inférence distribuée de topologie de graphe à partir de flots de données

La deuxième décennie du millénaire actuel peut être résumée en une courte phrase : l'essor des données. Le nombre de sources de données s'est multiplié : du streaming audio-vidéo aux réseaux sociaux et à l'Internet des Objets, en passant par les montres intelligentes, les équipements industriels et les véhicules personnels, pour n'en citer que quelques-unes. Le plus souvent, ces sources forment des réseaux afin d'échanger des informations. En conséquence directe, le domaine du Traitement de Signal sur Graphe a prospéré et a évolué. Son but : traiter et donner un sens à tout le déluge de données environnant.

Dans ce contexte, le but principal de cette thèse est de développer des méthodes et des algorithmes capables d'utiliser des flots de données, de manière distribuée, afin d'inférer les réseaux sous-jacents qui relient ces flots. Ensuite, ces topologies de réseau estimées peuvent être utilisées avec des outils développés pour le Traitement de Signal sur Graphe afin de traiter et d'analyser les données supportées par des graphes.

Après une brève introduction suivie d'exemples motivants, nous développons et proposons d'abord un algorithme en ligne, distribué et adaptatif pour l'inférence de topologies de graphes pour les flots de données qui sont linéairement dépendants. Une analyse de la méthode s'ensuit, afin d'établir des relations entre les performances et les paramètres nécessaires à l'algorithme. Nous menons ensuite une série d'expériences afin de valider l'analyse et de comparer ses performances avec celles d'une autre méthode proposée dans la littérature.

La contribution suivante est un algorithme doté des mêmes capacités en ligne, distribuées et adaptatives, mais adapté à l'inférence de liens entre des données qui interagissent de manière non-linéaire. À ce titre, nous proposons un modèle additif simple mais efficace qui utilise l'*usine* du noyau reproduisant afin de modéliser lesdites non-linéarités. Les résultats de son analyse sont convaincants, tandis que les expériences menées sur des données biomédicales donnent des réseaux estimés qui présentent un comportement prédit par la littérature médicale.

Enfin, une troisième proposition d'algorithme est faite, qui vise à améliorer le modèle non-linéaire en lui permettant d'échapper aux contraintes induites par l'additivité. Ainsi, le nouveau modèle proposé est aussi général que possible, et utilise une manière naturelle et intuitive d'imposer la parcimonie des liens, basée sur le concept de dérivés partiels. Nous analysons également l'algorithme proposé, afin d'établir les conditions de stabilité et les relations entre ses paramètres et ses performances. Une série d'expériences est menée, montrant comment le modèle général est capable de mieux saisir les liens non-linéaires entre les données, tandis que les réseaux estimés se comportent de manière cohérente avec les estimations précédentes.

Mots-clés : topologie des réseaux, traitement de signaux sur graphe, apprentissage distribué, estimation de graphe en ligne, dépendance linéaire, dépendance non-linéaire, noyau reproduisant, réseaux parcimonieux, analyse des algorithmes

To my beloved Diana

Pentru părinţii mei Simona și Dănuț

Pentru sora mea Mirela

Acknowledgements

First and foremost I would like to extend my utmost respect and gratitude towards my supervisor, Prof. Cédric Richard. His trust and support, supplemented by patience and investment, ensured a fruitful and enriching experience for me during these past three years. *Merci beaucoup !*

I would also like to thank Rémi Flamary for his limitless and contagious enthusiasm. His dedication towards research and science in general proved to be the nudge I needed in seizing the opportunity of a doctorate.

I want to express my gratitude towards Daniel Gaffé for his care, dedication and calm demeanor, qualities which make one a role model for their students. Sincere thanks to the academic body I was lucky to interact with before and during my doctorate for their investment, professionalism and encouragement. In particular, *un grand merci* to Prof. François Verdier, Gilles Menez, Prof. Adam Parusiński, Prof. Claire Migliaccio, Prof. Cédric Bernardin and Prof. Florea Hănţilă.

Many thanks to Prof. Hichem Snoussi, Dr. Pierre Borgnat, Prof. André Ferrari, Prof. Stephen McLaughlin, Prof. François Septier and Prof. Jean-Yves Tourneret for having accepted to be part of my thesis committee.

I am lucky to have met Roula Nassif and Fei Hua. Their advice was priceless during the beginning of my doctorate and I am grateful to them. A big and sincere *obrigado* to Ricardo Borsoi, a hard-working and passionate person which won my respect and appreciation. Thank you all for your advice and discussions!

Many people proved to be great and supportive friends. Among them, I would especially like to thank Lyes Khacef for being a kind and inspiring person, alongside Paul Jégat, Rémy Garcia and Katarzyna Tomasiak for their sincere friendships. Thanks to them, and many others, I managed to keep going even during hardship. I wish them the best of luck in their endeavors.

My dear Diana has my sincere love, respect and appreciation. Not only because she has always been alongside me, but also because she had the care and patience to make me continuously want to surpass myself.

My parents Simona and Dănuț have always supported me and my sister in every step we took. I thank them for their sacrifice and unconditional love. My sister, Mirela, has my love and care for always being an oasis of fun and stability. *Vă mulţumesc și vă iubesc enorm!*

Table of Contents

List of Figures

List of Tables

List of Algorithms

Notations

a	Normal font lowercase letters denote scalars
$\lvert a \rvert$	Absolute value of scalar a; can also be applied element-wise
$\text{sign}\{\cdot\}$	Sign operator; can also be applied element-wise
$\lceil a \rceil$	Ceiling operator, i.e., returns the least integer greater or equal to a
$\text{mod}(a, b)$	Modulo operator, i.e., returns the remainder of the integer divion of a by b
$\boldsymbol{a}$	Boldface lowercase letters denote column vectors
$\boldsymbol{a}^\top$	Transpose of vector $\boldsymbol{a}$, i.e., the row-vector form of $\boldsymbol{a}$
$a_k\,, [\boldsymbol{a}]_k$	k^{th} entry of vector $\boldsymbol{a}$
$\boldsymbol{A}$	Boldface uppercase letters denote matrices
$\boldsymbol{A}^\top$	Transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{-1}$	Inverse of matrix $\boldsymbol{A}$
$A_{ij}\,, [\boldsymbol{A}]_{ij}$	$(i,j)^{\text{th}}$ entry of matrix $\boldsymbol{A}$
$[\boldsymbol{A}]_{i,\bullet}$	i^{th} row of matrix $\boldsymbol{A}$
$[\boldsymbol{A}]_{\bullet,j}$	j^{th} column of matrix $\boldsymbol{A}$
$\text{Tr}\{\boldsymbol{A}\}$	Trace of the matrix $\boldsymbol{A}$, i.e., the sum of all main-diagonal entries
$\det\{\boldsymbol{A}\}$	Determinant of the matrix $\boldsymbol{A}$, i.e., the product of all eigenvalues
$\text{supp}\{\boldsymbol{A}\}$	Support of the matrix $\boldsymbol{A}$, i.e., the set of its non-zero entries
$\text{sym}\,\{\boldsymbol{A}\}$	Shorthand for $\boldsymbol{A} + \boldsymbol{A}^\top$
$\boldsymbol{A} \otimes \boldsymbol{B}$	Kronecker product of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$
$\boldsymbol{A} \circ \boldsymbol{B}$	Hadamard product of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, i.e., element-wise product
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^N$	N-dimensional Euclidean space
$\mathbb{N}_+$	Set of positive integers
$\mathcal{A}$	Normal font caligraphic letters denote sets
$\text{card}\{\mathcal{A}\}$	Cardinality of set $\mathcal{A}$, i.e., the number of elements in set $\mathcal{A}$
$\mathcal{A} \setminus \mathcal{B}$	Set difference between sets $\mathcal{A}$ and $\mathcal{B}$
$\lVert \cdot \rVert_0$	Pseudo ℓ_0-norm of its vector/matrix argument, i.e.,the number of non-zero entries
$\lVert \cdot \rVert_1$	ℓ_1-norm of its vector argument; sums the absolute values of the entries
$\lVert \cdot \rVert$	ℓ_2-norm of its vector argument; also known as the Euclidean norm
$\lVert \cdot \rVert_{\text{F}}$	Frobenius norm of its matrix argument; the square root of the sum of squares of all entries

$\mathbf{0}$	Vector or matrix with all-zero entries
$\mathbb{1}$	Vector with all-one entries
$\boldsymbol{I}$	Identity matrix
$\mathbb{E}\{\cdot\}$	Expected value operator
$\nabla_x f$	Gradient vector of function f with respect to $\boldsymbol{x}$
$\mathfrak{N}(m, \sigma^2)$	Normal distribution with mean m and variance σ^2; also known as a Gaussian distribution
$\mathfrak{U}(a, b)$	Uniform distribution with parameters a and b
$\text{col}\{\cdot\}$	Column vector or block-vector obtained by stacking its arguments
$\text{vec}\{\cdot\}$	Column block-vector obtained by stacking the columns of its matrix argument
$\text{vec}^{-1}\{\cdot\}$	Square matrix whose columns are the blocks of its block-vector argument
$\text{diag}\{\cdot\}$	Diagonal matrix whose non-zero entries are the arguments of the operator, in order
$\lambda_{\max}(\cdot)$	Maximum eigenvalue of its matrix argument
$\rho(\cdot)$	Spectral radius of its matrix argument, i.e., the largest absolute value of its eigenvalues

Acronyms & Abbreviations

ADMM	Alternating Direction Method of Multipliers
EIER	Edge Identification Error Rate
GFT	Graph Fourier Transform
GSO	Graph Shift Operator
GSP	Graph Signal Processing
i.i.d.	independent and identically distributed
LMS	Least Mean Squares
MSD	Mean Square Deviation
MSE	Mean Square Error
NMSD	Normalized Mean Square Deviation
r.h.s.	right-hand side
RKHS	Reproducing Kernel Hilbert Space
w.r.t.	with respect to

CHAPTER 1

INTRODUCTION

Contents

This first chapter serves as a review and brief introduction into the field of Graph Signal Processing (GSP). We recall useful definitions, establish motivating examples, as well as enumerate a selection of pertinent applications. We then move on to an overview of the objectives of the thesis and, ultimately, its contributions. The final section of this introductory chapter is dedicated to presenting the overarching structure of the manuscript alongside a list of published or to-be-published works.

1.1 About graphs and distributed processing

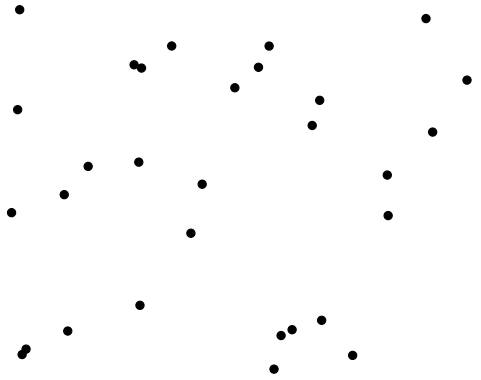
GSP is an active research field, given its inherent properties of exploiting underlying relationships between certain data, and even more so when dealing with large quantities of data [Sandryhaila and Moura, 2014]. Another inherent advantage of graphs is the distributive aspect. In many cases, e.g., graphs based on geographical distances when measuring temperatures at different stations [Zhou et al., 2010, Spelta and Martins, 2018] or image similarity in image processing [Sanfeliu et al., 2002, Tremeau and Colantoni, 2000], one node (be it a meteorological station or a pixel) is *linked* with only its close neighborhood, nodes among which an interdependence relationship exists. This fact allows for the developing of methods and algorithms which can take advantage of distributed computations, in which every node is able to process information locally, even when network-wide they work together in , e.g., graph filter estimation, towards the same global goal [Nassif et al., 2017a], or towards locally similar goals [Nassif et al., 2017b].

The graph-based framework has been successfully applied in many fields and applications, of which we are listing a few in order to illustrate its success:

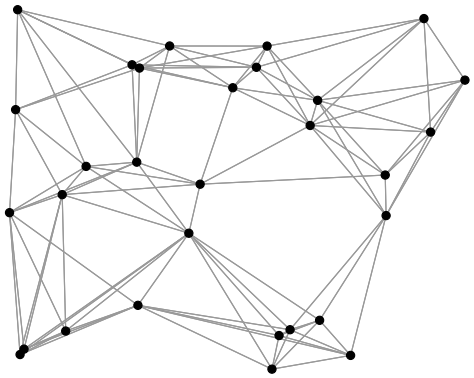
- Low-dimensional representation of high-dimensional data[Belkin and Niyogi, 2003]
- Source identification of a rumor or contagion [Shah and Zaman, 2011, Lesot et al., 2012]
- Recommendation systems [Narang et al., 2013]
- Environmental monitoring in a smart city [Jabłoński, 2017]
- Ship detection in radar images [Salembier et al., 2018]
- City traffic analysis [Deri and Moura, 2016]
- Genomics and biology [Alekseyenko et al., 2011, Kim et al., 2019]
- Terrain and topology modeling [Cioacă et al., 2019]
- Pharmacological property prediction in drug development [Lukovits, 1992]

Each and any of these or other applications can be abstractly summarized in the same unique manner, using both graph and signal processing terminology. A set of separate agents are spread across a certain area, relevant to the application. These agents represent the nodes of the graph. Each one of them is able to acquire and transmit a measure of a certain quantity. These measurements represent the signal. The agents interact among each other, and these interactions are modeled by the edges of a graph. Together with the nodes, they represent the domain which supports the signal. For the simple example of a road network, the agents can be cities within a region. The signal at every agent can be a scalar representing the difference between the number of cars exiting and entering within a time interval. Finally, the edges are, quite intuitively, a model of the road network. The concepts of neighborhood and local interaction can also be explained via this particular example: more traffic can be expected to occur locally, between closer cities, due to, e.g., commuters. As such, the exchange of *information* is more relevant on a local scale. These concepts are illustrated in Fig. 1.1. Under the graph signal processing framework, the goal in this context may be modeling and predicting the flow of traffic and jams.

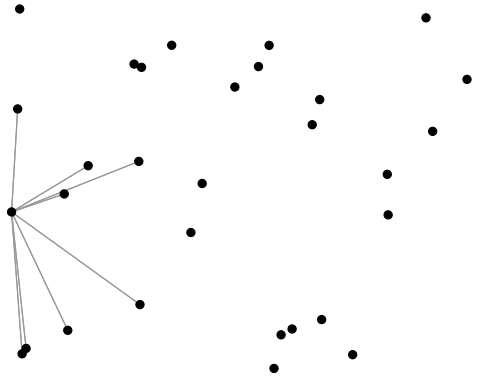
The notion of *graph* is used interchangeably with the one of *network* throughout this manuscript, because of how the latter can be mathematically modeled as an instance of the former. Moreover, the applicability of the developed methods to real-world cases allows, at least in the context of this work, for the interchangeable use of the two terms. The same arguments hold for the couple *node*, which is the main structural component of a graph, and *agent*, which represents the main acting entity in a network.



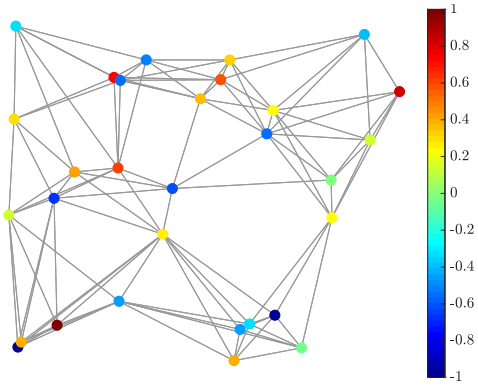
(a) The nodes of the graph. They can also be numbered



(b) The graph, formed by the nodes and edges which connect them



(c) The neighborhood of a certain node



(d) The graph signal

Figure 1.1: Various graph and graph signal concepts

1.2 Thesis structure and publications

Each chapter is aimed to build upon the previous one. As such, the structure and organization of this manuscript is linear, as depicted in Fig. 1.2.

The second chapter mainly serves to motivate the direction in research that we undertook. It further develops and introduces concepts in GSP and emphasize the ubiquity of the graph topology in this field. Based on this fact, we note how most, if not all, processing methods require knowledge of this structure, which in many cases is not available. We end the chapter with an overview of state of the art methods in inferring such topologies. They are separated in two groups, where the first showcases previously proposed methods which consider linear dependencies between agents, while the second treats the case of nonlinear dependencies.

The third chapter introduces our proposed method for topology inference under the assumption of linear dependencies. We present an online, distributed and adaptive solution, supported by a performance analysis. This chapter ends with a set of experimental results which aim to emphasize the simplicity and adaptability of our method. Moreover, they show how the estimated topologies can then be used in follow-up processing algorithms on graphs, such as clustering.

The main results recalled in this chapter were published in:

- M. Moscu, R. Nassif, F. Hua, and C. Richard. Learning causal networks topology from streaming graph signals. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019. doi: 10.23919/EUSIPCO.2019.8902826

- M. Moscu, R. Nassif, F. Hua, and C. Richard. Apprentissage distribué de la topologie d'un graphe à partir de signaux temporels sur graphe. In *Actes du 27e Colloque GRETSI sur le Traitement du Signal et des Images*, 2019

Chapter four aims to further develop the method previously introduced, by considering the case of nonlinear dependencies between nodes. It starts by introducing an additive nonlinear model, before moving on to the concept of Reproducing Kernel Hilbert Space (RKHS). Supported by real-world applications, this new method makes use of kernel functions in order to model nonlinear relationships between agents. A convincing set of experiments is presented, including some on real data.

The work presented in this chapter was published in:

- M. Moscu, R. Borsoi, and C. Richard. Online graph topology inference with kernels for brain connectivity estimation. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2020a

- M. Moscu, R. Borsoi, and C. Richard. Convergence analysis of the graph-topology-inference kernel LMS algorithm. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021a

The fifth chapter works under the same nonlinear premise as the previous one. It introduces a more general nonlinear model, in order to obtain an algorithm able to breach the constraints of the previous additive model. We then continue by introducing a sparsity-imposing regularizer based on partial derivatives. The choice of this particular regularizer is motivated by real-world network examples. An algorithm analysis is then provided, just before introducing a set of convincing experiments.

The work presented in this chapter is based on:

- M. Moscu, R. Borsoi, and C. Richard. Online kernel-based graph topology identification with partial-derivative-imposed sparsity. In *28th European Signal Processing Conference (EUSIPCO)*, pages 2190–2194, 2021b. doi: 10.23919/Eusipco47968.2020.9287624

- M. Moscu, R. Borsoi, and C. Richard. Graph Topology Inference with Kernels and Partial-derivative-imposed Sparsity: Algorithm and Convergence Analysis. 2020b. submitted

The manuscript ends with a set of concluding remarks. Their aim is to summarize the contributions presented throughout this work, while proposing future research directions.

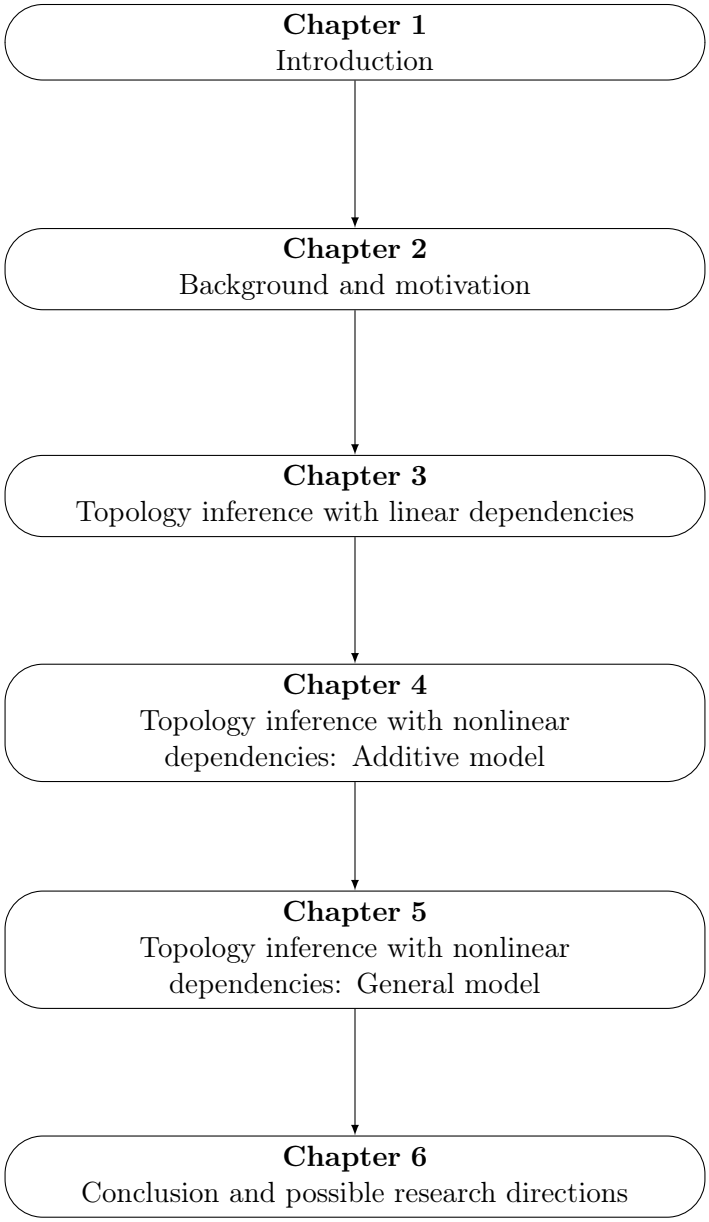


Figure 1.2: Thesis structure

CHAPTER 2

Background and motivation

Contents

Representing a relatively new area of study, the field of GSP is in continuous expansion. During a short time span, less than ten years, it has seen an impressive growth, being adapted to many applications and recently introduced in the spotlight with the rise of the Internet of Things [Paul, 2013, George and Thampi, 2018], due to its innate capability of modeling and analyzing networks. Along the present chapter, the first section briefly introduces the notion of graph, followed by some tools developed for GSP, formalizing some notions introduced in the previous chapter. We then follow with a set of arguments to motivate our work on the problem of graph topology inference. The next subsection presents an overview of our general objectives and contributions, before ending the chapter with the state of the art in solving the same problem of graph topology inference.

2.1 Notions of Graph Signal Processing

We start by formally defining a graph and then follow with the definition of a signal on said graph. We introduce the concept of graph shift, before ending the section with a list of tools adapted to the particularities of GSP.

2.1.1 Definitions

Defining a graph: A graph $\mathfrak{G}$ consists of a set $\mathcal{N}$ of N nodes, and a set $\mathcal{E}$ of edges such that if nodes m and n are *linked*, then $(m, n) \in \mathcal{E}$. By *linked* we mean that there exists a relation of influence between the two nodes, be it bi- or unidirectional. For undirected graphs, these node pairs are unordered. Particular to this work, we consider that self-loops do not appear, i.e., $a_{nn} = 0, \forall n$. This is to support the fact that the goal of GSP is to analyze connections between node couples, and thus self-loops are not useful. Notation $\mathcal{N}_n$ stands for the set of indices of nodes in the neighbourhood of node n, i.e., $\mathcal{N}_n = \{m \colon (m, n) \in \mathcal{E}\}$.

The concepts of graph signal and shift matrix: From each of the N nodes, we collect a signal $\boldsymbol{x} \triangleq [x_1, \ldots, x_N]^\top$, assumed to be real-valued, where x_n is the sample of the signal $\boldsymbol{x}$ at node n. For a visual representation of $\mathcal{N}$, $\mathcal{E}$, $\mathcal{N}_n$ and $\boldsymbol{x}$, see Fig. 1.1. We endow the graph $\mathfrak{G}$ with a shift operator [Shuman et al., 2013], defined as an $N \times N$ matrix $\boldsymbol{S}$, which is the algebraic representation of the graph. Entries s_{nm} are zero if $(m, n) \notin \mathcal{E}$, and non-zero real scalars otherwise. This matrix encodes the underlying graph connectivity and dictates the flow of information within the network. Valid choices for this operator are the adjacency matrix $\boldsymbol{A}$, the weighted adjacency matrix $\boldsymbol{W}$ or the Laplacian matrix $\boldsymbol{L}$ (and its variations) [Biggs, 1993].

Choices for the shift matrix: As mentioned above, one choice for the shift matrix is represented by the adjacency matrix $\boldsymbol{A}$, whose elements are defined as:

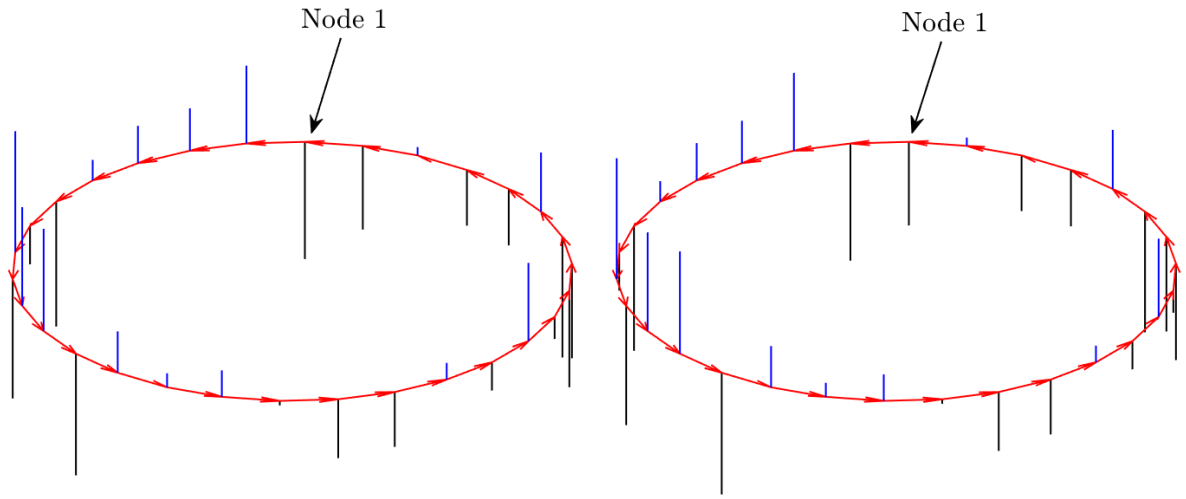
$$A_{nm} = \begin{cases} 0\,, & \text{if } (n, m) \notin \mathcal{E} \\ 1\,, & \text{if } (n, m) \in \mathcal{E} \end{cases} . \tag{2.1}$$

The entries of $\boldsymbol{A}$ are binary, only showing if a link exists or not. On existing links, weights can be affected in order to denote the strength of the connection, thus obtaining the weighted adjacency matrix $\boldsymbol{W}$, with entries:

$$W_{nm} = \begin{cases} 0\,, & \text{if } (n, m) \notin \mathcal{E} \\ w_{nm} \in \mathbb{R} \setminus \{0\}\,, & \text{if } (n, m) \in \mathcal{E} \end{cases} . \tag{2.2}$$

One of the most common choices for the shift matrix $\boldsymbol{S}$ is represented by the so-called Laplacian $\boldsymbol{L}$, defined as:

$$\boldsymbol{L} \triangleq \boldsymbol{D} - \boldsymbol{W}, \tag{2.3}$$



(a) Before the graph shift (b) After the graph shift

Figure 2.1: A graph shift applied on a cyclic, directed, graph. Signal is represented under the form of bars, for easier visualization. For the GSO, the adjacency matrix $\boldsymbol{A}$ was chosen

with $\boldsymbol{D}$ being the degree matrix, whose diagonal entries are:

$$D_{nn} = \sum_{m=1}^{N} W_{nm} \,. \tag{2.4}$$

The graph shift: Once the GSO chosen, it can be applied on the graph signal in order to generate a *shift*. Operation $\boldsymbol{Sx}$ is called a graph shift and can be performed locally at each node n by aggregating samples in its neighborhood, i.e., $\sum_{m \in \mathcal{N}_n} s_{nm} x_m$. Also, $\boldsymbol{S}^k \boldsymbol{x}$ represents a shift of order k that aggregates samples from k-hop neighbors. Node m is a k-hop neighbor of n if n can be reached from m by traveling across at least k edges. For the simple case of a cyclic graph, i.e., $\mathcal{E} = \{(m, m+1) : m \in \{1, \dots, N-1\}\} \cup \{(N, 1)\}$, see Fig. 2.1. An immediate observation concerning this operation is that, depending on the spectrum of the GSO, the energy of the signal may not be preserved when shifts are applied. Works such as [Gavili and Zhang, 2017, Dees et al., 2019] propose a set of shift operators which do not alter the energy of the graph signal when graph shifts are applied.

2.1.2 Graph filters and Graph Fourier Transform

In this subsection we consider an undirected graph $\mathfrak{G}$, meaning that $\boldsymbol{A}$, $\boldsymbol{W}$ and $\boldsymbol{L}$ are symmetric. We note that solutions exist for applying the following tools on directed graphs [Sardellitti et al., 2017], but these remain out of the scope of this chapter.

Graph filters: A linear, shift-invariant graph filter is any matrix $\boldsymbol{H}$ which is a polynomial in the GSO $\boldsymbol{S}$ [Sandryhaila and Moura, 2013b]:

$$\boldsymbol{H} \triangleq h_0\boldsymbol{I} + h_1\boldsymbol{S} + \ldots + h_{L-1}\boldsymbol{S}^{L-1}\,, \tag{2.5}$$

with $\{h_\ell\}_{\ell=0}^{L-1} \in \mathbb{R}^L$ representing the filter taps of the L-order filter $\boldsymbol{H}$. The result of applying the graph filter on a signal yields, just like in classical signal processing, another signal.

Graph Fourier Transform: Consider the eigen-decomposition of $\boldsymbol{S}$:

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}\,, \tag{2.6}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues $\{\lambda_k\}_{k=1}^N$ of $\boldsymbol{S}$. Then the Graph Fourier Transform (GFT) matrix is [Sandryhaila and Moura, 2013c]:

$$\boldsymbol{F} = \boldsymbol{V}^{-1}\,. \tag{2.7}$$

Interestingly, the eigenvalues of the Laplacian, collectively called graph Laplacian spectrum, can be intuitively considered as *frequencies*. See works such as [Shuman et al., 2016] for further details and Fig. 2.2 – 2.3 for a visualization of their behavior. In particular, notice how the first *frequency* $\lambda_1 = 0$ and its corresponding eigenvector v_1 is constant, behavior reminiscent of frequency analysis in classical signal processing.

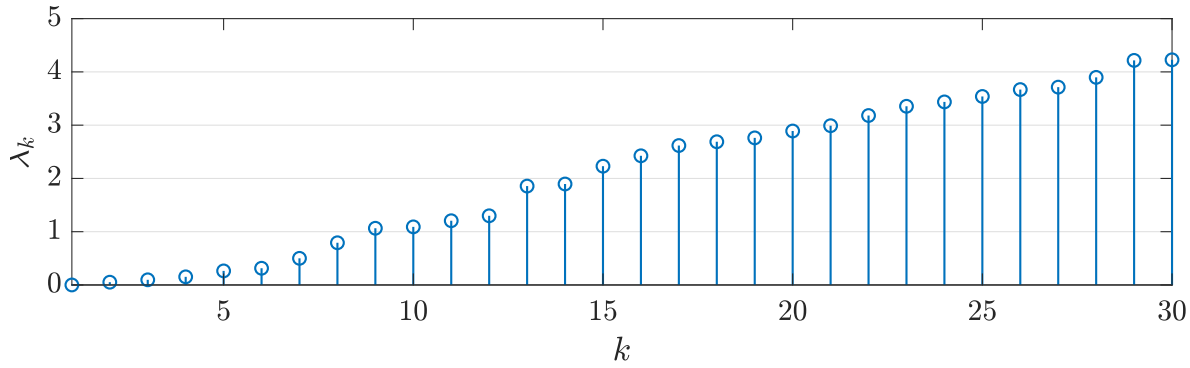
Altogether, notions and tools such as graph signal, filters and Fourier transforms represent the workhorse in the field of Graph Signal Processing, just like their counterparts in classical signal processing.

2.2 Motivating our work

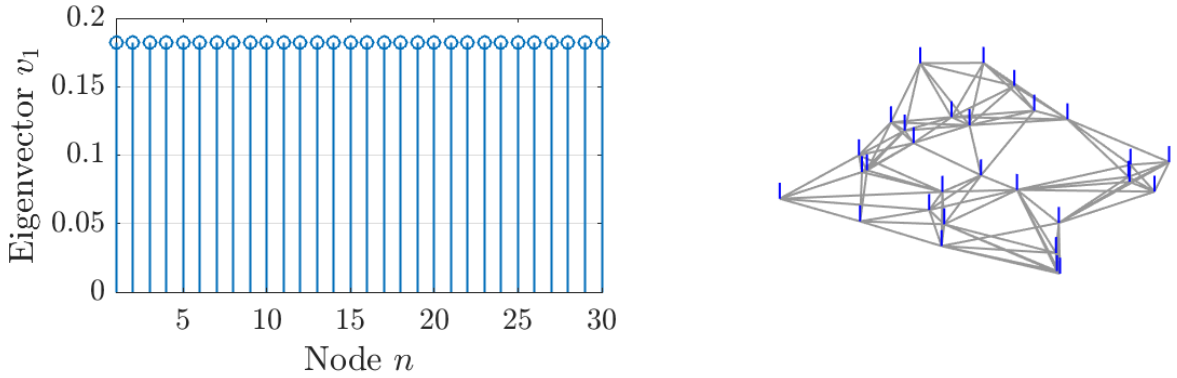
Modern data analysis and processing tasks usually involve large sets of data structured by a graph. Typical examples include data shared by users of social media, traffic on transportation or energy networks, and gene regulatory networks. There are often settings where the network structure is not readily available and the underlying graph explaining the different interactions between participating agents is unknown. In situations such as these, the graph topology has to be estimated from the available data, i.e., the measured graph signals. Moreover, some graphs can be dynamic, such as brain activity supported by neurons or brain regions.

In the particular domain of functional brain imaging, the tools derived by GSP, such as filters and the GFT, can prove insightful in the analysis of brain imaging data [Huang et al., 2018]. The Internet of Things is another highly active area of research and development. In this context, tools and concepts from GSP are highly useful, especially in filtering [Spachos and Plataniotis, 2018].

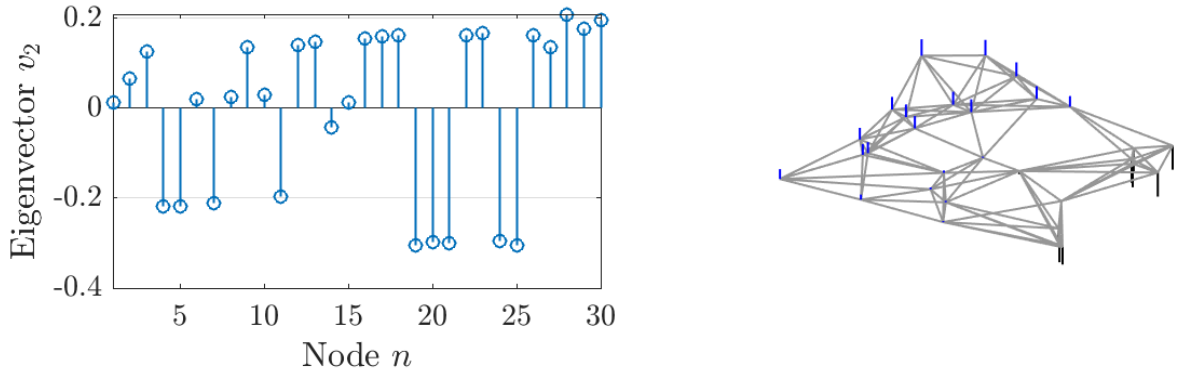
Encoding the graph topology with the adjacency matrix $\boldsymbol{A}$, the weighted adjacency matrix $\boldsymbol{W}$ or the graph Laplacian $\boldsymbol{L}$ is ubiquitous in graph signal models. This operator describes the



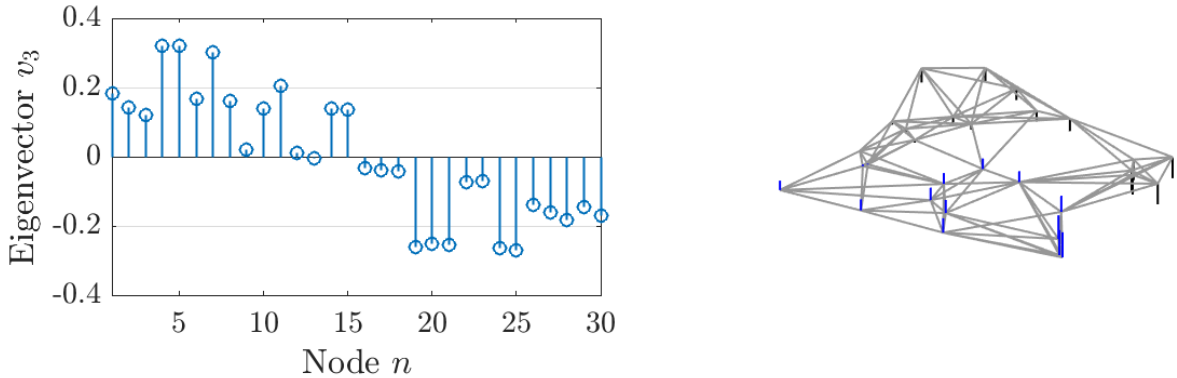
(a) Graph Laplacian spectrum



(b) The 1st eigenvector plotted as a signal (left) and as a graph signal (right)

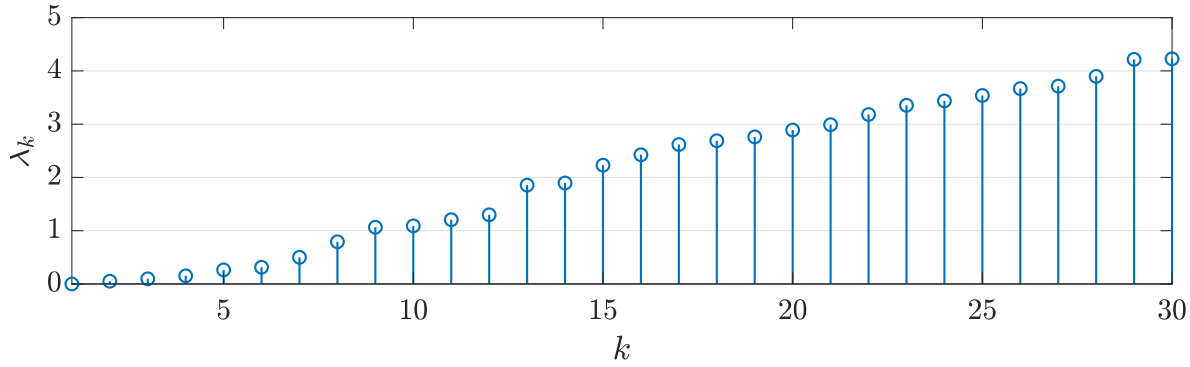


(c) The 2nd eigenvector plotted as a signal (left) and as a graph signal (right)

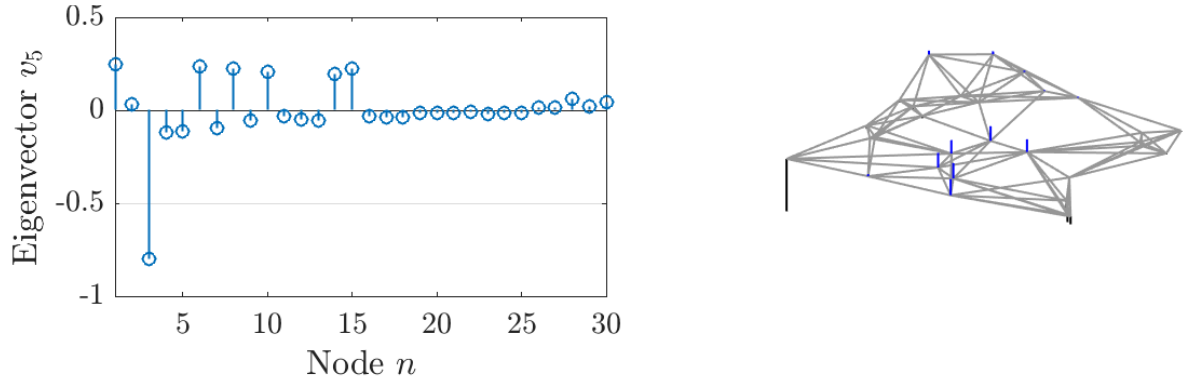


(d) The 3rd eigenvector plotted as a signal (left) and as a graph signal (right)

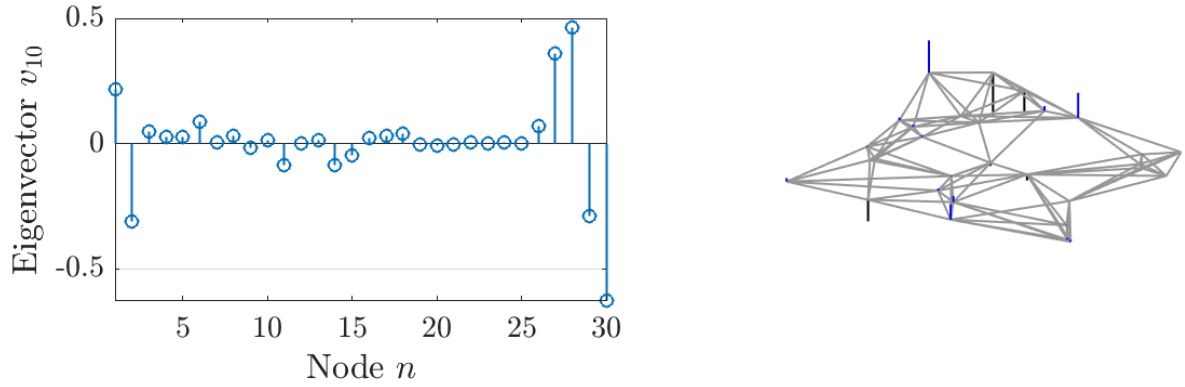
Figure 2.2: The behavior of the eigenvalues and eigenvectors $1, 2, 3$ of the graph Laplacian $\boldsymbol{L}$



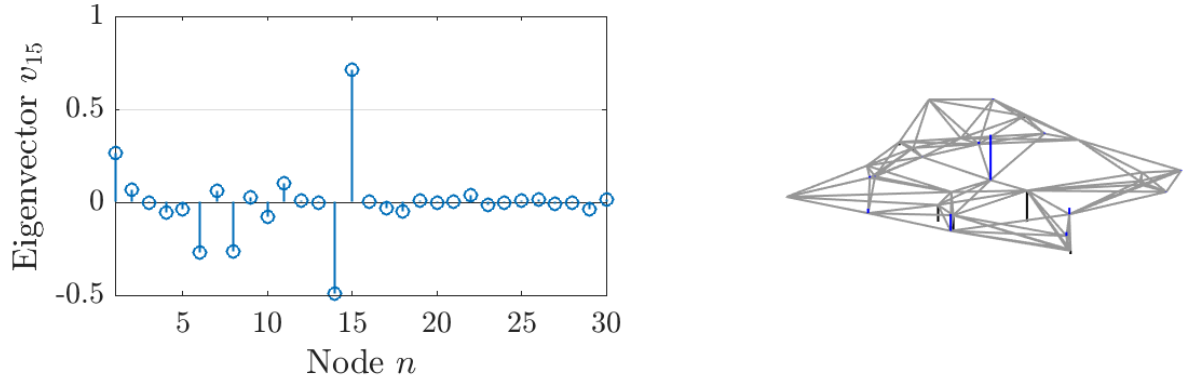
(a) Graph Laplacian spectrum



(b) The 5^{th} eigenvector plotted as a signal (left) and as a graph signal (right)



(c) The 10^{th} eigenvector plotted as a signal (left) and as a graph signal (right)



(d) The 15^{th} eigenvector plotted as a signal (left) and as a graph signal (right)

Figure 2.3: The behavior of the eigenvalues and eigenvectors $5, 10, 15$ of the graph Laplacian $\boldsymbol{L}$

interactions between entities and, by extension, it can be considered as a tool for representing relationships between data. Moreover, as seen in applications such as those from section 1.1, the signal processing tools introduced in section 2.1, and in the research domains cited earlier in this section, knowledge of $\boldsymbol{S}$ is paramount in order to process data on graphs. The context in which we place our work is just before any graph processing can take place, at the step of forming and estimating a pertinent shift matrix $\boldsymbol{S}$. This pre-processing context based on the needed knowledge of the Graph Shift Operator motivates our chosen research direction.

2.3 Objectives and contributions

The main objectives that we set for our research is the development of algorithms of graph topology inference, based on acquired graph signals. To this main objective we add secondary goals, which aim at further improving the developed methods. Firstly, we consider the online framework in which nodal signals represent time series. As such, a network-wide signal $\boldsymbol{y}(i)$ is acquired at every time instant i and used in order to improve the current estimate of $\boldsymbol{S}$. Secondly, we aim at endowing the algorithms with distributive capacities. Indeed, for reasons of privacy and energy consumption in the network, being able to process data in a distributed manner is an advantage [Lee and Zomaya, 2010]. Thirdly, we consider the capacity of adapting to slow changes in said topology. These can occur in areas such as, e.g., brain activity and social interactions.

In terms of contributions, the existing literature on topology inference, as presented in section 2.4, rarely took into consideration the online aspect of some applications. Under this premise, Chapter 3 presents and develops an online algorithm of topology inference, able to adapt to changes in network interactions. The method is then analyzed and compared with another proposition of the literature.

Based on a lack of online methods considering the existence of nonlinear dependencies between nodal measurements, Chapter 4 introduces a novel, node-dependent, algorithm capable of topology inference. The proposed model is additive and takes advantage of the reproducing kernel *machinery*, as well as of dictionary-sparsifying methods. The thorough analysis of the algorithm is then complemented by experiments based on real data. The method reveals topologies which present qualities and metrics coherent with findings by medical studies, as long as other topology inference methods.

A further development of our proposed nonlinear topology inference solution is presented in Chapter 5. The introduced model does not assume any additive relations between nodes, thus allowing for a better representative capacity. Furthermore, it makes use of a natural solution in introducing sparsity in the networks' links. This solution avoids *artificially* introducing sparsity by the use of the ℓ_1-norm, instead relying on partial derivatives. A complex analysis is then set forth, for both the non-regularized and regularized case. The experimental section of this chapter further proves the inferring capacities of the method.

2.4 State of the art in topology inference

This section highlights a few works in the existing literature on topology inference, with more examples and complementary information given on a case by case basis in each of the following chapters. A comprehensive review of the state of the art methods for graph topology inference is given in [Dong et al., 2019].

The remainder of this section showcases a selection of both linear and nonlinear methods. This separation is done in order to match the structure of the future chapters, in which this separation exists as well.

2.4.1 Linear dependencies

For topology identification, several works have been put forward. A very early proposition is in [Dempster, 1972], where a covariance estimation based method of inferring links is introduced. On the same line, in [Friedman et al., 2008] the graphical LASSO is employed in order to estimate the inverse covariance matrix from available data. Other solutions include the use of dictionaries. The authors in [Ding et al., 2020] devise a method for learning a dictionary that is able to efficiently represent the signals as linear combinations of atoms, from which they establish a similarity graph between the data.

Work [Zaman et al., 2017] proposes a vector auto-regressive data model, leading to the development of an online topology inference algorithm. The parameters of the proposed auto-regressive model are estimated and used in order to reveal an underlying directed graph, based on causality [Bolstad et al., 2011]. A scalable algorithm is devised, based on a block coordinate descent implementation. The method is, however, centralized. A similar solution, based on causal graph processes, is proposed in [Ramezani-Mayiami and Beferull-Lozano, 2017]. In [Segarra et al., 2017], the authors advocate that connectivity can be recovered from estimated spectral templates, while the authors of works such as [Sardellitti et al., 2016, 2019] use a similar method with a focus on band-limited signals, i.e., signals whose GFT are sparse. In [Vlaski et al., 2018] the authors propose an online adaptive algorithm for learning the topology from streaming graph signals driven by a diffusion process. Linearity of interactions and signal stationarity are assumed in [Shafipour et al., 2019], while developing an ADMM algorithm. The authors of [Shafipour et al., 2017] introduce and develop a method designed for topology inference for the case when the measured signals are non-stationary.

2.4.2 Nonlinear dependencies

In modeling non-linear phenomena, works such as [Harring et al., 2012, Finch, 2015] focus on polynomial structural equation models, while the authors of [Lim et al., 2015] use their non-linear counterparts. Structural equation models are also used in [Baingana et al., 2013] to track slowly time-varying networks, with application to contagion propagation. They, however, have some

limitations, such as assuming knowledge of certain connections or the form of the non-linear functions. Reproducing kernels have seen widespread use in topology inference problems. One of these works is [Shen et al., 2017] where kernels, chosen to best fit the data, model nonlinear relationships between nodes based on measurements at successive time instants. The authors present an auto-regressive framework that allows to track graph connectivity over time, proving useful in providing insights on brain connectivity. The multi-kernel approach in [Zhang et al., 2017] uses partial correlations to encode graph topology and ℓ_p-norm regression to enhance performance. Another solution is developed in [Lippert et al., 2009], where an unsupervised kernel-based method is implemented. One particularity of the algorithm is that it requires, as a parameter, the number of sought edges. It also offers the possibility of statistical significance testing when setting this parameter. In [Giannakis et al., 2018], a thorough analysis of the kernel-based topology inference problem is given. This last work focuses on capturing both nonlinear and dynamic links, i.e., connections that vary with time.

CHAPTER 3

Topology inference with linear dependencies

Contents

The current chapter focuses on developing a framework of estimating a network structure in an online, distributed and adaptive fashion. Several works proposed centralized offline solutions to address this problem, without paying much attention to the inherent distributed nature of networks. A few other works proposed online, adaptive methods, but are still centralized. A focus is placed on distributed algorithms, throughout the current and the following chapters, for reasons of reducing computational burden, as well as introducing a layer of privacy and security. The principle is that should an agent of the network be compromised, then only the

locally stored and processed information is at risk. Another advantage of distributed algorithms is that should an agent fail, the network would still be able to continue processing until nominal functioning parameters are reestablished.

We depart from a centralized setting and show how, by introducing a simple yet powerful data model, we can infer a graph structure from streaming data with a distributed online learning algorithm. By capturing the dependencies among streaming graph signals, an estimate is obtained in the form of a possibly directed, weighted adjacency matrix. The online and distributed aspects of the method allow for the estimation of networks which change in time, endowing it with topology-tracking capabilities. A performance analysis of the algorithm is proposed, both in the mean and in the mean square sense, as well as a brief study in stability. Our proposed approach is then tested experimentally to illustrate its usefulness, and successfully compared to a centralized offline solution of the existing literature. For illustration purposes, we consider both a symmetry-imposing regularization, tending to the estimation of undirected graphs, as well as a sparsity-imposing one, reliant on the ℓ_1-norm, tending to the estimation of sparse graphs.

The work presented in this chapter was published in:

- M. Moscu, R. Nassif, F. Hua, and C. Richard. Learning causal networks topology from streaming graph signals. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019. doi: 10.23919/EUSIPCO.2019.8902826

- M. Moscu, R. Nassif, F. Hua, and C. Richard. Apprentissage distribué de la topologie d'un graphe à partir de signaux temporels sur graphe. In *Actes du 27e Colloque GRETSI sur le Traitement du Signal et des Images*, 2019

3.1 Introduction

In the last decade, data have become a raw resource that needs to be collected and refined before becoming useful information. Data are abundant and diverse, taking different forms and stemming from different sources: e-commerce, sporting events, entertainment media, and social interactions, to name a few. Structured data, which have a defined format and where each component is linked in some way to others, are ubiquitous and generally evolve over time, making it difficult to process and analyze. Since seminal works such as [Shuman et al., 2013, Sandryhaila and Moura, 2013a], the field of Graph Signal Processing has attracted great attention due to the large array of potential applications it offers. Typical examples include functional brain topology, social media analysis, and transportation or energy networks monitoring.

Most GSP algorithms introduced in the past years assume prior knowledge of the graph structure. However, there are often settings where the graph is not readily available, and has to be inferred from data by capturing the underlying relationship between the characteristics of the observations at each node. This chapter focuses on developing a framework able to estimate

a network structure by capturing the linear dependencies among streaming graph signals. The estimated GSO is in the form of a possibly directed, weighted adjacency matrix, which governs said dependencies.

Under a graph signal smoothness assumption, the so-called pairwise distances matrix $\boldsymbol{Z}$ with entries defined as $z_{nm} \triangleq \|y_n - y_m\|^2$, is introduced in [Kalofolias, 2016] to estimate a weighted adjacency matrix $\boldsymbol{W}$. Notations y_n and y_m denote a scalar measurement acquired at nodes n and m, respectively. The optimization problem they propose is:

$$\underset{\boldsymbol{W}}{\operatorname{argmin}} \|\boldsymbol{W} \circ \boldsymbol{Z}\|_1 - \alpha \mathbb{1}^\top \log(\boldsymbol{W}\mathbb{1}) + \beta \|\boldsymbol{W}\|_{\mathrm{F}}^2 , \tag{3.1}$$

where the ℓ_1-norm and the $\log(\cdot)$ operators are element-wise. The logarithmic constraint helps reducing exceedingly large weights, while the Frobenius-norm regularization reduces the number of less connected nodes and hinders the existence of non-connected ones. The work [Yamada et al., 2019] improves upon problem (3.1) by adding a regularization term to impose temporal sparseness, under the assumption that changes in topology are sparse in time.

A recent work is [Natali et al., 2020], in which the authors consider the connectivity of the network as known, i.e., $\operatorname{supp}\{\boldsymbol{S}\}$, and they focus on jointly estimating interaction weights, as well as filter coefficients which best explain the input – output relation. In order to tackle the non-convexity in the proposed optimization problem due to the joint estimation, the authors employ the alternating minimization approach, iterating alternatively between the filter coefficients and the Graph Shift Operator.

In [Segarra et al., 2017, 2018], the authors advocate that connectivity can be recovered from spectral templates, under the assumption that the graph signal $\boldsymbol{x}$ is stationary and generated through a diffusion process. Starting from the sample covariance, they obtain an estimate of its eigenvectors, which are shown to be the same as those of the Graph Shift Operator, commutation rendered possible by the mild requirements previously imposed on the signal $\boldsymbol{x}$. They then proceed to estimate the shift operator $\boldsymbol{S}$ under a set of constraints that yields a matrix with desirable properties, such as zeros on the diagonal, sparsity or symmetry. The method is shown to be robust to noisy or incomplete spectral templates.

These previous proposals share the limiting characteristic of the approach being offline. As such, they lack in adaptive capabilities. The authors of [Shafipour et al., 2019, Shafipour and Mateos, 2020] improve the spectral-template-based method previously discussed by endowing it with online-estimating capacities. They formulate an optimization problem amenable for an Alternating Direction Method of Multipliers (ADMM) algorithm and also reduce the complexity of the necessary eigen-decomposition from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$, rendering it suitable for the online setting. The method is, however, feasible only when the input signal $\boldsymbol{x}$ is stationary, since this represents the base condition for spectral templates commuting between the estimated covariance matrix and the Graph Shift Operator.

Table 3.1: List of notations and symbols present in Chapter 3

Symbol	Definition
$\boldsymbol{S}$	Shift matrix of a graph
$\boldsymbol{A}$	Adjacency matrix of a graph
$\boldsymbol{W}$	Weighted adjacency matrix of a graph
$\mathcal{N}$	Set of nodes of the graph
$\mathcal{N}_n$	Set of nodes in the neighborhood of node n, excluding node n
$\mathcal{N}_{\backslash n}$	Set of all nodes, excluding node n
$J(\cdot)$	Global cost function
$J_n(\cdot)$	Local cost function
N	Total number of nodes in the graph
K	Number of filter taps

Unlike existing methods, this chapter focuses on developing a method of identifying the topology of a graph from streaming graph signals in a distributed and online manner, without strong constraints on the nature of the input signal.

A set of symbols used throughout the remainder of this chapter are collected in Table 3.1, while others are defined and used locally.

3.2 Centralized problem statement

3.2.1 Shift-invariant graph filtering

The proposed method focuses on a graph-based filtering framework. A graph filter takes a signal on graph $\boldsymbol{x}(i)$ as input, and outputs a signal $\boldsymbol{y}(i)$ given by $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$ indexed by the same graph [Sandryhaila and Moura, 2013b]. For an application such as functional brain topology, the input could be a voltage applied in different brain regions, while the output can be the voltage measured at each of the same regions [Penfield, 1947, Histed et al., 2009]. Another input – output pair can be found in the context of the contagion-like spread of *tweets* on Twitter. For instance, a certain piece of news can be posted by a number of users in a relatively short time span, representing the input, whereas a binary output can be represented by the users which retweet (or not) the news, within a chosen time frame [Lerman et al., 2012].

Different forms have been considered for $\boldsymbol{H}$ in the literature. As a starting point in developing our method, the K^{th} order linear shift invariant graph filter is defined as [Sandryhaila and Moura, 2013a, Shuman et al., 2018]:

$$\boldsymbol{y}(i) = \sum_{k=0}^{K-1} h_k \boldsymbol{S}^k \boldsymbol{x}(i)\,, \quad i \geq 0\,, \tag{3.2}$$

with $\boldsymbol{S}$ denoting a shift matrix and $\{h_k\}_{k=0}^{K-1}$ being the filter coefficients. The shift-invariant property is expressed by the relation $\boldsymbol{S}(\boldsymbol{H}\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{S}\boldsymbol{x})$, meaning that a shift applied on the filtered signal is the same as filtering a shifted signal. Observe that the previous model assumes

the instantaneous diffusion of information, which may appear as a limitation of this model. A dynamical model was proposed to overcome this restriction [Nassif et al., 2018], in which a realistic delay on the input is introduced:

$$\boldsymbol{y}(i) = \sum_{k=0}^{K-1} h_k \boldsymbol{S}^k \boldsymbol{x}(i-k)\,, \quad i \geq K-1\,. \tag{3.3}$$

This model has seen extended use in graph filtering [Nassif et al., 2018, Hua et al., 2018].

3.2.2 Network topology inference

Taking into account the input signal delay, we propose the multivariate, centralized data model defined as:

$$\boldsymbol{y}(i) = \sum_{k=0}^{K-1} \boldsymbol{S}^k \boldsymbol{x}(i-k) + \boldsymbol{v}(i), \quad i \geq K-1\,, \tag{3.4}$$

where $\boldsymbol{S}^k \triangleq \{s_{nm,k}\}$ in the above power series contains regressive coefficients that describe the influence of node m on node n at a distance of k hops, and $\boldsymbol{v}(i)$ is innovation noise. The chosen model is helpful to asses Granger causality, based on the *cause-before-effect* principle [Granger, 1988]. In our context, x_m is said to Granger-cause x_ℓ if knowledge of the former improves the prediction of the latter [Bolstad et al., 2011]. We remark upon the fact that the simplification of model (3.3) into (3.4) is possible due to the goal being the estimation of a suitable GSO, able to explain the underlying connectivity. Assuming that the shift matrix $\boldsymbol{S}$ is known, the authors in [Nassif et al., 2018] show how diffusion adaptation strategies can be applied to estimate the filter coefficients $\{h_k\}_{k=0}^{K-1}$ from streaming data $\{\boldsymbol{x}(i), \boldsymbol{y}(i)\}$. Moreover, h_1 cannot be zero, since this would imply the nonexistence of the network.

Consider a connected network with N nodes. Model (3.4) allows for the use of the assumption that each node ℓ knows the set of its neighbors $\mathcal{N}_\ell$ with which it communicates, while the support of $\boldsymbol{S}$ remains unknown. In order to keep our algorithm as general as possible, we do not assume any connections as known beforehand. The problem is to estimate $\boldsymbol{S}$ from streaming data $\{\boldsymbol{x}(i), \boldsymbol{y}(i)\}$. We assume that signal $\boldsymbol{x}(i)$ is zero-mean wide-sense stationary, i.e., correlation sequence $\boldsymbol{R}_x(k) \triangleq \mathbb{E}\{\boldsymbol{x}(i)\boldsymbol{x}^\top(i-k)\}$ is a function of the time lag k only. The noise $\boldsymbol{v}(i) = [v_1(i), \ldots, v_N(i)]^\top$ is assumed zero-mean, independent and identically distributed (i.i.d.), with covariance $\boldsymbol{R}_v = \operatorname{diag}\left\{\{\sigma_{v,n}\}_{n=1}^N\right\}$. Under these assumptions, estimating matrix $\boldsymbol{S}$ in (3.4) can be performed by solving the following problem:

$$\begin{gathered}\boldsymbol{S}^* = \underset{\boldsymbol{S}}{\operatorname{argmin}}\ \mathbb{E}\left\{\left\|\boldsymbol{y}(i) - \sum_{k=0}^{K-1} \boldsymbol{S}^k \boldsymbol{x}(i-k)\right\|^2\right\} + \eta\,\Psi(\boldsymbol{S})\,, \\ \text{subject to } s_{nm} = 0 \ \text{ if } \ m \notin \mathcal{N}_n\,, s_{nn} = 0\,, \quad n = 1, \ldots, N\end{gathered} \tag{3.5}$$

with $\eta > 0$. The objective function in (3.5) includes a data-fidelity term alongside regularization term $\Psi(\boldsymbol{S})$, which can account for some prior knowledge of $\boldsymbol{S}$ such as symmetry or sparsity. The

constraints aim at forcing to zero the entries s_{nm} of $\boldsymbol{S}$ corresponding to node pairs (n, m) not belonging to the edge set $\mathcal{E}$.

Formulation (3.5) is, however, non-convex due to the matrix polynomial. This leads any resolution algorithm to possibly converge toward a local minimum rather than a global one. Reference [Mei and Moura, 2017] considers a similar problem in a centralized setting where the data across the network are collected and processed by a fusion center. In the next section, we shall show how the entries of $\boldsymbol{S}$ can be estimated in a distributed manner where nodes perform local computations and exchange information only with their one-hop neighbors.

3.3 Distributed solution

The following strategy allows each node to locally estimate its own non-zero entries in $\boldsymbol{S}$. According to (3.4), the output $y_n(i)$ at each node n is given by:

$$y_n(i) = \sum_{k=0}^{K-1} \Big[\boldsymbol{S}^k \boldsymbol{x}(i-k)\Big]_n + v_n(i)\,. \tag{3.6}$$

This can be rewritten as:

$$\begin{aligned} y_n(i) =& \boldsymbol{s}_n^\top \left[\boldsymbol{x}(i-1)\right]_{m\in\mathcal{N}_{\setminus n}} + \boldsymbol{s}_n^\top \left[\boldsymbol{S}\boldsymbol{x}(i-2)\right]_{m\in\mathcal{N}_{\setminus n}} + \ldots + \boldsymbol{s}_n^\top \left[\boldsymbol{S}^{K-2}\boldsymbol{x}(i-K+1)\right]_{m\in\mathcal{N}_{\setminus n}} \\ &+ x_n(i) + v_n(i)\,, \end{aligned} \tag{3.7}$$

with $\boldsymbol{s}_n = \mathrm{col}\{s_{nm} \colon m \in \mathcal{N}_{\setminus n}\}$ the $(N-1)\times 1$ vector aggregating all entries of the n^{th} row of $\boldsymbol{S}$, expect for the n^{th} one. We remark that the constraint $s_{nn} = 0$ is included, for the case of this local model, directly in the definition of $\boldsymbol{s}_n$. Given that the diffusion of information is achieved in the manner depicted in Fig. 3.1, and that node n only needs to estimate $\boldsymbol{s}_n$, matrices $\boldsymbol{S}$ in (3.7) are replaced with their past available estimates $\hat{\boldsymbol{S}}(i-k), k = 1, \ldots, K-2$. By subtracting $x_n(i)$ from $y_n(i)$, (3.7) can be expressed as:

$$\bar{y}_n(i) \triangleq y_n(i) - x_n(i) = \boldsymbol{z}_n^\top(i)\boldsymbol{s}_n + v_n(i)\,, \tag{3.8}$$

where $\boldsymbol{z}_n(i)$ is a $(N-1)\times 1$ column vector defined as:

$$\boldsymbol{z}_n(i) = \sum_{k=0}^{K-2} \Big[\hat{\boldsymbol{S}}_k \boldsymbol{x}(i-k-1)\Big]_{m\in\mathcal{N}_{\setminus n}}\,, \tag{3.9}$$

with:

$$\hat{\boldsymbol{S}}_k(i) \triangleq \hat{\boldsymbol{S}}(i-1)\hat{\boldsymbol{S}}(i-2)\ldots\hat{\boldsymbol{S}}(i-k)\,, \quad \hat{\boldsymbol{S}}_0 = \boldsymbol{I}\,. \tag{3.10}$$

This solution reduces to approximating the powers of $\boldsymbol{S}$ with products of past estimates of $\boldsymbol{S}$ which are available network-wide, i.e., $\boldsymbol{S}^k \approx \hat{\boldsymbol{S}}(i-1)\ldots\hat{\boldsymbol{S}}(i-k)$. Regressor $\boldsymbol{z}_n(i)$ is non-stationary, since its statistical properties depend on the current chosen estimates of $\boldsymbol{S}$. It also

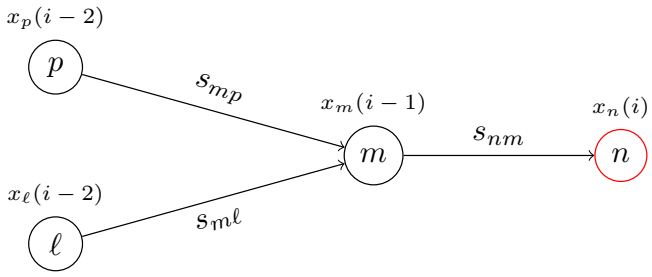


Figure 3.1: Data paths toward node n. Links are depicted as directed edges in order to illustrate the flow of weighted data. In order to estimate its own s_{nm}, node n receives from its neighbour m the $(K-1)$-element vector $[x_m(i-1), s_{mp}x_p(i-2) + s_{m\ell}x_\ell(i-2)]^\top$

represents a quantity that is known at every instant i by the current node, which entails a removal of the initial non-convexity of the problem.

Reformulating (3.7) in the form (3.8) has the following rationale. Consider node n. At time instant i, this node weights the incoming data from its neighbors with the corresponding entries of the n^{th} row of $\boldsymbol{S}$. The same reasoning holds for any neighboring node m of n, as illustrated in Fig. 3.1, which weights its own incoming data with entries of the m^{th} row of $\boldsymbol{S}$. This means that two-hop data sent by node ℓ at time instant $i-2$, passing through node m at time instant $i-1$, and received by node n at time instant i, are successively weighted by $s_{m\ell}$ and s_{nm}. Therefore, when estimating $\boldsymbol{S}$, node n can simply focus on its own weights stored in the n^{th} row of $\boldsymbol{S}$ provided that every other node in the network does the same with its own weights.

Reformulation (3.8) – (3.27) comes along with several benefits compared to the centralized solution in [Mei and Moura, 2017]. The main one concerns computational efficiency since only one-hop regressors $\boldsymbol{z}_n(i)$ are considered at each node n. These one-hop transfers also translate into lower overall communication costs.

We reformulate problem (3.5) by introducing the following aggregate cost function:

$$J(\boldsymbol{S}) = \sum_{n=1}^{N} J_n(\boldsymbol{s}_n)\,, \tag{3.11}$$

where $J_n(\boldsymbol{s}_n)$ denotes the cost at node n, namely:

$$J_n(\boldsymbol{s}_n) \triangleq \mathbb{E}\left\{ |\bar{y}_n(i) - \boldsymbol{z}_n^\top(i)\boldsymbol{s}_n|^2 \,\middle|\, \hat{\boldsymbol{S}}_k, k = 0, \ldots, K-2 \right\} + \eta_n \psi(\boldsymbol{s}_n)\,. \tag{3.12}$$

The expected value is now conditioned by the fact that past estimates of $\boldsymbol{S}$ are fixed and, most importantly for the proposed approach, known. Form (3.12) allows each node n to estimate its known entries $\boldsymbol{s}_n$ of $\boldsymbol{S}$, and to possibly account for some prior knowledge of $\boldsymbol{S}$ via $\psi(\boldsymbol{s}_n)$. We also note that the decomposition (3.11) of the global cost function entails the condition $\Psi(\boldsymbol{S}) = \sum_{n=1}^{N} \psi(\boldsymbol{s}_n)$, under the central assumption of separability of the regularization function.

3.3.1 Symmetry constraint

For illustration purposes, in this subsection, we shall promote symmetry in matrix $\boldsymbol{S}$ via $\Psi(\boldsymbol{S}) = \left\|\text{vec}\left\{\boldsymbol{S} - \boldsymbol{S}^\top\right\}\right\|^2$. This locally translates into the regularizer:

$$\psi_{\text{sym}}(\boldsymbol{s}_n) = \sum_{m \in \mathcal{N}_{\backslash n}} (s_{nm} - s_{mn})^2 \,. \tag{3.13}$$

Following the strategy in [Nassif et al., 2017b], we address this problem by considering the following local cost function:

$$J_n(\boldsymbol{s}_n) \triangleq \mathbb{E}\left\{|\bar{y}_n(i) - \boldsymbol{z}_n^\top(i)\boldsymbol{s}_n|^2\right\} + \eta_n \sum_{m \in \mathcal{N}_{\backslash n}} (s_{nm} - s_{mn})^2 \,, \tag{3.14}$$

where parameter $\eta_n > 0$ controls the relative importance of respecting the symmetry constraint on $\boldsymbol{S}$ [Towfic and Sayed, 2014]. To minimize (3.14), we propose an incremental solution based on gradient descent, namely:

$$\hat{\boldsymbol{s}}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n[\boldsymbol{r}_{z_n y} - \boldsymbol{R}_{z_n}\hat{\boldsymbol{s}}_n(i) - \eta_n\boldsymbol{\delta}(i)] \,, \tag{3.15}$$

where $\boldsymbol{\delta}(i) = \hat{\boldsymbol{s}}_n(i) - \tilde{\boldsymbol{s}}_n(i)$, with $\tilde{\boldsymbol{s}}_n = \text{col}\{\hat{s}_{mn} : m \in \mathcal{N}_{\backslash n}\}$, μ_n a sufficiently small positive step-size, and:

$$\boldsymbol{R}_{z_n} \triangleq \mathbb{E}\{\boldsymbol{z}_n(i)\boldsymbol{z}_n^\top(i)\} \,, \quad \boldsymbol{r}_{z_n y} \triangleq \mathbb{E}\{\boldsymbol{z}_n(i)\bar{y}_n(i)\} \,. \tag{3.16}$$

Note that vector $\tilde{\boldsymbol{s}}_n$ aggregates entries on the n^{th} column of $\hat{\boldsymbol{S}}$, i.e., connections *leaving* from local node n towards any other node m, while the estimated vector $\hat{\boldsymbol{s}}_n$ aggregates connections *arriving* to local node n from any other node m. Under this particular symmetry-enforcing constraint, step-size μ_n in (3.15) must satisfy:

$$0 < \mu_n < \frac{2}{\lambda_{\max}(\boldsymbol{R}_{z_n} + \eta_n\boldsymbol{I})} \,, \tag{3.17}$$

as to guarantee stability in the mean under certain independence conditions on the data [Sayed, 2008].

Update (3.15) may prove difficult to perform in an online fashion, since second-order moments are rarely available beforehand. As a consequence, a stochastic gradient descent strategy can be devised. It consists of choosing instantaneous approximations, such as:

$$\boldsymbol{R}_{z_n} \approx \boldsymbol{z}_n(i)\boldsymbol{z}_n^\top(i) \,, \quad \boldsymbol{r}_{z_n y} \approx \boldsymbol{z}_n(i)\bar{y}_n(i) \,. \tag{3.18}$$

We used the adapt-then-penalize approach introduced in [Yu and Sayed, 2017] to implement the algorithm (3.15) with approximations (3.18). Let us note the local instantaneous error $\varepsilon_n(i)$:

$$\varepsilon_n(i) \triangleq \bar{y}_n(i) - \boldsymbol{z}_n^\top(i)\hat{\boldsymbol{s}}_n(i) \,, \tag{3.19}$$

and introduce the intermediate estimate $\boldsymbol{\phi}_n(i)$, leading to a development of strategy (3.15) into:

$$\boldsymbol{\phi}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n\boldsymbol{z}_n(i)\varepsilon_n(i) \,, \tag{3.20a}$$

$$\hat{\boldsymbol{s}}_n(i+1) = \boldsymbol{\phi}_n(i+1) - \mu_n\eta_n\left[\boldsymbol{\phi}_n(i+1) - \widetilde{\boldsymbol{\phi}}_n(i+1)\right] \,, \tag{3.20b}$$

with $\widetilde{\boldsymbol{\phi}}_n(i+1) \triangleq \operatorname{col}\left\{[\boldsymbol{\phi}_m(i+1)]_n : m \in \mathcal{N}_{\setminus n}\right\}$ the column vector that aggregates all partial estimates related to node n in partial estimates of its neighboring nodes. Note that couple $\boldsymbol{\phi}_n$ – $\widetilde{\boldsymbol{\phi}}_n$ works in a similar way to $\hat{\boldsymbol{s}}_n$ – $\tilde{\boldsymbol{s}}_n$: vector $\widetilde{\boldsymbol{\phi}}_n$ aggregates the entries of the intermediate estimates corresponding to connections *leaving* from node n, while $\boldsymbol{\phi}_n$ aggregates the entries of the intermediate estimates corresponding to connections *arriving* to n. The algorithm is synthesized hereafter.

Algorithm 1: Local estimation of graph topology

For every node n:
Inputs: Parameters μ_n and η_n
Initialization: Initialize all entries of $\hat{\boldsymbol{s}}_n(0)$
Algorithm: At each time instant $i \geq 1$
Collect weighted data $\left[\hat{\boldsymbol{S}}_{k-1}\boldsymbol{x}(i-k)\right]_{m\in\mathcal{N}_{\setminus n}}$ from neighbors
Compute the regressor $\boldsymbol{z}_n(i)$ with (3.27)
Update the local estimate $\hat{\boldsymbol{s}}_n$ with either (3.20), (3.25a) or (3.25b)

3.3.2 Sparsity constraint

In this subsection, we enforce a sparsity constraint on $\boldsymbol{S}$. Such a constraint helps take into account that large real-world graphs tend to be sparse, i.e., one particular node is usually connected to a small subset of the available nodes [Danisch et al., 2018]. This behavior can be noticed in, e.g., web graphs [Gibson et al., 2005], social networks [Speriosu et al., 2011], and biological networks [Harbison et al., 2004]. Thus, we use the *zero-attracting* regularization (ZA):

$$\psi_{\mathrm{ZA}}(\boldsymbol{s}_n) = \sum_{m\in\mathcal{N}_{\setminus n}} |s_{nm}|\,, \tag{3.21}$$

and the *reweighted zero-attracting* regularization (RZA):

$$\psi_{\mathrm{RZA}}(\boldsymbol{s}_n) = \sum_{m\in\mathcal{N}_{\setminus n}} \log\left(1+\frac{|s_{nm}|}{\epsilon}\right)\,, \quad \epsilon>0\,, \tag{3.22}$$

with (3.12), both of which are introduced in [Chen et al., 2009]. In order to minimize (3.12), we employ an incremental solution based on gradient descent, namely [Jin et al., 2018b]:

$$\hat{\boldsymbol{s}}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n\left[\boldsymbol{r}_{z_n y} - \boldsymbol{R}_{z_n}\hat{\boldsymbol{s}}_n(i) - \eta_n \operatorname{sign}\{\hat{\boldsymbol{s}}_n(i)\}\right]\,, \tag{3.23a}$$

$$\hat{\boldsymbol{s}}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n\left[\boldsymbol{r}_{z_n y} - \boldsymbol{R}_{z_n}\hat{\boldsymbol{s}}_n(i) - \eta_n\epsilon^{-1}\frac{\operatorname{sign}\{\hat{\boldsymbol{s}}_n(i)\}}{\mathbb{1}+\epsilon^{-1}|\,\hat{\boldsymbol{s}}_n(i)|}\right]\,, \tag{3.23b}$$

for ZA and RZA, respectively. The division on the right-hand side (r.h.s.) of (3.23b) is performed element-wise. Note that the sign function is always bounded, hence the stability condition for updates (3.23) is [Sayed, 2008]:

$$0 < \mu_n < \frac{2}{\lambda_{\max}(\boldsymbol{R}_{z_n})}\,. \tag{3.24}$$

Table 3.2: List of notations and symbols employed throughout the analysis in Chapter 3

Symbol	Equation
$\boldsymbol{S}^* = \boldsymbol{R}_{zy}\boldsymbol{R}_z^{-1}$	(3.29)
$\boldsymbol{B}_i \triangleq \boldsymbol{I} - \mu\boldsymbol{z}(i)\boldsymbol{z}^\top(i)$	(3.35)
$\boldsymbol{G}_i \triangleq \mu\boldsymbol{v}(i)\boldsymbol{z}^\top(i)$	(3.35)

The instantaneous approximations (3.18) can be once again used, due to the potential unavailability of second order moments in (3.23). This leads to the stochastic gradient updates:

$$\hat{\boldsymbol{s}}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n\boldsymbol{z}_n(i)\varepsilon_n(i) - \mu_n\eta_n\,\mathrm{sign}\{\hat{\boldsymbol{s}}_n(i)\}\,, \tag{3.25a}$$

$$\hat{\boldsymbol{s}}_n(i+1) = \hat{\boldsymbol{s}}_n(i) + \mu_n\boldsymbol{z}_n(i)\varepsilon_n(i) - \mu_n\eta_n\epsilon^{-1}\frac{\mathrm{sign}\,\{\hat{\boldsymbol{s}}_n(i)\}}{\mathbb{1} + \epsilon^{-1}|\,\hat{\boldsymbol{s}}_n(i)|}\,. \tag{3.25b}$$

3.4 Algorithm analysis

The centralized problem is analyzed in this section, since its local counterpart follows the same train of thought. No regularization is taken into consideration in the analysis, due to the large number of possible regularization functions. The delicate case of an ℓ_1-norm constraint, which is the base for the ZA regularization, is analyzed in [Chen et al., 2016]. Consider the centralized conditional cost function, without regularization:

$$J(\boldsymbol{S}) = \frac{1}{2}\mathbb{E}\left\{\left\|\bar{\boldsymbol{y}}(i) - \boldsymbol{S}\boldsymbol{z}(i)\right\|^2 \,\middle|\, \hat{\boldsymbol{S}}_k, k = 0, \ldots, K-2\right\}\,, \tag{3.26}$$

with:

$$\boldsymbol{z}(i) = \sum_{k=0}^{K-2}\hat{\boldsymbol{S}}_k\boldsymbol{x}(i-k-1)\,. \tag{3.27}$$

Its gradient with respect to (w.r.t.) $\boldsymbol{S}$ is then:

$$\nabla_{\boldsymbol{S}}J(\boldsymbol{S}) = \mathbb{E}\left\{-\bar{\boldsymbol{y}}(i)\boldsymbol{z}^\top(i) + \boldsymbol{S}\boldsymbol{z}(i)\boldsymbol{z}^\top(i)\right\}\,, \tag{3.28}$$

which leads to:

$$\begin{aligned}\nabla_{\boldsymbol{S}}J(\boldsymbol{S}) = 0 &\iff \boldsymbol{S}\mathbb{E}\left\{\boldsymbol{z}(i)\boldsymbol{z}^\top(i)\right\} = \mathbb{E}\left\{\bar{\boldsymbol{y}}(i)\boldsymbol{z}^\top(i)\right\}\\ &\iff \boldsymbol{S}^* = \boldsymbol{R}_{yz}\boldsymbol{R}_z^{-1}\,,\end{aligned} \tag{3.29}$$

with $\boldsymbol{R}_{yz} \triangleq \mathbb{E}\left\{\bar{\boldsymbol{y}}(i)\boldsymbol{z}^\top(i)\right\}$, $\boldsymbol{R}_z \triangleq \mathbb{E}\left\{\boldsymbol{z}(i)\boldsymbol{z}^\top(i)\right\}$, and $\boldsymbol{S}^*$ representing the optimal value which minimizes cost function (3.26). Recall that $\boldsymbol{S}$ does not allow self-loops.

3.4.1 Weight error recursion

Let $\hat{\boldsymbol{S}}_{(i)}$ denote the estimate at step i of $\boldsymbol{S}$. The classic gradient descent step is:

$$\hat{\boldsymbol{S}}_{(i+1)} = \hat{\boldsymbol{S}}_{(i)} - \mu \nabla_{\boldsymbol{S}} J(\boldsymbol{S}) \,, \tag{3.30}$$

where μ is a small enough step size. Using this alongside relation (3.28), we obtain the gradient step:

$$\hat{\boldsymbol{S}}_{(i+1)} = \hat{\boldsymbol{S}}_{(i)} + \mu \left(\boldsymbol{R}_{yz} - \hat{\boldsymbol{S}}_{(i)} \boldsymbol{R}_z \right) \,, \tag{3.31}$$

with its stochastic variant:

$$\hat{\boldsymbol{S}}_{(i+1)} = \hat{\boldsymbol{S}}_{(i)} + \mu \left(\bar{\boldsymbol{y}}(i) - \hat{\boldsymbol{S}}_{(i)} \boldsymbol{z}(i) \right) \boldsymbol{z}^\top(i) \,. \tag{3.32}$$

Let us denote by $\boldsymbol{D}_{(i)} \triangleq \hat{\boldsymbol{S}}_{(i)} - \boldsymbol{S}^*$ the difference between the current available estimate and the optimal solution (3.29).

Assumption 3.1. *Assume regressors $\boldsymbol{z}(i)$ arise from a random process, temporally white, and independent of $\boldsymbol{D}_{(i)}$.*

We can now write:

$$\bar{\boldsymbol{y}}(i) - \boldsymbol{S}_{(i)} \boldsymbol{z}(i) = \boldsymbol{v}(i) - \boldsymbol{D}_{(i)} \boldsymbol{z}(i) \,. \tag{3.33}$$

Subtracting $\boldsymbol{S}^*$ from both sides of (3.32) and using (3.33), we obtain:

$$\begin{aligned} & \hat{\boldsymbol{S}}_{(i+1)} - \boldsymbol{S}^* = \hat{\boldsymbol{S}}_{(i)} - \boldsymbol{S}^* + \mu \left(\boldsymbol{v}(i) - \boldsymbol{D}_{(i)} \boldsymbol{z}(i) \right) \boldsymbol{z}^\top(i) \\ \iff & \boldsymbol{D}_{(i+1)} = \boldsymbol{D}_{(i)} + \mu \boldsymbol{v}(i) \boldsymbol{z}^\top(i) - \mu \boldsymbol{D}_{(i)} \boldsymbol{z}(i) \boldsymbol{z}^\top(i) \\ \iff & \boldsymbol{D}_{(i+1)} = \boldsymbol{D}_{(i)} \left(\boldsymbol{I} - \mu \boldsymbol{z}(i) \boldsymbol{z}^\top(i) \right) + \mu \boldsymbol{v}(i) \boldsymbol{z}^\top(i) \,. \end{aligned} \tag{3.34}$$

The analysis is hereafter split in two main parts: the mean and the mean error behaviors. The goal of this analysis is three-fold: first, it establishes stability conditions for the algorithm, i.e., the conditions needed for its convergence in the mean; second, it offers the possibility to fix the step-size μ in accordance with the needs of the considered application and the desired mean square performance; third, it predicts the evolution of the estimated quantity as a function of the time instant i.

3.4.2 Mean error behavior

We now aim to analyze the evolution of the estimate $\hat{\boldsymbol{S}}_{(i)}$ of $\boldsymbol{S}$ in relation with the optimal solution (3.29), for each instant i. Let us first introduce the notations:

$$\boldsymbol{B}_i \triangleq \boldsymbol{I} - \mu \boldsymbol{z}(i) \boldsymbol{z}^\top(i) \,, \quad \boldsymbol{G}_i \triangleq \mu \boldsymbol{v}(i) \boldsymbol{z}^\top(i) \,. \tag{3.35}$$

Applying the expected value operator on both sides of recursion (3.34), we obtain:

$$\mathbb{E} \left\{ \boldsymbol{D}_{(i+1)} \right\} = \mathbb{E} \left\{ \boldsymbol{D}_{(i)} \right\} \boldsymbol{B} + \boldsymbol{G} \,, \tag{3.36}$$

where $\boldsymbol{B} = \mathbb{E}\{\boldsymbol{B}_i\}$ and $\boldsymbol{G} = \mathbb{E}\{\boldsymbol{G}_i\}$. It is useful to note that $\boldsymbol{G} = \boldsymbol{0}$, since the noise $\boldsymbol{v}(i)$ is i.i.d. and zero-mean.

For stability reasons, we have that [Sayed, 2008]:

$$\rho(\boldsymbol{B}) < 1 \iff 0 < \mu < \frac{2}{\lambda_{\max}(\boldsymbol{R}_z)}\,. \tag{3.37}$$

Should the stability condition (3.37) be respected, it follows that:

$$\lim_{i\to\infty} \mathbb{E}\left\{\boldsymbol{D}_{(i)}\right\} = \boldsymbol{0}\,. \tag{3.38}$$

3.4.3 Mean square error behavior

We now analyze the behavior of the mean square error in the estimate of $\boldsymbol{S}$. Using recursion (3.34) and notations (3.35), we have:

$$\boldsymbol{D}_{(i+1)} = \boldsymbol{D}_{(i)}\boldsymbol{B}_i + \boldsymbol{G}_i\,. \tag{3.39}$$

Thus, we are interested in computing:

$$\mathbb{E}\left\{\boldsymbol{D}_{(i+1)}\boldsymbol{D}_{(i+1)}^\top\right\} = \mathbb{E}\left\{\boldsymbol{D}_{(i)}\boldsymbol{B}_i\boldsymbol{B}_i^\top\boldsymbol{D}_{(i)}^\top\right\} + \mathbb{E}\left\{\boldsymbol{G}_i\boldsymbol{G}_i^\top\right\} + 2\mathbb{E}\left\{\boldsymbol{D}_{(i)}\boldsymbol{B}_i\boldsymbol{G}_i^\top\right\}\,. \tag{3.40}$$

We remark upon the fact that the last term on the r.h.s. of (3.40) is $\boldsymbol{0}$, due to the properties of the noise $\boldsymbol{v}(i)$. We also denote $\boldsymbol{B}_i\boldsymbol{B}_i^\top \triangleq \boldsymbol{K}_i$ and $\mathbb{E}\{\boldsymbol{K}_i\} = \boldsymbol{K} \triangleq \boldsymbol{B}\boldsymbol{B}^\top$. Applying the trace operator on both sides of (3.40) and using its properties, we obtain:

$$\mathbb{E}\left\{\|\boldsymbol{D}_{(i+1)}\|_{\mathrm{F}}^2\right\} = \mathbb{E}\left\{\|\boldsymbol{D}_{(i)}\|_{\mathrm{F},\boldsymbol{K}}^2\right\} + \mu^2\mathrm{Tr}\left\{\boldsymbol{R}_v\boldsymbol{R}_z\right\}\,, \tag{3.41}$$

where $\|\boldsymbol{D}_{(i)}\|_{\mathrm{F},\boldsymbol{K}}^2$ is the Frobenius norm by the metric $\boldsymbol{K}$, i.e., $\|\boldsymbol{D}_{(i)}\|_{\mathrm{F},\boldsymbol{K}}^2 = \mathrm{Tr}\left\{\boldsymbol{D}_{(i)}\boldsymbol{K}\boldsymbol{D}_{(i)}^\top\right\}$. We obtain that, if μ is small enough, the algorithm is mean-square stable as $i \to \infty$, since $\rho(\boldsymbol{K}^i) \to 0$, and converges towards:

$$\lim_{i\to\infty} \mathbb{E}\left\{\|\boldsymbol{D}_{(i)}\|_{\mathrm{F}}^2\right\} = \mu^2\frac{\mathrm{Tr}\left\{\boldsymbol{R}_v\boldsymbol{R}_z\right\}}{\mathrm{Tr}\left\{\boldsymbol{I}-\boldsymbol{K}\right\}}\,, \tag{3.42}$$

which represents the steady-state Mean Square Deviation (MSD) of our algorithm.

We now iterate from $i = 0$:

$$\mathbb{E}\left\{\|\boldsymbol{D}_{(i+1)}\|_{\mathrm{F}}^2\right\} = \mathbb{E}\left\{\|\boldsymbol{D}_{(0)}\|_{\mathrm{F},\boldsymbol{K}^{i+1}}^2\right\} + \mu^2\sum_{j=0}^{i}\mathrm{Tr}\left\{\boldsymbol{R}_v\boldsymbol{R}_z\boldsymbol{K}^j\right\}\,, \tag{3.43}$$

where $\boldsymbol{D}_{(0)}$ represents the initial error. This relation, alongside recursion (3.41), allows the introduction of the learning curve $\zeta(i) \triangleq \mathbb{E}\left\{\|\boldsymbol{D}_{(i)}\|_{\mathrm{F}}^2\right\}$:

$$\zeta(i+1) = \zeta(i) + \mathbb{E}\left\{\|\boldsymbol{D}_{(0)}\|_{\mathrm{F},\boldsymbol{K}^i(\boldsymbol{K}-\boldsymbol{I})}^2\right\} + \mu^2\mathrm{Tr}\left\{\boldsymbol{R}_v\boldsymbol{R}_z\boldsymbol{K}^i\right\}\,. \tag{3.44}$$

3.5 Theoretical validation and experimental results

Setting: An undirected community graph was generated using GSPBOX [Perraudin et al., 2014], with N nodes forming two clusters, i.e., communities. The corresponding adjacency matrix is shown in Fig. 3.3a. The ground truth GSO $\boldsymbol{S}$ was chosen to be:

$$\boldsymbol{S} = \frac{\boldsymbol{W}}{1.1 \cdot \lambda_{\max}(\boldsymbol{W})}, \tag{3.45}$$

which represents the normalized weighted adjacency matrix. The entries of $\boldsymbol{W}$ were set to:

$$w_{nm} = \exp(-\gamma \left\| \boldsymbol{c}_n - \boldsymbol{c}_m \right\|^2), \tag{3.46}$$

where $\gamma \in [0, 1]$ and $\boldsymbol{c}_n$ are the coordinates of the 2D embedding for node n. Changing the parameter γ in (3.46) during our experiments allows us to simulate a sudden change in the networks' underlying connectivity.

We considered an i.i.d. zero-mean Gaussian signal $\boldsymbol{x}(i)$ with covariance matrix $\boldsymbol{R}_x$. To reaffirm our ideas, this matrix was chosen to be the solution of the discrete Lyapunov equation $\boldsymbol{S}\boldsymbol{R}_x\boldsymbol{S}^\top - \boldsymbol{R}_x + \boldsymbol{I} = \boldsymbol{0}$, which can be computed using $\text{vec}\{\boldsymbol{R}_x\} = (\boldsymbol{I} - \boldsymbol{S} \otimes \boldsymbol{S})^{-1} \text{vec}\{\boldsymbol{I}\}$ [Kitagawa, 1977]. We note that this choice is arbitrary, as any positive matrix $\boldsymbol{R}_x$ is eligible. Noise $\boldsymbol{v}(i)$ was also zero-mean Gaussian with covariance matrix $\boldsymbol{R}_v = \text{diag}\left\{\{\sigma_{v,n}^2\}_{n=1}^N\right\}$. Variances $\sigma_{v,n}^2$ were generated from the uniform distribution $\mathfrak{U}(0.1, 0.15)$. We set the filter order to $K = 3$. Output data $\boldsymbol{y}(i)$ were generated with model (3.4). We used a constant step-size μ for all nodes. The theoretical validation was ran over 100 Monte-Carlo runs. For each of the other experiments, estimates were averaged over 50 Monte-Carlo runs.

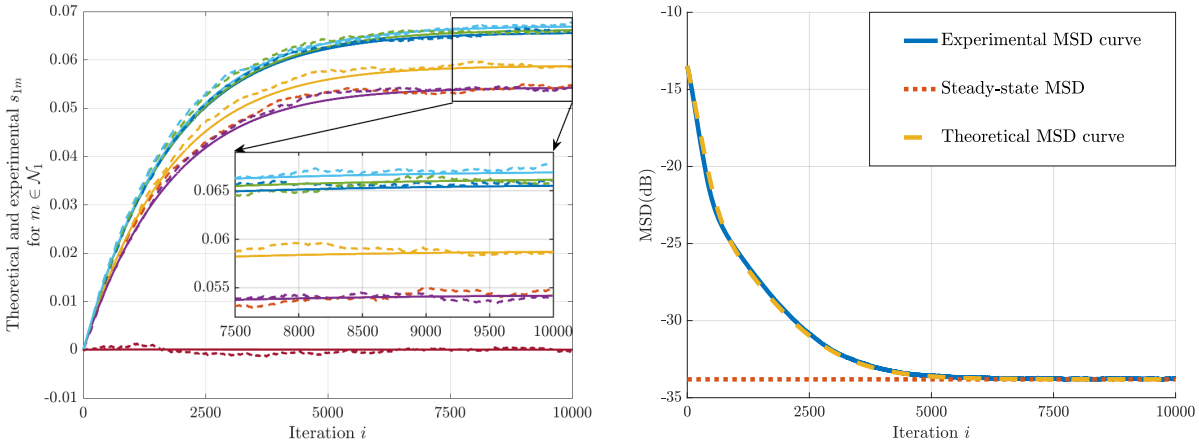
3.5.1 Theoretical validation

The step-size used was $\mu_n = 10^{-4}$ for all nodes. Parameter γ was equal to 0.1. No regularization term was used. Due to its dependency on quantities $\hat{\boldsymbol{S}}_k$, matrix $\boldsymbol{R}_{z_n}$ was re-computed at every iteration, via relation:

$$\begin{aligned}\boldsymbol{R}_{z_n} &\triangleq \mathbb{E}\left\{\boldsymbol{z}_n(i)\boldsymbol{z}_n^\top(i)\right\} \overset{(3.27)}{=} \mathbb{E}\left\{\sum_{k=0}^{K-2}\left[\hat{\boldsymbol{S}}_k\boldsymbol{x}(i-k-1)\right]_{m\in\mathcal{N}_{\setminus n}}\sum_{\ell=0}^{K-2}\left[\boldsymbol{x}^\top(i-\ell-1)\hat{\boldsymbol{S}}_\ell^\top\right]_{m\in\mathcal{N}_{\setminus n}}\right\} \\ &= \sum_{k=0}^{K-2}\sum_{\ell=0}^{K-2}\left[\hat{\boldsymbol{S}}_k\boldsymbol{R}_x\hat{\boldsymbol{S}}_\ell^\top\right]_{m\in\mathcal{N}_{\setminus n}}. \end{aligned} \tag{3.47}$$

The mean error analysis was done locally, at node $n = 1$. Fig. 3.2a depicts the estimated non-zero entries on the first row of $\boldsymbol{S}$. These curves correspond to the mean sense analysis, and are obtained via (3.36).

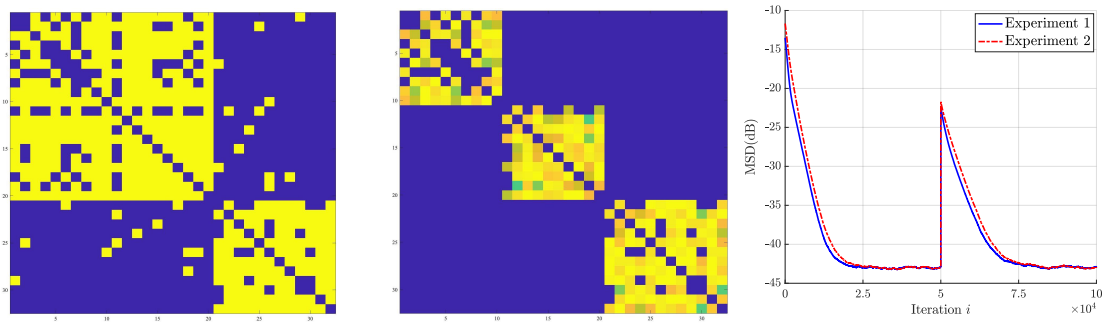
The mean square error analysis was conducted globally. Fig. 3.2b depicts the MSD. The theoretical curve used relation (3.44), while the steady-state MSD was computed using (3.42). The experimental MSD was evaluated using (3.48). We observe that the theoretical curves match well the experimental ones, thus confirming the validity of our model and theoretical analysis. This observation is valid for the behavior of both the mean and mean square error.



(a) Theoretical and experimental entries of $\boldsymbol{S}$. Continuous lines are the theoretical curves, while dashed line are experimental. Corresponding curves are color coded

(b) Experimental, steady-state, and theoretical MSD

Figure 3.2: Validation for the analysis in both the mean and mean square sense



(a) Adjacency matrix $\boldsymbol{A}$

(b) Shift matrix $\boldsymbol{S}'$

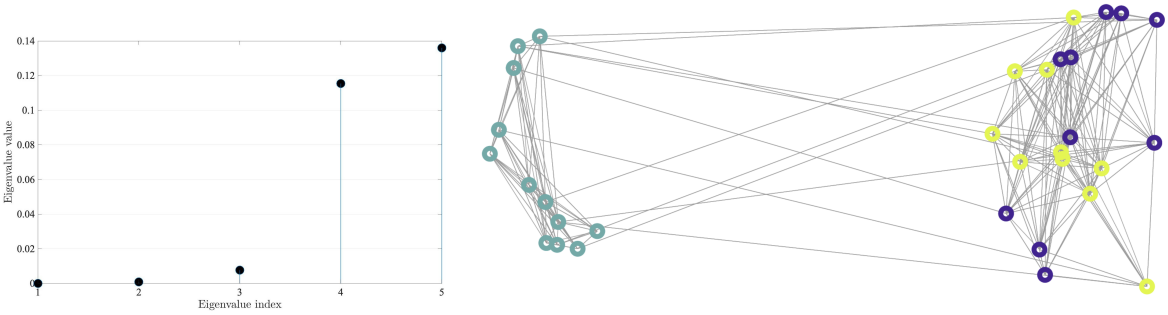
(c) MSD learning curves

Figure 3.3: Adjacency matrix considered for Experiments 1 and 2, shift matrix $\boldsymbol{S}'$ used in Experiment 2 , and the MSD learning curves

3.5.2 Experimental results

Experiment 1: Learning Algorithm 1 was run in order to estimate the GSO $\boldsymbol{S}$ for a network with $N = 32$ nodes. The symmetry-enforcing regularizer (3.13) was used, alongside updates (3.20). In order to illustrate the adaptation abilities of the method, we changed the shift operator mid-way during the experiment. As such, parameter γ in (3.46) was set to 0.1 during the first part of the experiment, and then changed to 0.6 for the second part in order, in order to simulate a sudden change in topology. Parameters η_n were set to 300, $\forall n \in \mathcal{N}$. The estimated MSD learning curve, defined as:

$$\text{MSD}(i) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}\left\{ \|\hat{\boldsymbol{s}}_n(i) - \boldsymbol{s}_n\|^2 \right\} , \tag{3.48}$$



(a) The eigengap

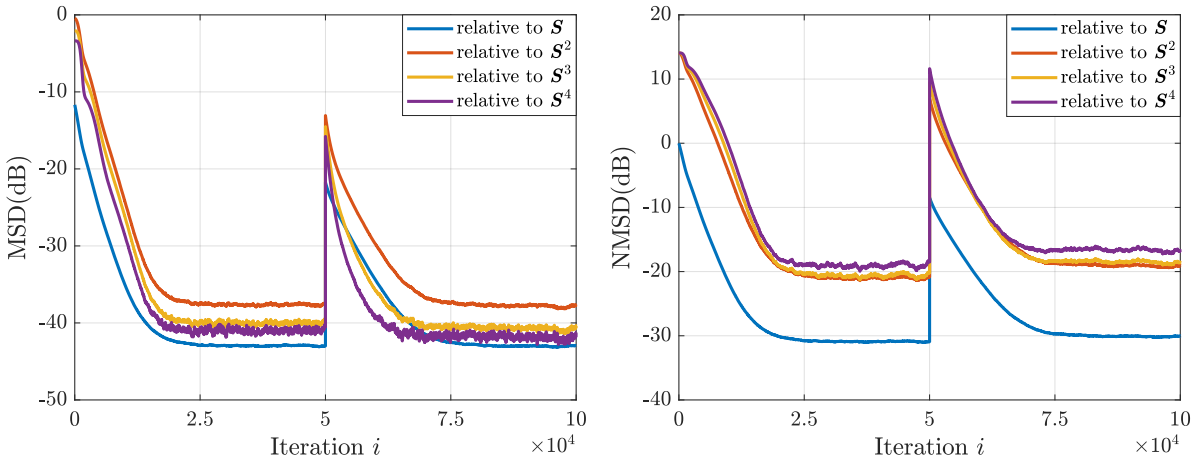
(b) The recovered clusters

Figure 3.4: Spectral clustering performed during Experiment 2. Two communities can be observed in the graph topology. When taking into consideration how the agents actually interact, three clusters are identified

is depicted in Fig. 3.3c. It shows that the algorithm converged monotonically to a reasonably low MSD, and succeeded in adapting to the change in $\boldsymbol{S}$ at time $i = 50000$.

Experiment 2: Data were generated as in Experiment 1, with a new shift matrix $\boldsymbol{S}'$ such that $\text{supp}\{\boldsymbol{S}'\} \subseteq \text{supp}\{\boldsymbol{S}\}$. This allowed us to consider a new setting where, even if a node m is linked to a node n, i.e., $(m, n) \in \mathcal{E}$, the output signal $y_n(i)$ at node n does not necessarily depend on the input signal $x_m(i-1)$ at node m via data model (3.4). To design $\boldsymbol{S}'$, we selected a subset of nodes in one of the two communities of the initial community graph, and we divided their connection weights in $\boldsymbol{S}$ with all other nodes by 10^2. The resulting shift matrix $\boldsymbol{S}'$ is depicted in Fig. 3.3b. In this way, we obtained two clusters according to the adjacency matrix $\boldsymbol{A}$, and three clusters according to the shift matrix $\boldsymbol{S}'$. Mid-way through the experiment, parameter γ in (3.46) was changed from 0.1 to 0.6 in order to simulate a change in topology.

Learning Algorithm 1 was ran in order to estimate $\boldsymbol{S}'$. As in the previous experiment, the symmetry-enforcing regularizer (3.13) was employed, with $\eta_n = 300, \forall n \in \mathcal{N}$, warranting the use of updates (3.20). The learning curve represented in Fig. 3.3c shows that the algorithm converged to a reasonably low MSD, at a slower rate than in Experiment 1 possibly because of the larger number of clusters. To check this assumption, we computed the eigen-decomposition of the estimated shift matrix to infer the number of clusters [Tremblay et al., 2016]. It was numerically found to be equal to 3. Finally, we performed a spectral clustering of the nodes with a k-means algorithm based on the first $k = 3$ eigenvectors. The result depicted in Fig. 3.4b is in accordance with the experimental setup. The aim of this experiments was to check if the obtained estimated topology, under the form of a weighted adjacency matrix, is useful for a post-processing algorithm, in this case a clustering one. This algorithm has numerous applications, across multiple fields, e.g., image segmentation [Wu and Leahy, 1993, Shi and Malik, 2000] or classification [Bengio et al., 2004]. As such, the estimated GSO managed to shed light on how actually groups of node communicate among themselves, even if connections exist that could



(a) Performance in terms of MSD

(b) Performance in terms of NMSD

Figure 3.5: MSD and NMSD curves showcasing how $\hat{\boldsymbol{S}}_k$ approach $\boldsymbol{S}^k$, $\forall k = 1, \ldots, 4$

facilitate an increase in communication capacities. This behavior arises in the right cluster, where two groups are formed, the yellow and the blue. Even if the two are well connected, the concerned agents tend to mostly influence and be influenced by other agents of the same color.

Experiment 3: The goal of this experiment is to experimentally show that, as $i \to \infty$, approximations $\hat{\boldsymbol{S}}_k$ approach $\boldsymbol{S}^k$. Algorithm 1 was run under the same conditions as in Experiment 1, except for the filter order which was set to $K = 5$. This was done in order to obtain approximations of higher order powers in $\boldsymbol{S}$, namely up to $\boldsymbol{S}^4$. The performance was quantified in terms and MSD and Normalized Mean Square Deviation (NMSD). The former is defined in (3.48), while the latter is defined as:

$$\text{NMSD}(i) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E} \left\{ \frac{\left\| \left[\hat{\boldsymbol{S}}_k(i) \right]_{n,\bullet} - \left[\boldsymbol{S}^k \right]_{n,\bullet} \right\|^2}{\left\| \left[\boldsymbol{S}^k \right]_{n,\bullet} \right\|^2} \right\}, \quad k = 1, \ldots, K-1 \,. \tag{3.49}$$

These metrics are depicted in Fig. 3.5. For the particular case of power $k > 1$ when $\rho(\boldsymbol{S}^k)$ decreases as k increases, the Normalized Mean Square Deviation is a better adapted metric to depict the performance. As in the previous experiments, the adaptive capabilities of to proposed method are showcased via the change in γ, present in (3.46), from 0.1 to 0.6.

Experiment 4: Comparisons were conducted with the centralized batch algorithm derived in [Mei and Moura, 2017], called benchmark algorithm (BA). We considered the same experimental setup as in Experiment 1, except that the number of nodes was set to $N = 20$, and each cluster now has 10 nodes. No regularization term $\psi(\cdot)$ was used. Since it deals with a more complex polynomial model than our algorithm, we simplified the BA model by setting its extra

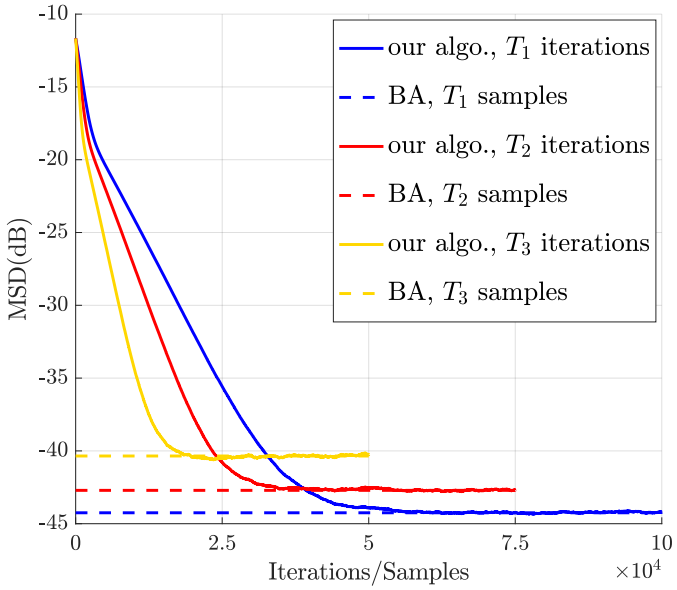
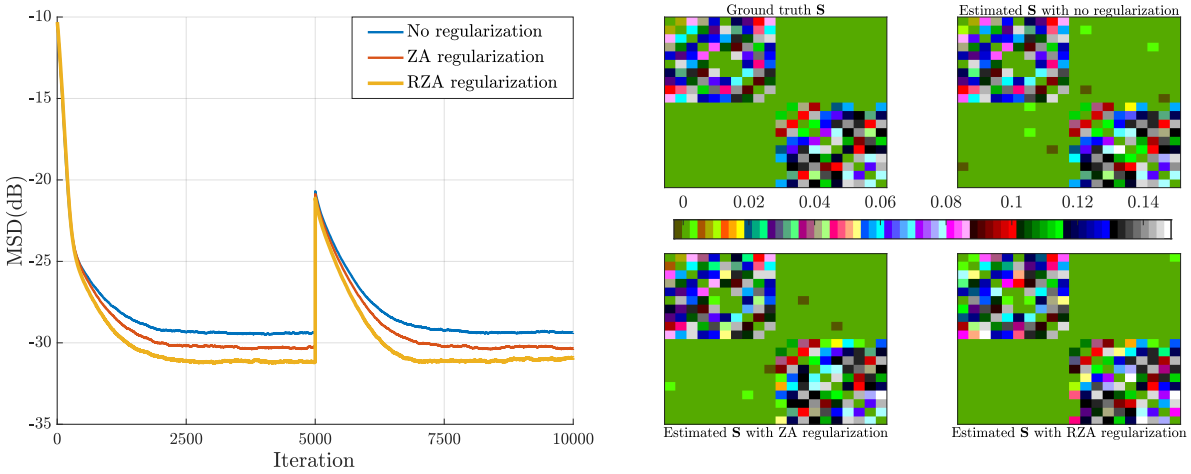


Figure 3.6: Comparison of our algorithm with BA for 3 training set sizes: $T_1 = 10^5, T_2 = 7.5 \cdot 10^4$ and $T_3 = 5 \cdot 10^4$ samples

coefficients to 0. As BA is a batch-mode algorithm which estimates model parameters from a batch of training data, we successively set the size of the training set to $T_1 = 10^5$, $T_2 = 7.5 \cdot 10^4$, and $T_3 = 5 \cdot 10^4$ samples, and ran the BA for each of the three setups. In each case, parameters of BA were set to achieve the best possible MSD. Next, we set the step-sizes μ_n of our algorithm to achieve the same MSD at steady-state as BA, by using relation (3.42). The results are presented in Fig. 3.6. We observe that our algorithm was able to achieve the same MSD after processing approximately half of the training samples. From a computational point of view, note that our method needs to process every sample only once, whereas the BA processes the whole training set many more times, depending on the chosen solver.

Experiment 5: Learning Algorithm 1 was run in order to estimate $\boldsymbol{S}$, depicted in Fig. 3.7b, which is the shift operator for a graph with $N = 20$ nodes split into two clusters of 10 nodes. Three cases have been showcased, namely with *(i)* $\eta_n = 0, \forall n \in \mathcal{N}$, i.e., without regularization, *(ii)* with ZA regularization, and *(iii)* with RZA regularization. Imposing a regularization based on the ℓ_1-norm leads to obtaining a good estimate for the topology of the graph of supp$\{\boldsymbol{S}\}$, which is a proxy for the binary adjacency matrix $\boldsymbol{A}$. This, in turn, translates into an improvement of MSD performance. The learning MSD curves are presented in Fig. 3.7a. They show how the algorithm converges monotonically towards a reasonably low steady-state MSD. Moreover, it also succeeded in adapting to the change of the topology encoded in $\boldsymbol{S}$ at instant $i = 5000$, change obtained by switching the parameter γ in (3.46) from 0.1 to 0.6, simulating a mid-way switch in the GSO.



(a) MSD curves

(b) Shift matrix and its estimates

Figure 3.7: The MSD curves are depicted in 3.7a. Shift matrix $\boldsymbol{S}$ and its estimates obtained via the different considered methods are shown in 3.7b

Visually, Fig. 3.7b showcases how the RZA regularizer is able to offer a better estimate of graph shift operator $\boldsymbol{S}$. Since the log-sum penalty it employs behaves more like an ℓ_0-norm than an ℓ_1-norm [Candes et al., 2008], it avoids introducing bias towards already large values, behavior exhibited by the ZA regularizer [Chen et al., 2009].

3.6 Conclusion

In this chapter, we defined and introduced a distributed and online strategy for topology identification based solely on measured graph signals. This framework allows to estimate a graph shift operator, under the form of a weighted adjacency matrix, based on local one-hop computations. The distributed approach allows not only for the division of the computational burden, but for the enhancement of privacy and security as well. While most of state of the art topology inference algorithms work in a batch mode, the developed online approach allows the algorithm to adapt to changes in the graph shift operator, rendering it able to track dynamic topologies.

Chapter 4 continues to develop the directions set in the current chapter, namely the online and distributed framework. A kernel-based framework is devised, which leads to an algorithm able to cope with nonlinear relationships between the agents of a network.

Chapter 4

Topology inference with nonlinear dependencies: Additive model

Contents

This chapter builds upon the previous one and proposes a method of estimating in an online and adaptive manner a network structure capturing the nonlinear dependencies among streaming graph signals. As such, a new data model is introduced. The estimated structure is presented under the form of a possibly directed adjacency matrix.

The online aspect has rarely been taken into consideration in the existing literature on nonlinear dependencies, and remains one of the main focuses of our work, alongside the capacities to

conduct distributed computations. A local problem is stated, followed by the introduction of Reproducing Kernel Hilbert Spaces. These spaces represent the tool which allows for the modeling of nonlinearities. By projecting data into a higher- or even infinite-dimensional space, we focus on capturing nonlinear relationships between agents. In order to mitigate the increasing number of data points, kernel dictionaries are employed. We follow up with an analysis of the proposed algorithm, before presenting a set of tests ran on both synthetic and real biomedical data. On the former, we obtain reasonable performance, while on the latter the results are comparable with those of state of the art methods and are supported by previous findings in medical literature.

The work presented in this chapter was published in:

- M. Moscu, R. Borsoi, and C. Richard. Online graph topology inference with kernels for brain connectivity estimation. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2020a

- M. Moscu, R. Borsoi, and C. Richard. Convergence analysis of the graph-topology-inference kernel LMS algorithm. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021a

4.1 Introduction

Graphs represent powerful mathematical objects able to model and analyze any kind of network. Due to their inherently distributed nature, they are also suited for big data analysis, since a distributed solution can be easily applied on such an object. Nonlinear interactions between agents appear in applications such as gene regulation systems [The International HapMap Consortium et al., 2007], socio-economical interactions [Heiberger, 2018], or brain activity [Kramer et al., 2008]. When further processing data in contexts such as these, information about network structure is of utmost importance, as motivated in section 2.2. Among graph topology inference methods, most assume linear dependencies between the agents (genes in a network, sectors of a market economy, regions of the brain). Therefore, here resides the need of developing algorithms capable of modeling nonlinear relationships, which are naturally present in real-world applications. Using the properties of reproducing kernels and their ability to model nonlinear relationships between nodal signals, we develop an algorithm capable of estimating the topology of a network, in an online, adaptive, and distributed framework.

Throughout the present chapter, we consider a setting where online nodal measurements are acquired and used in order to infer the topology of the underlying network. In the developed approach, the goal is to model nonlinear dependencies, via an online and adaptive algorithm, capable of tracking changes in the network structure. To that, we add the capability of distributivity over the different agents. The developed method estimates a possibly directed adjacency matrix. The block ℓ_1-norm regularization sees extensive use in this chapter, given its block

sparsity-inducing properties. This is motivated by real-word examples where edge sparsity is present, such as social graphs.

To the best of our knowledge, a kernel-based online solution to the topology inference problem has not yet been considered. As such, we propose an online approach that can sensibly reduce computational stress, as well as adapt to slow changes in the topology that occur in dynamic environments. For the particular case of brain connectivity estimation, the developed method has the advantage of adaptability concerning data availability: due to its online nature, the data acquisition process can be stopped exactly when the desired estimate is obtained. This, in turn, can render the medical process of signal acquisition less strenuous for both the patient and medical personnel.

An early work on graph inference based on nonlinear modeling is [Vert and Yamanishi, 2005]. The method takes into account nodal measurements, but also considers that a subset of the set of edges $\mathcal{E}$ is known beforehand.

One of the recent developments is introduced in [Shen et al., 2017]. The authors propose a batch optimization problem with a regularization term that promotes sparsity in the solution. The method is, however, lacking in adaptability and can prove to be computationally costly. It shows, however, that leveraging kernels in modeling nonlinear connections and feasible, and their results on real data provide insights on brain connectivity.

Along the current chapter, a focus is represented by the inference of functional connectivity of different brain regions, inferred from measures of brain activity. Works such as [Sporns, 2010, Zalesky et al., 2010] propose anatomically-motivated networks, where different chosen areas are linked with the others depending on the estimated number of axonal connections. The field of neuroscience has been able to show that indeed, different actions and functions are mostly, but not completely, directed from certain brain regions. In particular, experiments have been conducted where the left hemisphere was shown to play a major role in speech and reasoning, and controls the right side of the body, while the right hemisphere processes spatial information, and controls the left side of the body [Dennerll, 1964]. The authors of [Sperry et al., 1969] considered the case of patients which, due to different causes, had a severed *corpus callosum* (a nerve tract which connects the left and right hemispheres of the brain). An interesting case is that of W.J., a former Second World War combatant [Gazzaniga, 2014]. When flashed an image of a square to his right side of his field of vision, he was able to identify the square. When flashed on his left side of his field of vision, W.J. stated he had seen nothing. Experiments such as these, as well as the works on topology inference presented earlier in this section, motivate the work in the current chapter.

A set of symbols used throughout the remainder of this chapter are collected in Table 4.1, while others are defined and used locally.

Table 4.1: List of notations and symbols present in Chapter 4

Symbol	Definition
$\boldsymbol{A}$	Adjacency matrix of a graph
$\mathcal{N}$	Set of nodes of the graph
$\mathcal{N}_n$	Set of nodes in the neighborhood of node n, excluding node n
$\mathcal{N}_{\backslash n}$	Set of all nodes, excluding node n
$J_n(\cdot)$	Local cost function
N	Total number of nodes in the graph
$\mathcal{H}_{\kappa_m}$	A Reproducing Kernel Hilbert Space associated to kernel κ_m
L_m	Time lag pertaining to the influence of node m
$\mathcal{D}_m$	The dictionary of node m
ξ_m	Dictionary admission threshold for node m

4.2 Nonlinear model and distributed problem statement

Consider an N-node graph with adjacency matrix $\boldsymbol{A}$ which models a system such as the brain network [Breakspear, 2017]. In this setting, the brain activity in every considered region 1 through N can be measured at different time instants i, thus obtaining a signal $\boldsymbol{y}(i)$. Each of these regions influences and is influenced by the other regions, and these links are encoded in the matrix $\boldsymbol{A}$. For the particular case of brain connectivity, the existence of nonlinear connections have been reported by studies such as [Freeman, 1979, de Zwart et al., 2009]. This motivates modeling certain systems, such as the brain, with nonlinear connections. With these remarks, we consider the following data model:

$$\boldsymbol{y}(i) = \boldsymbol{A}\boldsymbol{f}(i) + \boldsymbol{v}(i)\,, \tag{4.1}$$

where $\boldsymbol{A}$ is the adjacency matrix of the graph. We recall that the estimated graph does not have self-loops, i.e., $a_{nn} = 0, \forall n \in \mathcal{N}$. This matrix models how entries of $\boldsymbol{f}(i) \triangleq \text{col}\left\{f_m\left(\boldsymbol{y}_{L_m}(i)\right)\right\}_{m=1}^{N}$ influence every node. We consider $f_m : \mathbb{R}^{L_m} \to \mathbb{R}$ a nonlinear function whose argument is $\boldsymbol{y}_{L_m}(i) = [y_m(i), \ldots, y_m(i - L_m + 1)]^\top$. Parameter L_m endows the algorithm with memory by making use of past data. This last characteristic is important in applications such as functional brain topology estimation, where there is a 10–20 ms delay in signal propagation between nodes [Petkoski and Jirsa, 2019]. The signal $\boldsymbol{v}(i)$ models innovation noise. The output at every node m is therefore nonlinearly dependent of all the other signals from the other nodes, including the past. Given nodal measurements $\boldsymbol{y}(i)$ acquired online, the goal is estimating the adjacency matrix $\boldsymbol{A}$.

Under the Least Mean Squares (LMS) criterion, the optimization problem can now be expressed under the form:

$$\begin{aligned} \boldsymbol{A}^* = \underset{\boldsymbol{A}}{\text{argmin}}\ & \frac{1}{2}\mathbb{E}\left\{\left\|\boldsymbol{y}(i) - \boldsymbol{A}\boldsymbol{f}(i)\right\|^2\right\} + \Psi(\boldsymbol{A}) \\ \text{subject to } & a_{nm} \in \{0,1\}\,, a_{nn} = 0, \forall n \in \mathcal{N}, \forall m \in \mathcal{N}_{\backslash n}\,, \end{aligned} \tag{4.2}$$

where $\Psi(\boldsymbol{A})$ is a regularization term to account for some prior knowledge of $\boldsymbol{A}$ such as symmetry or sparsity. The constraint aims at forcing the entries of the adjacency matrix to be binary. A relaxation of the constraint may be employed, by instead enforcing $a_{nm} \in [0, 1]$.

Focusing on a single node n, model (4.1) locally becomes:

$$y_n(i) = \sum_{m \in \mathcal{N}_{\setminus n}} a_{nm} f_m(\boldsymbol{y}_{L_m}(i)) + v_n(i)\,, \tag{4.3}$$

and the local optimization problem is now:

$$\begin{aligned} \boldsymbol{a}_n^* = \underset{\boldsymbol{a}_n}{\operatorname{argmin}}\ & \frac{1}{2}\mathbb{E}\left\{ \left| y_n(i) - \sum_{m \in \mathcal{N}_{\setminus n}} a_{nm} f_m(\boldsymbol{y}_{L_m}(i)) \right|^2 \right\} + \psi(\boldsymbol{a}_n) \\ & \text{subject to } a_{nm} \in \{0, 1\}\,, \end{aligned} \tag{4.4}$$

where $\boldsymbol{a}_n$ is the n^{th} row of $\boldsymbol{A}$, with entries $a_{nm}, m = 1, \ldots, N$. We note that constraint $a_{nn} = 0$ is taken into consideration into the local optimization problem (4.4) through the sum over the indexes $m \in \mathcal{N}_{\setminus n}$.

Problem (4.4) has to be solved based only on local measurements $\boldsymbol{y}(\ell), \ell \leq i$ that are available at a certain time instant i.

4.3 Reproducing Kernel Hilbert Spaces and kernel dictionaries

Several solutions in modeling nonlinearities exist, such as nonlinear and polynomial Structural Equation Models [Jöreskog et al., 1996], as well as function selection from an existing function set [Song et al., 2013]. The focus of the present chapter is on kernel methods, which are able to deal with nonlinearities in classification or regression problems by applying linear algorithms over a high-dimensional representation of the input data on an RKHS $\mathcal{H}_\kappa$ associated with a positive definite reproducing kernel $\kappa(\cdot, \cdot)$.

The reproducing property: We briefly recall the reproducing property in an RKHS. For further details, see [Paulsen and Raghupathi, 2016]. Let $\mathcal{H}_\kappa$ be an RKHS associated with kernel $\kappa(\cdot, \cdot)$ and endowed with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_\kappa}$. Let f be a function, $f \in \mathcal{H}_\kappa$, and $\mathcal{X}$ an input space. Then, evaluating $f(x), x \in \mathcal{X}$ can be performed as follows:

$$f(x) = \langle f, \kappa(\cdot, x) \rangle_{\mathcal{H}_\kappa}\,. \tag{4.5}$$

For the particular case where $f = \kappa(\cdot, y), y \in \mathcal{X}$, we have that:

$$\kappa(x, y) = \langle \kappa(\cdot, x), \kappa(\cdot, y) \rangle_{\mathcal{H}_\kappa}\,, \tag{4.6}$$

meaning that simple kernel evaluations suffice when computing inner products in $\mathcal{H}_\kappa$. This simplifying property is also known as the *kernel trick*.

4.3.1 Formulating the problem in a Reproducing Kernel Hilbert Space

Let us denote $\phi_{nm} = a_{nm} f_m$, which allows us to incorporate the binary variable a_{nm} and turn (4.4) into a problem that is linear in ϕ_{nm}. Assuming that ϕ_{nm} belongs to an RKHS $\mathcal{H}_{\kappa_m}$, for $m \in \mathcal{N}_{\backslash n}$, and approximating the expected value in (4.4) by empirical averages computed over the available measurements for $\ell \leq i$, a non-parametric version of the local optimization problem for node n and time instant i can be written as:

$$\{\phi_{nm}^*\}_{m=1}^N = \underset{\substack{\phi_{nm} \in \mathcal{H}_{\kappa_m} \\ m=1,\ldots,N}}{\operatorname{argmin}} \frac{1}{2i} \sum_{\ell=1}^{i} \mathbb{E} \left\{ \left| y_n(\ell) - \sum_{m \in \mathcal{N}_{\backslash n}} \phi_{nm}(\boldsymbol{y}_{L_m}(\ell)) \right|^2 \right\} + \Psi \left(\|\phi_{n1}\|_{\mathcal{H}_{\kappa_m}}, \ldots, \|\phi_{nN}\|_{\mathcal{H}_{\kappa_m}} \right) , \tag{4.7}$$

where $\Psi : \mathbb{R}^N \to [0, \infty[$ is a regularization functional which promotes sparsity in the underlying adjacency matrix by favoring solutions in which many functions ϕ_{nm} are zero. We assume this function to be separable, i.e., $\Psi \left(\|\phi_{n1}\|_{\mathcal{H}_{\kappa_m}}, \ldots, \|\phi_{nN}\|_{\mathcal{H}_{\kappa_m}} \right) = \sum_{m \in \mathcal{N}} \psi_{\mathcal{H}_{\kappa_m}}(\|\phi_{nm}\|_{\mathcal{H}_{\kappa_m}})$, where $\psi_{\mathcal{H}_{\kappa_m}} : \mathbb{R} \to [0, \infty[$ are non-decreasing functions. Since (4.7) employs a convex loss function, the conditions of the linear representer theorem are satisfied [Schölkopf et al., 2001]. Thus, at instant i, the solution to (4.7) admits a finite-dimensional representation of the form:

$$\phi_{nm}^*(\cdot) = \sum_{p=1}^{i} \alpha_{nmp} \kappa_m \left(\cdot, \boldsymbol{y}_{L_m}(p) \right) , \quad m = 1, \ldots, N \,, m \neq n \,, \tag{4.8}$$

where coefficients $\alpha_{nmp} \in \mathbb{R}$.

Introducing sparsity: An important question is how to introduce sparsity in the graph connections now that the problem is formulated in terms of ϕ_{nm}. Since $\phi_{nm} = a_{nm} f_m$, $a_{nm} = 0$ implies that function $\phi_{nm} \equiv 0$. Thus, promoting sparsity over $\boldsymbol{A}$ is equivalent to promoting sparsity over the functions ϕ_{nm}, for $m \in \mathcal{N}_{\backslash n}$. Fortunately, the coefficient-based representation (4.8) means that this can be performed equivalently by promoting sparsity of groups of variables $\{\alpha_{nmp}\}_{p=1}^i$, for $m \in \mathcal{N}_{\backslash n}$. This can be done very efficiently by using a block-sparse regularization over the coefficients. This leads to the following optimization problem at every time instant i:

$$\boldsymbol{\alpha}_n^* = \underset{\boldsymbol{\alpha}_n}{\operatorname{argmin}} \frac{1}{2i} \sum_{\ell=1}^{i} \mathbb{E} \left\{ \left| y_n(\ell) - \boldsymbol{\alpha}_n^\top \boldsymbol{k}(\ell) \right|^2 \right\} + \eta_n \left\| \boldsymbol{\alpha}_n \right\|_{\mathrm{B},1} , \tag{4.9}$$

where the $Ni \times 1$ block vectors $\boldsymbol{\alpha}_n$ and $\boldsymbol{k}(\ell)$ are defined as:

$$\boldsymbol{\alpha}_n = \left[\tilde{\boldsymbol{\alpha}}_{n1}^\top, \ldots, \tilde{\boldsymbol{\alpha}}_{nN}^\top \right]^\top , \qquad \tilde{\boldsymbol{\alpha}}_{nm} = \operatorname{col} \left\{ \alpha_{nmp} \right\}_{p=1}^i , \forall m \in \mathcal{N}_{\backslash n} \,, \tag{4.10a}$$

$$\boldsymbol{k}(\ell) = \left[\tilde{\boldsymbol{k}}_1^\top(\ell), \ldots, \tilde{\boldsymbol{k}}_N^\top(\ell) \right]^\top , \qquad \tilde{\boldsymbol{k}}_m(\ell) = \operatorname{col} \left\{ \kappa_m(\boldsymbol{y}_{L_m}(\ell), \boldsymbol{y}_{L_m}(p)) \right\}_{p=1}^i , \forall m \in \mathcal{N}_{\backslash n} \,. \tag{4.10b}$$

Constant $\eta_n > 0$ is a regularization parameter. Also, a block sparsity-inducing regularisation on $\boldsymbol{\alpha}_n$ was added through the term $\|\boldsymbol{\alpha}_n\|_{\mathrm{B},1}$, the block-wise ℓ_1-norm, i.e., $\|\boldsymbol{\alpha}_n\|_{\mathrm{B},1} =$

$\sum_{m\in\mathcal{N}_{\setminus n}} \|\tilde{\boldsymbol{\alpha}}_{nm}\|_2$. This norm is known to promote group sparsity [Yuan and Lin, 2006], favoring solutions with entire blocks of variables $\tilde{\boldsymbol{\alpha}}_{nm}$ equal to $\mathbf{0}$, from which it can be inferred that $a_{nm} = 0$ and thus there will be no connection from node m towards n.

It is important to note now that the model allows for data from each node m to exist in their own space, therefore being allotted their own separate kernel $\kappa_m(\cdot,\cdot)$.

4.3.2 Optimization

Solving (4.9) in batch mode is, however, impractical and computationally costly. This is why we propose a stochastic gradient descent-based solution in order to update $\boldsymbol{\alpha}_n$ every instant i. Remark that a sub-gradient of the block-ℓ_1 regularization $\|\boldsymbol{\alpha}_n\|_{\mathrm{B},1} = \sum_{m\in\mathcal{N}_{\setminus n}} \|\tilde{\boldsymbol{\alpha}}_{nm}\|_2$ is given by the block vector $\boldsymbol{\Gamma}_n = [\boldsymbol{\Gamma}_{n1}^\top, \ldots, \boldsymbol{\Gamma}_{nN}^\top]^\top$ [Jin et al., 2018a,b], where each block $\boldsymbol{\Gamma}_{nm}$ is:

$$\boldsymbol{\Gamma}_{nm} = \begin{cases} \dfrac{\tilde{\boldsymbol{\alpha}}_{nm}}{\|\tilde{\boldsymbol{\alpha}}_{mn}\|_2} & \text{if } \|\tilde{\boldsymbol{\alpha}}_{mn}\|_2 \neq 0 \\ \mathbf{0} & \text{if } \|\tilde{\boldsymbol{\alpha}}_{mn}\|_2 = 0 \end{cases} . \tag{4.11}$$

This entails the use of group zero-attracting LMS (GZA-LMS) [Jin et al., 2018a], leading to the following update rule, for every time instant i:

$$\hat{\boldsymbol{\alpha}}_n(i+1) = \hat{\boldsymbol{\alpha}}_n(i) + \mu_n[\boldsymbol{r}_{ky} - \boldsymbol{R}_{kk}\hat{\boldsymbol{\alpha}}_n(i) - \eta_n\boldsymbol{\Gamma}_n(i)] , \tag{4.12}$$

where $\boldsymbol{r}_{ky} = \mathbb{E}\{\boldsymbol{k}(i)y_n(i)\}, \boldsymbol{R}_{kk} = \mathbb{E}\{\tilde{\boldsymbol{k}}(i)\tilde{\boldsymbol{k}}^\top(i)\}$. Estimating these second order moments can however prove to be unattainable or computationally costly. This warrants for the use of approximations, such as those based on instantaneous realizations:

$$\boldsymbol{r}_{ky} \approx y_n(i)\boldsymbol{k}(i), \quad \boldsymbol{R}_{kk} \approx \boldsymbol{k}(i)\boldsymbol{k}^\top(i) . \tag{4.13}$$

The use of (4.13) leads to the stochastic GZA-LMS update:

$$\hat{\boldsymbol{\alpha}}_n(i+1) = \hat{\boldsymbol{\alpha}}_n(i) + \mu_n\boldsymbol{k}(i)[y_n(i) - \boldsymbol{k}^\top(i)\hat{\boldsymbol{\alpha}}_n(i)] - \mu_n\eta_n\boldsymbol{\Gamma}_n(i) . \tag{4.14}$$

4.3.3 Kernel dictionaries

An immediate observation concerning update (4.14) is that the size of $\boldsymbol{k}(i)$ can become prohibitive as i increases, since each acquired measurement increases the number of kernel functions, in accordance with (4.10b). A solution to this problem are kernel dictionaries which admit a new candidate kernel function only if the candidate function passes a certain sparsification rule. Under this framework, each node m in the network creates, updates, and stores a dictionary of kernel functions, $\mathcal{D}_m = \{\kappa_m(\cdot, \boldsymbol{y}_{L_m}(\omega_j)) : \omega_j \in \mathcal{I}_m^i \subset \{1,\ldots,i-1\}\}$, where $\mathcal{I}_m^i$ represents the set of time indices of elements selected for the dictionary, up to time instant i. Multiple dictionary sparsification rules exist in the literature. One of the simplest is the Nyström method [Williams and Seeger, 2001], which is based on random selection of the dictionary elements. The authors

of [Engel et al., 2004] propose the approximate linear dependence (ALD) criterion. When a new sample arrives, the method checks if its corresponding kernel function is approximately linearly dependent of the already selected entries, and added to the dictionary should it bring new information, i.e., be independent of previous elements. A recent work is [Bueno and Silva, 2020], where an improvement of the ALD method is introduced. A fixed-size dictionary based on Fourier features of the kernel drawn from a probability density function is developed in [Bouboulis et al., 2016]. This method has the advantage of allowing the user to set the exact number of entries beforehand.

The chosen method of dictionary sparsification is the coherence criterion [Richard et al., 2009]. Under this framework, a candidate kernel function $\kappa_m(\cdot, y_m(i))$ is added in $\mathcal{D}_m$ if the following sparsification condition holds:

$$\max_{\omega_j \in \mathcal{I}_m^i} |\kappa_m(\boldsymbol{y}_{L_m}(i), \boldsymbol{y}_{L_m}(\omega_j))| \leq \xi_m \,, \tag{4.15}$$

where $\xi_m \in [0, 1[$ determines the level of sparsity and coherence of the dictionary [Richard et al., 2009]. Vector $\boldsymbol{k}(i)$ now only stores functions which satisfy (4.15), leading to a rewriting of (4.10b) under the form:

$$\boldsymbol{k}(\ell) = \left[\tilde{\boldsymbol{k}}_1^\top(\ell), \ldots, \tilde{\boldsymbol{k}}_N^\top(\ell)\right]^\top, \quad \tilde{\boldsymbol{k}}_m(i) = \text{col}\{\kappa_m(\boldsymbol{y}_{L_m}(i), \boldsymbol{y}_{L_m}(\omega_j))\}_{\omega_j \in \mathcal{I}_m^i}, \forall m \in \mathcal{N}_{\backslash n} \,. \tag{4.16}$$

It is worth noting that using this approach, every time one kernel function is added to a $\mathcal{D}_m$, all the blocks $\tilde{\boldsymbol{\alpha}}_{nm}$ increase in size by one new entry, $\forall n \in \mathcal{N}$. Also, at each instant i, $\boldsymbol{\alpha}_n$ and $\boldsymbol{k}(i)$ are of size $\sum_{m \in \mathcal{N}_{\backslash n}} \text{card}\,\{\mathcal{D}_m\} \times 1$. Most importantly, the number of entries in the dictionary tends to stop increasing with the total number of currently available samples, i.e., $\text{card}\,\{\mathcal{D}_m\} < \infty$ when $i \to \infty$ [Richard et al., 2009].

Algorithm 2 summarizes the developed method. In the last step, τ_n acts as an edge identification threshold. This parameter is used in order to identify the topology from the estimated coefficients $\hat{\boldsymbol{\alpha}}_n(i)$, determining whether there exist links from each node $m \in \mathcal{N}_{\backslash n}$ towards n. When processing real data, due to a lack of a ground truth matrix, τ_n can be set as to obtain an estimated topology which realistically explains the studied process, method already successfully applied in works such as [Shen et al., 2019]. Its value can be further adjusted as to obtain a connected graph, i.e., a graph in which there exists a path between any node couple. The execution of the algorithm yields $\mathcal{N}_n$, the neighborhood of node n.

4.4 Algorithm analysis

The distributed problem is analyzed in the current section. The dictionary elements are considered as chosen and set beforehand. Consider the local conditional cost function:

$$J_n(\boldsymbol{\alpha}_n) = \frac{1}{2}\mathbb{E}\left\{ \left| y_n(i) - \boldsymbol{\alpha}_n^\top \boldsymbol{k}(i) \right|^2 \middle| \{\mathcal{D}_m\}_{m \in \mathcal{N}_{\backslash n}} \right\} . \tag{4.17}$$

Algorithm 2: Kernel-based online topology inference

For every node n:
Inputs: Parameters μ_n, η_n, $\kappa_n(\cdot,\cdot)$, ξ_n, and τ_n
Initialization: Set all entries of $\hat{\boldsymbol{\alpha}}_n(0)$ to 0
Algorithm: At each time instant $i \geq 1$
Introduce $\kappa_n(\cdot, y_n(i))$ in $\mathcal{D}_n$ if (4.15) is verified
Receive and store the updated dictionaries $\mathcal{D}_m$, $\forall m \neq n$
Compute $\boldsymbol{k}(i)$ defined in (4.16)
Update $\hat{\boldsymbol{\alpha}}_n(i)$ using (3.20)
Set $\hat{a}_{nm}(i)$ to 1 if $\|\hat{\boldsymbol{\alpha}}_{nm}(i)\| \geq \tau_n$, to 0 otherwise

Table 4.2: List of notations and symbols employed throughout the analysis in Chapter 4

Symbol	Equation
$\boldsymbol{\alpha}_n^* = \boldsymbol{R}_{kk}^{-1}\boldsymbol{r}_{ky}$	(4.19)
$\boldsymbol{d}_{(i)} \triangleq \hat{\boldsymbol{\alpha}}_{n(i)} - \boldsymbol{\alpha}_n^*$	(4.23)
$\varepsilon(i) = y_n(i) - \boldsymbol{k}^\top(i)\boldsymbol{d}(i) - \boldsymbol{k}^\top(i)\boldsymbol{\alpha}_n^*$	(4.25)
$J_{n,\min} \triangleq J_n(\boldsymbol{\alpha}_n^*) = \mathbb{E}\left\{y_n^2(i)\right\} - \boldsymbol{r}_{ky}^\top \boldsymbol{R}_{kk}^{-1}\boldsymbol{r}_{ky}$	(4.35)
$\left[\boldsymbol{K}^{(u,v)}\right]_{ab} = \mathbb{E}\left\{[\boldsymbol{k}(i)]_u [\boldsymbol{k}(i)]_v [\boldsymbol{k}(i)]_a [\boldsymbol{k}(i)]_b\right\}$	(4.42)
$[\boldsymbol{Q}(i)]_{uv} = \mathrm{Tr}\left\{\boldsymbol{K}^{(u,v)}\boldsymbol{D}_{(i)}\right\}$	(4.43)
$\boldsymbol{F}_0 = \boldsymbol{I}_2 - \mu(\boldsymbol{I}\otimes\boldsymbol{R}_{kk} + \boldsymbol{R}_{kk}\otimes\boldsymbol{I}) + \mu^2\boldsymbol{F}_1$	(4.48)
$J_{n,\mathrm{MSE}}(\infty) = J_{n,\min} + \mathrm{Tr}\left\{\boldsymbol{R}_{kk}\boldsymbol{D}_{(\infty)}\right\}$	(4.50)

Its corresponding gradient w.r.t. $\boldsymbol{\alpha}_n$ is:

$$\nabla_{\boldsymbol{\alpha}_n} J_n(\boldsymbol{\alpha}_n) = \mathbb{E}\left\{-y_n(i)\boldsymbol{k}(i) + \boldsymbol{k}(i)\boldsymbol{k}^\top(i)\boldsymbol{\alpha}_n\right\}, \tag{4.18}$$

leading to:

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}_n} J(\boldsymbol{\alpha}_n) = 0 &\iff \mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{k}^\top(i)\right\}\boldsymbol{\alpha}_n = \mathbb{E}\left\{y_n(i)\boldsymbol{k}(i)\right\} \\ &\iff \boldsymbol{\alpha}_n^* = \boldsymbol{R}_{kk}^{-1}\boldsymbol{r}_{ky}\,,\end{aligned} \tag{4.19}$$

with $\boldsymbol{R}_{kk} \triangleq \mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{k}^\top(i)\right\}, \boldsymbol{r}_{ky} \triangleq \mathbb{E}\left\{y_n(i)\boldsymbol{k}(i)\right\}$, and $\boldsymbol{\alpha}_n^*$ representing the optimal value which minimizes cost function (4.17).

4.4.1 Weight error recursion

Let $\hat{\boldsymbol{\alpha}}_{n(i)}$ denote the estimate at step i of $\boldsymbol{\alpha}_n$. The classic gradient descent step is:

$$\hat{\boldsymbol{\alpha}}_{n(i+1)} = \hat{\boldsymbol{\alpha}}_{n(i)} - \mu\nabla_{\boldsymbol{\alpha}_n} J_n(\boldsymbol{\alpha}_n)\,, \tag{4.20}$$

where μ is a small enough step size. Replacing the gradient via relation (4.18), we obtain the gradient step:

$$\hat{\boldsymbol{\alpha}}_{n(i+1)} = \hat{\boldsymbol{\alpha}}_{n(i)} + \mu(\boldsymbol{r}_{ky} - \boldsymbol{R}_{kk}\hat{\boldsymbol{\alpha}}_{n(i)})\,, \tag{4.21}$$

with its stochastic variant:

$$\hat{\boldsymbol{\alpha}}_{n(i+1)} = \hat{\boldsymbol{\alpha}}_{n(i)} + \mu \boldsymbol{k}(i)\varepsilon(i)\,, \tag{4.22}$$

where $\varepsilon(i) \triangleq y_n(i) - \boldsymbol{k}^\top(i)\hat{\boldsymbol{\alpha}}_{n(i)}$ represents the instantaneous error.Let us denote the difference between the current available estimate and the optimal solution (4.19) by:

$$\boldsymbol{d}_{(i)} \triangleq \hat{\boldsymbol{\alpha}}_{n(i)} - \boldsymbol{\alpha}_n^*\,. \tag{4.23}$$

Assumption 4.1. *We assume the use of the Gaussian kernel.*

It is defined as:

$$\kappa^{\mathrm{G}}(\boldsymbol{a}, \boldsymbol{b}) = \exp\left(-\frac{\|\boldsymbol{a} - \boldsymbol{b}\|^2}{2\sigma^2}\right)\,. \tag{4.24}$$

This kernel choice is made due to its capacities as an universal approximator [Liu et al., 2010].

Assumption 4.2. *We suppose that dictionaries* $\{\mathcal{D}_m\}_{m\in\mathcal{N}_{\setminus n}}$ *are already set beforehand. Inputs* $\boldsymbol{y}(i)$ *are assumed independent, zero-mean Gaussian random vectors with auto-correlation matrix* $\boldsymbol{R}_{yy} = \mathbb{E}\left\{\boldsymbol{y}(i)\boldsymbol{y}^\top(i)\right\}$.

Assumption 4.3. *Quantity* $\boldsymbol{k}(i)\boldsymbol{k}^\top(i)$ *is statistically independent of the error vector* $\boldsymbol{d}_{(i)}$.

A justification for the feasibility of the latter assumption is presented in [Minkoff, 2001].

Error $\varepsilon(i)$ can be expressed in terms of the error vector $\boldsymbol{d}_{(i)}$:

$$\varepsilon(i) = y_n(i) - \boldsymbol{k}^\top(i)\boldsymbol{d}_{(i)} - \boldsymbol{k}^\top(i)\boldsymbol{\alpha}_n^*\,. \tag{4.25}$$

Replacing (4.25) into relation (4.22) leads to the following error vector recursion:

$$\boldsymbol{d}_{(i+1)} = \boldsymbol{d}_{(i)} + \mu y_n(i)\boldsymbol{k}(i) - \mu \boldsymbol{k}(i)\boldsymbol{k}^\top(i)\boldsymbol{d}_{(i)} - \mu \boldsymbol{k}(i)\boldsymbol{k}^\top(i)\boldsymbol{\alpha}_n^*\,. \tag{4.26}$$

4.4.2 Mean error behavior

Moment-generating function of a Gaussian random variable: Before proceeding to the mean behavior analysis, consider the quadratic form ξ of a Gaussian vector $\boldsymbol{\zeta}$:

$$\xi = \boldsymbol{\zeta}\boldsymbol{B}\boldsymbol{\zeta}^\top + \boldsymbol{b}^\top\boldsymbol{\zeta}\,, \tag{4.27}$$

with $\mathbb{E}\{\boldsymbol{\zeta}\} = \boldsymbol{0}, \boldsymbol{R}_{\zeta\zeta} = \mathbb{E}\left\{\boldsymbol{\zeta}\boldsymbol{\zeta}^\top\right\}$. The moment-generating function of the random variable ξ is [Omura and Kailath, 1965, p. 101]:

$$\begin{aligned}\Psi_\xi(t) &\triangleq \mathbb{E}\{\exp(t\xi)\} \\ &= \det\{\boldsymbol{I} - 2t\boldsymbol{B}\boldsymbol{R}_{\zeta\zeta}\}^{-\frac{1}{2}} \exp\left(\frac{t^2}{2}\boldsymbol{b}^\top\boldsymbol{R}_{\zeta\zeta}\left(\boldsymbol{I} - 2t\boldsymbol{B}\boldsymbol{R}_{\zeta\zeta}\right)^{-1}\boldsymbol{b}\right), \quad t \in \mathbb{R}\,.\end{aligned} \tag{4.28}$$

This result will be useful in the remainder of this subsection.

The goal of the analysis in the mean is to determine the stability conditions of the algorithm, i.e., the conditions in which the algorithm converges in the mean. As such, we start by taking the expectation of relation (4.26) and employing Assumptions 4.3 – 4.4, which leads to the following mean error recursion:

$$\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\} = (\boldsymbol{I} - \mu \boldsymbol{R}_{kk})\,\mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\} . \tag{4.29}$$

Block matrix $\boldsymbol{R}_{kk}$ contains blocks $\boldsymbol{R}_{kk}^{(m_1,m_2)} \triangleq \mathbb{E}\left\{\boldsymbol{k}_{m_1}(i)\boldsymbol{k}_{m_2}^{\top}(i)\right\}, \forall m_1, m_2 \in \mathcal{N}_{\backslash n}$. Each entry $(u, v), u = 1, \ldots, \text{card}\{\mathcal{D}_{m_1}\}, v = 1, \ldots, \text{card}\{\mathcal{D}_{m_2}\}$ of every block $\boldsymbol{R}_{kk}^{(m_1,m_2)}$, for both $m_1 = m_2$ and $m_1 \neq m_2$, is:

$$\begin{aligned}\left[\boldsymbol{R}_{kk}^{(m_1,m_2)}\right]_{uv} =& \mathbb{E}\left\{\exp\left(-\frac{1}{2\sigma^2}\left(\left\|\boldsymbol{y}_{L_{m_1}}(i) - \boldsymbol{y}_{L_{m_1}}(\omega_u)\right\|^2 + \left\|\boldsymbol{y}_{L_{m_2}}(i) - \boldsymbol{y}_{L_{m_2}}(\omega_v)\right\|^2\right)\right)\right\} \\ =& \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{y}^{(uv)}\right\|^2\right)\mathbb{E}\left\{\exp\left(-\frac{1}{\sigma^2}\left(\frac{1}{2}\left\|\boldsymbol{y}^{(ii)}\right\|^2 - \left(\boldsymbol{y}^{(uv)}\right)^{\top}\boldsymbol{y}^{(ii)}\right)\right)\right\} ,\end{aligned} \tag{4.30}$$

where $\boldsymbol{y}^{(ii)} = \begin{bmatrix}\boldsymbol{y}_{L_{m_1}}(i) \\ \boldsymbol{y}_{L_{m_2}}(i)\end{bmatrix}$ and $\boldsymbol{y}^{(uv)} = \begin{bmatrix}\boldsymbol{y}_{L_{m_1}}(\omega_u) \\ \boldsymbol{y}_{L_{m_2}}(\omega_v)\end{bmatrix}$. We now make use of relation (4.28), with $\boldsymbol{B} = \frac{1}{2}\boldsymbol{I}$, $\boldsymbol{b} = -\boldsymbol{y}^{(uv)}$ and $t = -\frac{1}{\sigma^2}$. Thus, we obtain:

$$\begin{aligned}\left[\boldsymbol{R}_{kk}^{(m_1,m_2)}\right]_{uv} =& \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{y}^{(uv)}\right\|^2\right)\det\left\{\boldsymbol{I} + \frac{1}{\sigma^2}\boldsymbol{R}_{yy}^{(m_1m_2)}\right\}^{-\frac{1}{2}} \\ &\times \exp\left(\frac{1}{2\sigma^4}\left(\boldsymbol{y}^{(uv)}\right)^{\top}\boldsymbol{H}^{(m_1m_2)}\boldsymbol{y}^{(uv)}\right) ,\end{aligned} \tag{4.31}$$

where $\boldsymbol{I}$ is of size $(L_{m_1} + L_{m_2}) \times (L_{m_1} + L_{m_2})$, $\boldsymbol{H}^{(m_1m_2)} = \boldsymbol{R}_{yy}^{(m_1m_2)}\left(\boldsymbol{I} + \frac{1}{\sigma^2}\boldsymbol{R}_{yy}^{(m_1m_2)}\right)^{-1}$ and $\boldsymbol{R}_{yy}^{(m_1m_2)} = \begin{bmatrix}[\boldsymbol{R}_{yy}]_{m_1m_1}\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_1}}^{\top} & [\boldsymbol{R}_{yy}]_{m_1m_2}\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_2}}^{\top} \\ [\boldsymbol{R}_{yy}]_{m_2m_1}\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_1}}^{\top} & [\boldsymbol{R}_{yy}]_{m_2m_2}\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_2}}^{\top}\end{bmatrix}$. We note that $\boldsymbol{R}_{yy}^{(m_1m_2)}$ has a sparser structure should $\boldsymbol{y}(i)$ be i.i.d.. Blocks on the main diagonal then become $[\boldsymbol{R}_{yy}]_{m_jm_j}\boldsymbol{I}$, where $\boldsymbol{I}$ is of size $m_j \times m_j$, for $j = 1, 2$. The remaining blocks, which are not necessarily square, are formed similarly, with entries $[\boldsymbol{R}_{yy}]_{m_jm_k}$, $j, k = 1, 2, j \neq k$, on their main diagonals and zero otherwise.

Given Assumption 4.3 and the preset dictionaries $\{\mathcal{D}_m\}_{m\in\mathcal{N}_{\backslash n}}$, the proposed algorithm converges if the following condition holds:

$$0 < \mu < \frac{2}{\lambda_{\max}(\boldsymbol{R}_{kk})} . \tag{4.32}$$

4.4.3 Mean square error behavior

The mean square error analysis allows for the prediction of the behavior of the algorithm under a deterministic model, thus removing the reliance on Monte-Carlo runs. Moreover, it allows for the selection of the step-size μ in accordance with the needs of the considered application.

We start this subsection by defining the optimal estimation error ε_0 as:

$$\varepsilon_0(i) = y_n(i) - \boldsymbol{k}^{\top}(i)\boldsymbol{\alpha}_n^* . \tag{4.33}$$

Assumption 4.4. *We assume that the optimal estimation error ε_0 given by the finite order model is close to the one obtained by the infinite length model, such that $\mathbb{E}\{e_0(i)\} \approx 0$.*

Let us denote $\boldsymbol{D}_{(i)} \triangleq \mathbb{E}\left\{\boldsymbol{d}_{(i)}\boldsymbol{d}_{(i)}^{\top}\right\}$. As such, using (4.26) and Assumptions 4.3 – 4.4, we obtain:

$$\boldsymbol{D}_{(i+1)} = \boldsymbol{D}_{(i)} - \mu\left(\boldsymbol{D}_{(i)}\boldsymbol{R}_{kk} + \boldsymbol{R}_{kk}\boldsymbol{D}_{(i)}\right) + \mu^2\boldsymbol{Q} + \mu^2\boldsymbol{R}_{kk}J_{n,\min}\,, \tag{4.34}$$

with $J_{n,\min}$ being the minimum value of the cost function (4.17), i.e.:

$$J_{n,\min} \triangleq J_n(\boldsymbol{\alpha}_n^*) = \mathbb{E}\left\{y_n^2(i)\right\} - \boldsymbol{r}_{ky}^{\top}\boldsymbol{R}_{kk}^{-1}\boldsymbol{r}_{ky}\,, \tag{4.35}$$

and:

$$\boldsymbol{Q} = \mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{k}^{\top}(i)\boldsymbol{d}_{(i)}\boldsymbol{d}_{(i)}^{\top}\boldsymbol{k}(i)\boldsymbol{k}^{\top}(i)\right\}\,. \tag{4.36}$$

We define the Mean Square Error (MSE) at instant i:

$$J_{n,\mathrm{MSE}}(i) \triangleq \mathbb{E}\left\{\left|y_n(i) - \boldsymbol{k}^{\top}(i)\hat{\boldsymbol{\alpha}}_{n(i)}\right|^2\right\}\,, \tag{4.37}$$

and the MSD at instant i:

$$\mathrm{MSD}(i) \triangleq \mathbb{E}\left\{\left\|\hat{\boldsymbol{\alpha}}_{n(i)} - \boldsymbol{\alpha}_n^*\right\|^2\right\} = \mathbb{E}\left\{\left\|\boldsymbol{d}_{(i)}\right\|^2\right\}\,. \tag{4.38}$$

The second-order weight moments $\boldsymbol{D}_{(i)}$ relate to the MSE via relation [Haykin, 2002, p. 268]:

$$J_{n,\mathrm{MSE}}(i) = J_{n,\min} + \mathrm{Tr}\left\{\boldsymbol{R}_{kk}\boldsymbol{D}_{(i)}\right\}\,, \tag{4.39}$$

and to the MSD through:

$$\mathrm{MSD}(i) = \mathrm{Tr}\left\{\boldsymbol{D}_{(i)}\right\}\,. \tag{4.40}$$

In order to compute these two metrics, the next step in the analysis is evaluating the entries of the matrix $\boldsymbol{Q}$. Let us note $k_D = \sum_{m\in\mathcal{N}_{\setminus n}} \mathrm{card}\{\mathcal{D}_m\}$, the total number of dictionary entries. We make use of Assumption 4.3, leading to the writing of the $(u,v)^{\mathrm{th}}$ entry of $\boldsymbol{Q}$ as:

$$[\boldsymbol{Q}]_{uv} = \sum_{a=1}^{k_D}\sum_{b=1}^{k_D}\mathbb{E}\left\{[\boldsymbol{k}(i)]_u[\boldsymbol{k}(i)]_v[\boldsymbol{k}(i)]_a[\boldsymbol{k}(i)]_b\right\}\left[\boldsymbol{D}_{(i)}\right]_{ab}\,. \tag{4.41}$$

For alleviating the notation, we introduce matrix $\boldsymbol{K}^{(u,v)}$, whose $(a,b)^{\mathrm{th}}$ entry is:

$$\left[\boldsymbol{K}^{(u,v)}\right]_{ab} = \mathbb{E}\left\{[\boldsymbol{k}(i)]_u[\boldsymbol{k}(i)]_v[\boldsymbol{k}(i)]_a[\boldsymbol{k}(i)]_b\right\}\,. \tag{4.42}$$

Now we can write relation (4.41) as:

$$[\boldsymbol{Q}(i)]_{uv} = \mathrm{Tr}\left\{\boldsymbol{K}^{(u,v)}\boldsymbol{D}_{(i)}\right\}\,. \tag{4.43}$$

It is important to note that indexes u, v, a, b in relations (4.41) and (4.42) act upon the whole block-vector $\boldsymbol{k}(i)$. As such, it is necessary to identify which particular block $m = 1, \ldots, N$ and specific dictionary entry j – helping in determining ω_j – any of these indexes point to.

Table 4.3: Identification of blocks and dictionary entries depending on the generic index h

Case	Block m	Dictionary entry j
$h \leq \operatorname{card}\{\mathcal{D}_1\}$	$m = 1$	$j = h$
$\sum_{\ell=1}^{\bar{\ell}-1} \operatorname{card}\{\mathcal{D}_\ell\} < h \leq \sum_{\ell=1}^{\bar{\ell}} \operatorname{card}\{\mathcal{D}_\ell\}$ with $\bar{\ell} = \{2, \ldots, N-1\}$	$m = \bar{\ell}$	$j = h - \sum_{\ell=1}^{\bar{\ell}-1} \operatorname{card}\{\mathcal{D}_\ell\}$
$\sum_{\ell=1}^{N-1} \operatorname{card}\{\mathcal{D}_\ell\} < h$	$m = N$	$j = h - \sum_{\ell=1}^{N-1} \operatorname{card}\{\mathcal{D}_\ell\}$

Knowing the number of entries of the dictionaries, this identification is straightforward. Let $h = \{u, v, a, b\}$ be a generic index, able to replace any and all of the other indexes. The corresponding identification process is summarized in Table 4.3.

After having identified indexes m_1, m_2, m_3, m_4 and $\omega_p, \omega_q, \omega_r, \omega_s$, we can write the following:

$$\begin{aligned}
\left[\boldsymbol{K}^{(u,v)}\right]_{ab} =& \mathbb{E}\left\{[\boldsymbol{k}(i)]_u [\boldsymbol{k}(i)]_v [\boldsymbol{k}(i)]_a [\boldsymbol{k}(i)]_b\right\} \\
=& \mathbb{E}\Bigg\{ \exp\Bigg(-\frac{1}{2\sigma^2}\Bigg(\left\|\boldsymbol{y}_{L_{m_1}}(i) - \boldsymbol{y}_{L_{m_1}}(\omega_p)\right\|^2 + \left\|\boldsymbol{y}_{L_{m_2}}(i) - \boldsymbol{y}_{L_{m_2}}(\omega_q)\right\|^2 \\
& + \left\|\boldsymbol{y}_{L_{m_3}}(i) - \boldsymbol{y}_{L_{m_3}}(\omega_r)\right\|^2 + \left\|\boldsymbol{y}_{L_{m_4}}(i) - \boldsymbol{y}_{L_{m_4}}(\omega_s)\right\|^2 \Bigg)\Bigg)\Bigg\} \qquad (4.44) \\
=& \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{y}^{\mathrm{d}}\right\|^2\right) \mathbb{E}\left\{\exp\left(-\frac{1}{\sigma^2}\left(\frac{1}{4}\left\|\boldsymbol{y}^{\mathrm{i}}\right\|^2 - \left(\boldsymbol{y}^{\mathrm{d}}\right)^\top \boldsymbol{y}^{\mathrm{i}}\right)\right)\right\}, \qquad (4.45)
\end{aligned}$$

with $\boldsymbol{y}^{\mathrm{d}} = \left[\boldsymbol{y}_{L_{m_1}}^\top(\omega_p) \quad \boldsymbol{y}_{L_{m_2}}^\top(\omega_q) \quad \boldsymbol{y}_{L_{m_3}}^\top(\omega_r) \quad \boldsymbol{y}_{L_{m_4}}^\top(\omega_s)\right]^\top$ collecting the dictionary entries and $\boldsymbol{y}^{\mathrm{i}} = \left[\boldsymbol{y}_{L_{m_1}}^\top(i) \quad \boldsymbol{y}_{L_{m_2}}^\top(i) \quad \boldsymbol{y}_{L_{m_3}}^\top(i) \quad \boldsymbol{y}_{L_{m_4}}^\top(i)\right]^\top$ collecting the instantaneous measurements. We now use relation (4.28), with $\boldsymbol{B} = \frac{1}{4}\boldsymbol{I}$, $\boldsymbol{b} = -\boldsymbol{y}^{\mathrm{d}}$ and $t = -\frac{1}{\sigma^2}$. Thus, we obtain:

$$\begin{aligned}
\left[\boldsymbol{K}^{(u,v)}\right]_{ab} =& \exp\left(-\frac{1}{2\sigma^2}\left\|\boldsymbol{y}^{\mathrm{d}}\right\|^2\right) \det\left\{\boldsymbol{I} + \frac{1}{2\sigma^2}\boldsymbol{R}_{yy}^{(m_1\to 4)}\right\}^{-\frac{1}{2}} \\
& \times \exp\left(\frac{1}{2\sigma^4}\left(\boldsymbol{y}^{\mathrm{d}}\right)^\top \boldsymbol{H}^{(m_1\to 4)}\boldsymbol{y}^{\mathrm{d}}\right), \qquad (4.46)
\end{aligned}$$

where $\boldsymbol{I}$ is of size $\sum_{\ell=1}^4 L_{m_\ell} \times \sum_{\ell=1}^4 L_{m_\ell}$, $\boldsymbol{H}^{(m_1\to 4)} = \boldsymbol{R}_{yy}^{(m_1\to 4)}\left(\boldsymbol{I} + \frac{1}{2\sigma^2}\boldsymbol{R}_{yy}^{(m_1\to 4)}\right)^{-1}$ and $\boldsymbol{R}_{yy}^{(m_1\to 4)}$ a block matrix formed similarly to $\boldsymbol{R}_{yy}^{(m_1 m_2)}$. Each of its blocks $(k, \ell) \in \{1, 2, 3, 4\}^2$ is equal to $[\boldsymbol{R}_{yy}]_{m_k m_\ell} \mathbb{1}_{L_{m_k}} \mathbb{1}_{L_{m_\ell}}^\top$. Its full form is given by (A.1). Recursion (4.34) can now be computed.

Steady-state MSD: In order to further compute the steady-state MSD, we stack columns of $\boldsymbol{D}_{(i)}$ on top of each other, i.e., $\bar{\boldsymbol{d}}_{(i)} = \operatorname{vec}\left\{\boldsymbol{D}_{(i)}\right\}$. Making use of the properties of the vectorization operator and relation (4.34), we obtain:

$$\bar{\boldsymbol{d}}_{(i+1)} = \boldsymbol{F}_0 \bar{\boldsymbol{d}}_{(i)} + \mu^2 J_{n,\min} \bar{\boldsymbol{r}}_{kk}, \qquad (4.47)$$

where $\bar{\boldsymbol{r}}_{kk} = \text{vec}\{\boldsymbol{R}_{kk}\}$, and:

$$\boldsymbol{F}_0 = \boldsymbol{I}_2 - \mu(\boldsymbol{I} \otimes \boldsymbol{R}_{kk} + \boldsymbol{R}_{kk} \otimes \boldsymbol{I}) + \mu^2 \boldsymbol{F}_1 \,. \tag{4.48}$$

We remark upon the fact that the identity matrix $\boldsymbol{I}_2$ is of size $k_D^2 \times k_D^2$, while $\boldsymbol{I}$ is if size $k_D \times k_D$. Also, entries of the matrix $\boldsymbol{F}_1$ are $[\boldsymbol{F}_1]_{u+(v-1)k_D,a+(b-1)k_D} = \left[\boldsymbol{K}^{(u,v)}\right]_{ab}$.

Assuming a small enough step size μ, the algorithm is mean-square stable as $i \to \infty$, and converges towards:

$$\lim_{i\to\infty} \bar{\boldsymbol{d}}_{(i)} = \mu^2 J_{n,\min} \left(\boldsymbol{I} - \boldsymbol{F}_0\right)^{-1} \bar{\boldsymbol{r}}_{kk} \triangleq \bar{\boldsymbol{d}}_{(\infty)} \,. \tag{4.49}$$

Using relations (4.39) – (4.40) and the matrix form $\boldsymbol{D}_{(\infty)}$ of $\bar{\boldsymbol{d}}_{(\infty)}$, i.e., $\boldsymbol{D}_{(\infty)} = \text{vec}^{-1}\left\{\bar{\boldsymbol{d}}_{(\infty)}\right\}$, the steady-state MSE is given by:

$$J_{n,\text{MSE}}(\infty) = J_{n,\min} + \text{Tr}\left\{\boldsymbol{R}_{kk}\boldsymbol{D}_{(\infty)}\right\} \,, \tag{4.50}$$

while the steady state MSD is:

$$\text{MSD}(\infty) = \text{Tr}\left\{\boldsymbol{D}_{(\infty)}\right\} \,. \tag{4.51}$$

4.5 Theoretical validation and experimental results

4.5.1 Theoretical validation

Data generation: We recall our local data model:

$$y_n(i) = \sum_{m\in\mathcal{N}_{\setminus n}} a_{nm} f_m(\boldsymbol{y}_{L_m}(i)) + v_n(i) \,. \tag{4.52}$$

For purposes of simplification, we consider the time dependence lags L_m equal to 1, i.e., $\boldsymbol{y}_{L_m}(i) = y_m(i)$, and that $f_m(y_m(i)) = y_m(i)$. We assumed that $\boldsymbol{y}(i) = \text{col}\left\{\{y_n(i)\}_{n=1}^N\right\}$ is an i.i.d., zero-mean Gaussian signal with covariance $\boldsymbol{R}_y \triangleq \mathbb{E}\left\{\boldsymbol{y}(i)\boldsymbol{y}^\top(i)\right\}$. With these remarks, computing $\boldsymbol{R}_y$ from the local model above reduces to:

$$[\boldsymbol{I} - \boldsymbol{A}]\boldsymbol{y}(i) = \boldsymbol{v}(i) \tag{4.53}$$

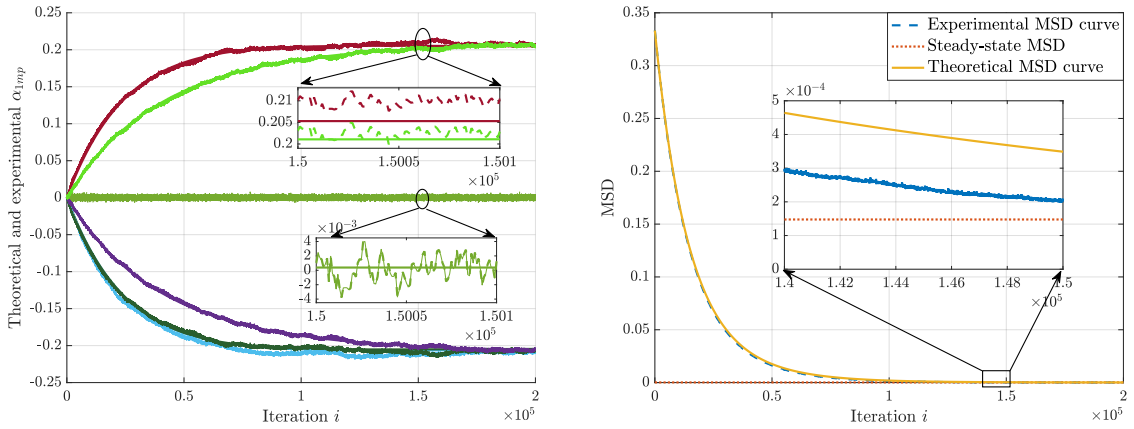
$$\Rightarrow \mathbb{E}\left\{[\boldsymbol{I} - \boldsymbol{A}]\boldsymbol{y}(i)\boldsymbol{y}^\top(i)[\boldsymbol{I} - \boldsymbol{A}]^\top\right\} = \mathbb{E}\left\{\boldsymbol{v}(i)\boldsymbol{v}^\top(i)\right\} \tag{4.54}$$

$$\Rightarrow [\boldsymbol{I} - \boldsymbol{A}]\boldsymbol{R}_y[\boldsymbol{I} - \boldsymbol{A}]^\top = \boldsymbol{R}_v \tag{4.55}$$

$$\Rightarrow \boldsymbol{R}_y = [\boldsymbol{I} - \boldsymbol{A}]^{-1}\boldsymbol{R}_v([\boldsymbol{I} - \boldsymbol{A}]^\top)^{-1} \,, \tag{4.56}$$

where we select $\boldsymbol{R}_v \triangleq \mathbb{E}\left\{\boldsymbol{v}(i)\boldsymbol{v}^\top(i)\right\} = \text{diag}\left\{\left\{\sigma_{v,m}^2\right\}_{m=1}^5\right\}$ as the noise covariance, and the adjacency matrix:

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix} \,. \tag{4.57}$$



(a) Theoretical and experimental entries of $\boldsymbol{\alpha}_1$. Continuous lines are the theoretical curves, while dashed line are experimental

(b) Experimental, steady-state, and theoretical MSD

Figure 4.1: Validation for the analysis in both the mean and mean square sense

We recall that we consider graphs without self-loops. Noise standard deviations were $\sigma_{v,m} = 0.05, \forall m \in \mathcal{N}$. We note that the theoretical validation simulations were ran for the first node, i.e., $n = 1$, with a step size $\mu = 5 \cdot 10^{-2}$. Our algorithm used the Gaussian kernel (4.59), with band-width $\sigma = 1$. The 5-node graph has adjacency matrix (4.57). Each node stored a dictionary $\mathcal{D}_m$ with 3 entries, chosen in a uniform grid on $[-1, 1]$. Simulations were averaged over 100 Monte-Carlo runs.

Quantifying the performance: Fig. 4.1a shows both the theoretical and experimental values for coefficients α_{1mp}. We remark upon the fact that, for visibility and clarity reasons, only a selection of non-zeros coefficients are depicted. Fig. 4.1b shows both the theoretical and experimental MSD curves, as well as the steady-state MSD. The experimental MSD was computed using:

$$\text{MSD}(i) = \mathbb{E}\left\{\|\hat{\boldsymbol{\alpha}}_n(i) - \boldsymbol{\alpha}_n^*\|^2\right\} . \tag{4.58}$$

The curves show how our theoretical curves are generally consistent with the experimental ones, for both the mean error and the mean square error case. The results of this analysis can prove useful in selecting an adequate step-size when processing data with known statistical properties. Moreover, they show that behavior of the algorithm as a function of i is predictable.

4.5.2 Experimental results

Experimental setup: Multiple experiments have been conducted, with the goal of showcasing different characteristics of the developed algorithm. Firstly, a simple 3-node graph is considered, where nonlinearities are present in node interactions. Secondly, on real biomedical data, the obtained results are coherent with the results obtained in other works. Thirdly, we take into

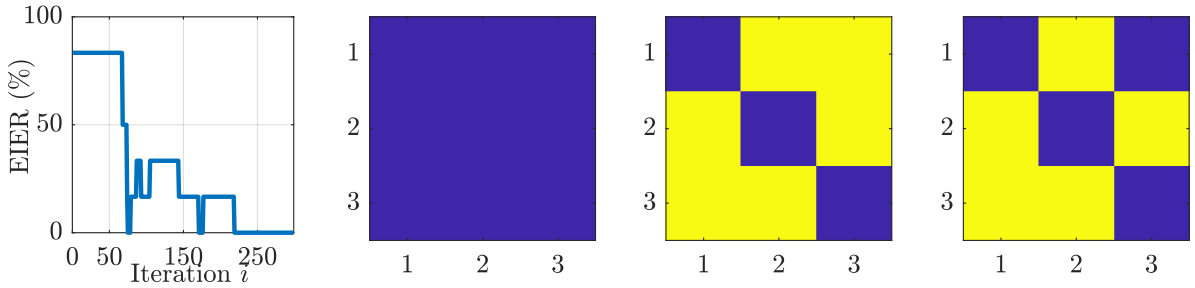


Figure 4.2: Performance in terms of EIER, as well as estimates of $\boldsymbol{A}$ at $i = 50, 150, 250$

consideration a dynamic graph setting in order to test the adaptive capabilities of the algorithm. Along our experiments, we either used the Gaussian kernel:

$$\kappa^{\mathrm{G}}(\boldsymbol{a}, \boldsymbol{b}) = \exp\left(-\frac{\|\boldsymbol{a} - \boldsymbol{b}\|^2}{2\sigma_n^2}\right), \quad \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^N, \tag{4.59}$$

or the exponential kernel:

$$\kappa^{\mathrm{E}}(\boldsymbol{a}, \boldsymbol{b}) = \exp\left(-\frac{\|\boldsymbol{a} - \boldsymbol{b}\|}{\sigma_n}\right), \quad \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^N, \tag{4.60}$$

where σ_n is a hyper-parameter of the kernel function.

Discretized Lorenz Attractor: Consider the discretized version of the Lorenz attractor [Lorenz, 1989, Tobar et al., 2014]:

$$\begin{bmatrix} y_1(i+1) \\ y_2(i+1) \\ y_3(i+1) \end{bmatrix} = \begin{bmatrix} y_1(i) \\ y_2(i) \\ y_3(i) \end{bmatrix} + 0.01 \begin{bmatrix} 10\,(y_2(i) - y_1(i)) \\ y_1(i)\,(28 - y_3(i)) - y_2(i) \\ y_1(i)y_2(i) - \dfrac{8}{3}y_3(i) \end{bmatrix}, \quad i \geq 0, \tag{4.61}$$

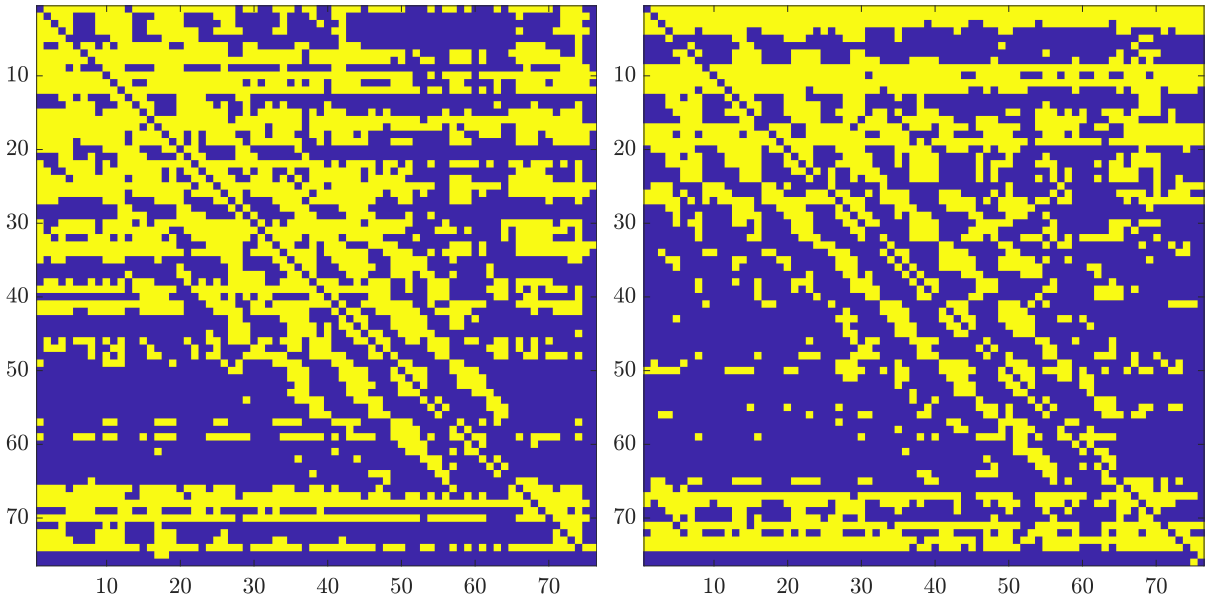
with initial conditions $[y_1(0), y_2(0), y_3(0)]^\top = [10^{-2}, 10^{-2}, 10^{-2}]^\top$. We used the Gaussian kernel and set $\mu_n = 0.1, \sigma_n = 8, \xi_n = 0.8$. Parameters η_n and τ_n were set as to achieve the best performance in terms of EIER:

$$\mathrm{EIER} \triangleq \frac{\|\boldsymbol{A}_{\mathrm{gt}} - \hat{\boldsymbol{A}}\|_0}{N(N-1)} \cdot 100\%, \tag{4.62}$$

where $\boldsymbol{A}_{\mathrm{gt}}$ is the ground truth. Its binary entries encode direct influence, based on (4.61), between node couples, excluding self-loops, i.e.:

$$\boldsymbol{A}_{\mathrm{gt}} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}. \tag{4.63}$$

Fig. 4.2 shows the EIER, and the estimates of $\boldsymbol{A}$ at iterations 50, 150, 250. These results, obtained after less than 300 samples with the dictionaries containing only between 6 and 8 kernel functions per node, show that the proposed method is able to infer links in a distributed and online manner, even if they are based on nonlinear interactions.

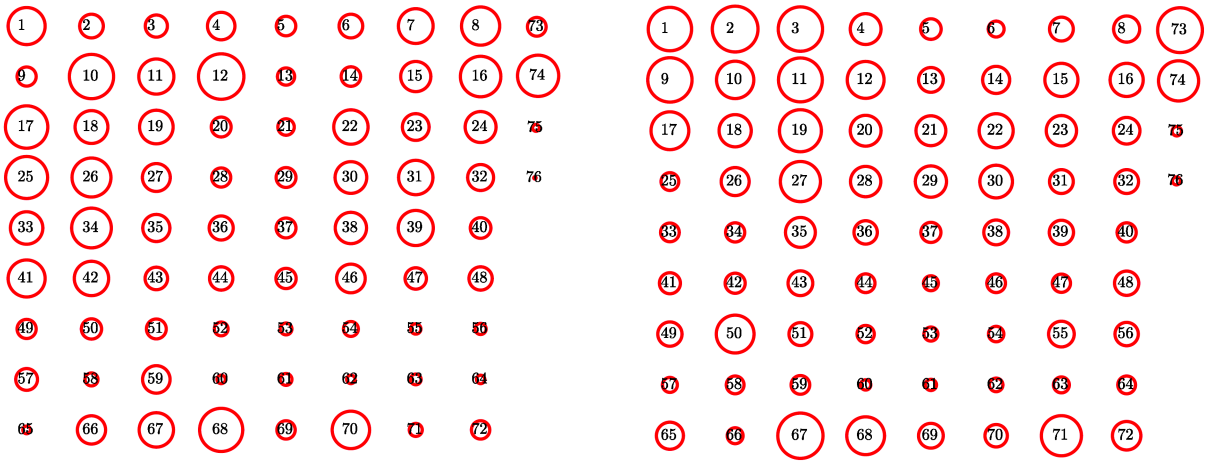


(a) *Preictal* interval (b) *Ictal* interval

Figure 4.3: Estimated adjacency matrices for each interval

Tests on epilepsy seizure data: In this experiment, we aimed to show how the estimated topology using the presented method is consistent with results presented in other works. The used data come from a 39-year-old female subject suffering from intractable epilepsy. The data acquisition and pre-processing information is provided in [Kramer et al., 2008]. The data set contains 8 instances of electrocorticography (ECoG) time series, each instance representing one seizure and contains voltage measurements from 76 different regions on and inside the brain, during the 10 seconds before the epilepsy seizure (*preictal* interval) and the first 10 seconds during the seizure (*ictal* interval). Studies in epilepsy were able to shed light on the relations between the different regions of the brain, as well as their functions [Dennerll, 1964]. This condition manifests through a plethora of seemingly random electrical signals discharging in a certain region, then propagating throughout the brain. The Gaussian kernel was used, and we set $\mu_n = 5 \cdot 10^{-5}, \sigma_n = 90, \xi_n = 0.9$ for each n. Concerning the choice of the η_n and τ_n, since there was no ground truth, they were set as to obtain coherent results with previous works, while ensuring a weakly-connected graph, i.e., the existence of a path between any node couple, independent of edge directionality.

Algorithm 2 was ran on these data. In Fig. 4.3 we show the estimated connectivity of the brain, during both these intervals, averaged over the 8 instances. For further insight, Table 4.4 depicts some graph metrics, which are further detailed in Annex B. Other detailed metrics are presented in Fig. 4.5. Interestingly, the betweenness measure is lower for nodes 5 – 8 and 56 – 65, an indication that these particular nodes do not influence the transit of information as much as the others. Moreover, the same nodes also have a relatively low *hub* centrality measure, which we can interpret as a another sign of their decreased influence on other nodes. We make a third



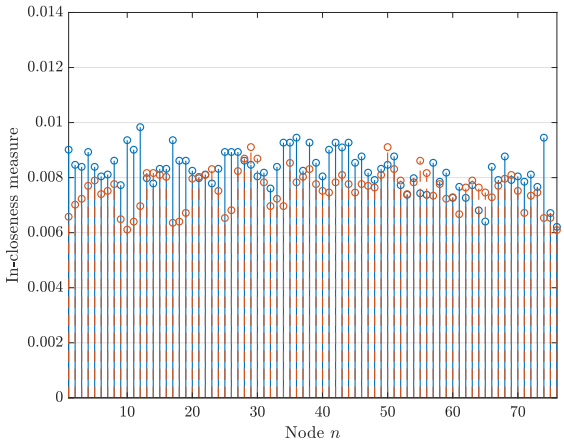
(a) *Preictal* interval (b) *Ictal* interval

Figure 4.4: Summed in- and out-degrees for the estimated graphs for both *preictal* and *ictal* intervals. The radii encode the values of their respective node degree, relative for each interval. The larger the radius corresponding to node n, the larger the summed degree of node n

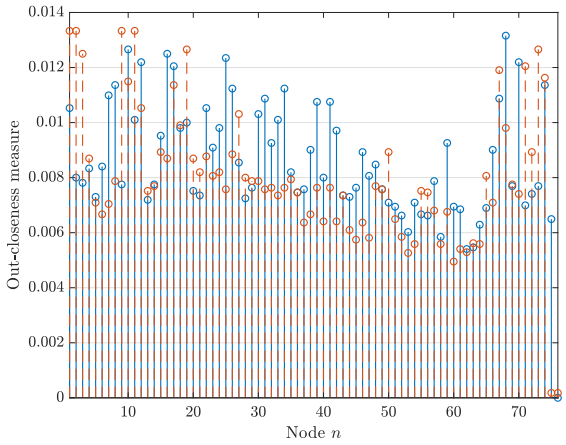
observation on the *authority* centrality measure of nodes 56 – 65, which is relatively low as well, which shows that these particular nodes are not influenced by their neighbors as much as the rest. Nodes 27, 50 and 66 exhibit a high betweenness measure in the *ictal* interval, meaning that their removal can seriously inhibit the information flow in the network.

Fig. 4.4 depicts the degree (sum of in- and out-degree), encoded in the radii of the circles, relative to each interval. Interestingly, our online estimate reveals roughly the same behavior before and during the seizure as the estimate obtained using the method developed in [Shen et al., 2019]. More precisely, the number of total connections decreases from one interval to the other, especially due to the variation of in-degrees for nodes 30 to 50. Further analyzing the connections, nodes 75 and 76 have a small in-degree, however they present a more important out-degree. Observe the decrease of the degree of node 26 or the major increase for node 73. This behavior is consistent with the findings of the aforementioned paper. Moreover, works in the field of epilepsy uncovered that metrics such as the average path length tend to be larger in the *ictal* interval [Ponten et al., 2007, Schindler et al., 2008, van Diessen et al., 2013], behavior obtained by our method as well.

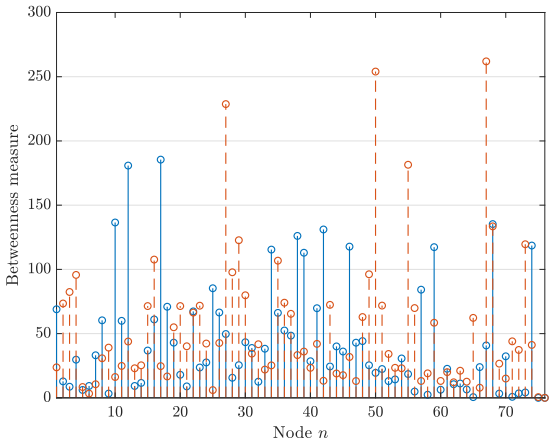
The algorithm is therefore able to obtain results similar to those obtained in previous works, while based on kernel dictionaries. For reference, the number of kernel functions inserted in the dictionaries varied between 15 to 30, after 4000 samples. These results show how only a reduced number of kernel functions are actually needed in order to obtain a topology estimate. This fact, alongside the online approach, can translate in reduced computational complexity, depending on the solver, due to the drastically reduced number of needed kernel functions.



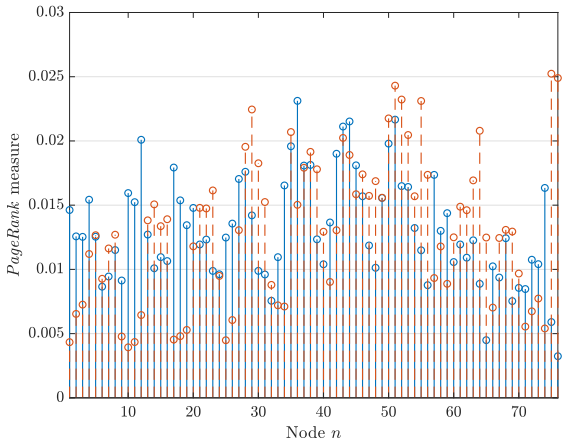
(a) The in-closeness centrality measure per node



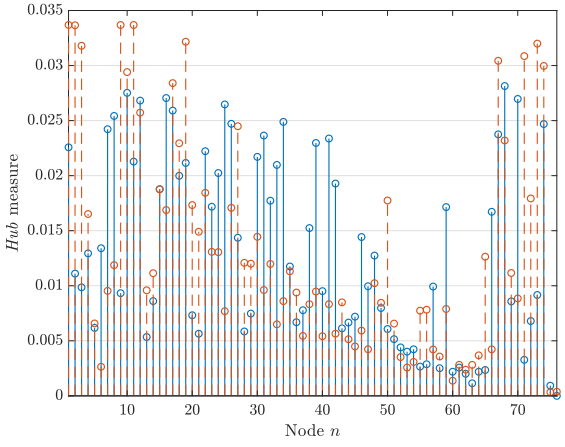
(b) The out-closeness centrality measure per node



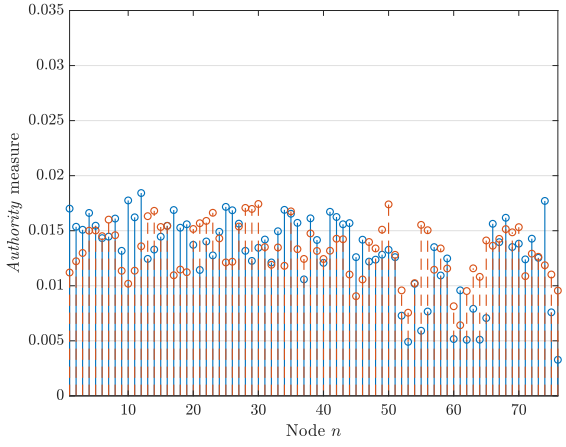
(c) The betweenness centrality measure per node



(d) The *PageRank* centrality measure per node



(e) The *hub* centrality measure per node



(f) The *authority* centrality measure per node

Figure 4.5: Various centrality measures per node, which are indicators of node importance within the graph. See Annex B for details on these measures. Blue continuous lines pertain to the *preictal*, while red dashed lines pertain to the *ictal* interval

Table 4.4: Metrics for the estimated topologies concerning the *preictal* and *ictal* intervals

Metric	*Preictal* interval	*Ictal* interval
Network density	0.436	0.377
Average in- (and out-) degree	32.67	28.29
Average path length	1.600	1.730
Average in-closeness	$8.287 \cdot 10^{-3}$	$7.570 \cdot 10^{-3}$
Average out-closeness	$8.617 \cdot 10^{-3}$	$8.007 \cdot 10^{-3}$
Average betweenness	44.45	53.34

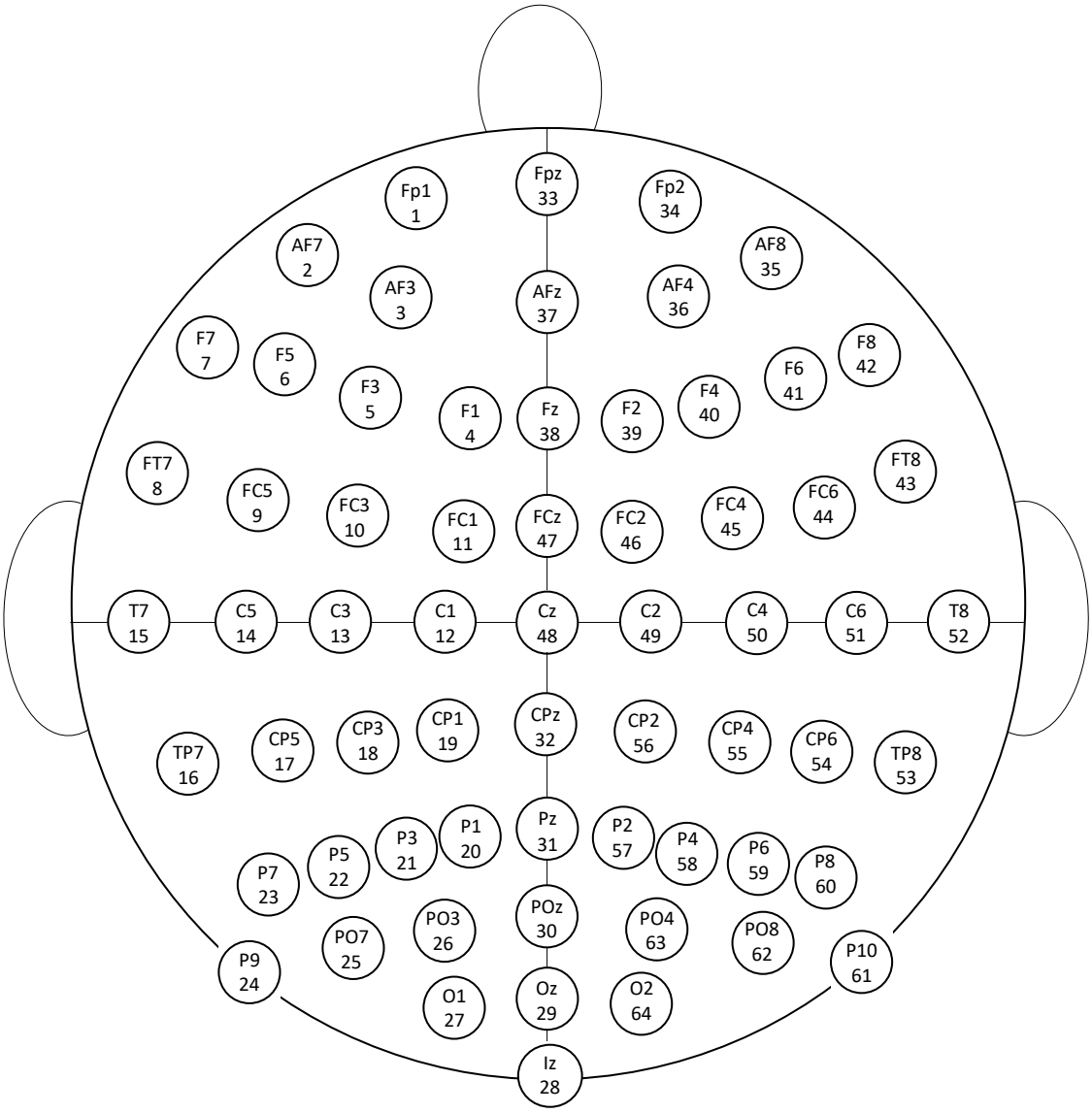


Figure 4.6: Electrode layout. Each circle represents one electrode. For each one, the site name is on the top, while the bottom is its corresponding node index

Real dynamic setting: The goal of this experiment is to analyze how the proposed method adapts to slow dynamic changes in topology. Once again, we use real data whose details are

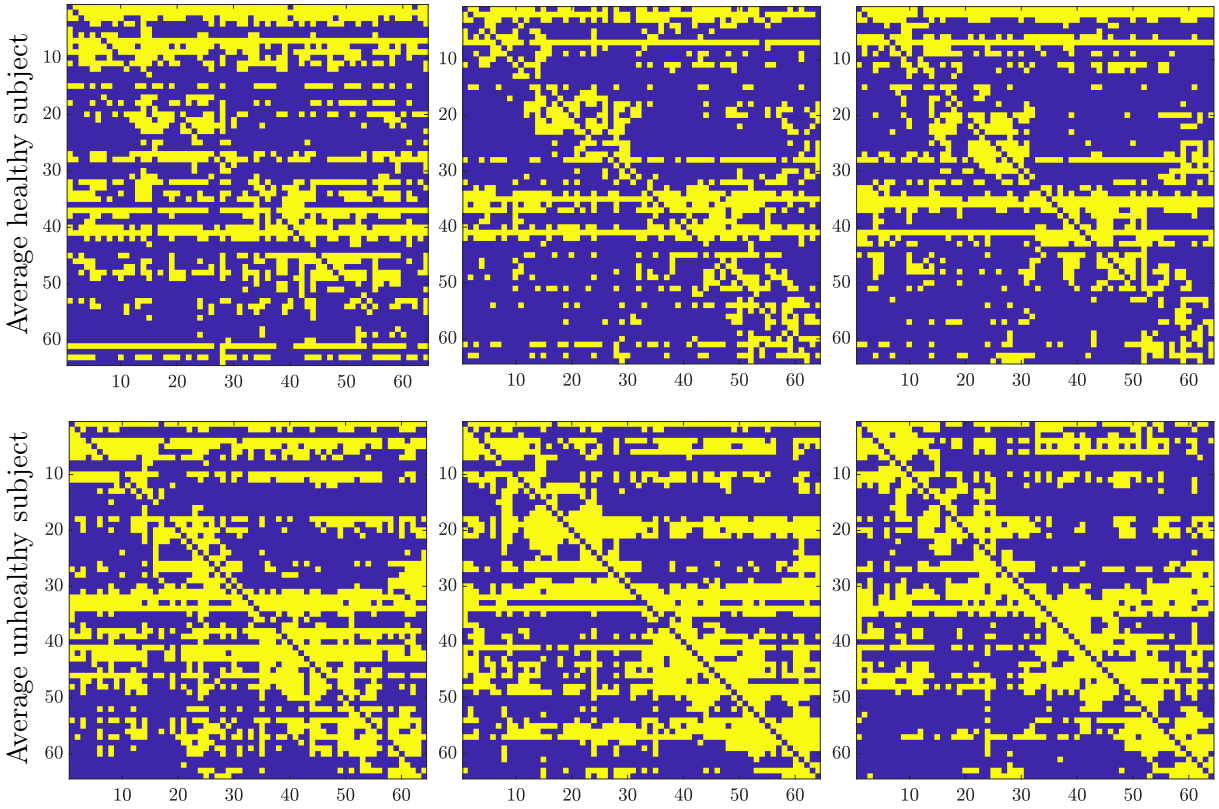
found in [Ford et al., 2013]. They represent electroencephalography (EEG) measurements taken from a group of 81 subjects in total, some of which are healthy and some of which suffer from schizophrenia. The electrode layout is presented in Fig. 4.6. A simple button-pressing task is set up, in three separate settings where subjects either:

1. pressed the button and a tone was immediately played;
2. listened to the tone without the button press;
3. pressed the button and the tone was not played.

The goal of the experiment was to check how the subjects' brains respond to sensory consequences of their own actions, in healthy and unhealthy subjects. This behavior arises when, for example, one voluntarily moves their eyes from side to side and their brain knows that the environment is not actually shifting. Patients suffering from schizophrenia have difficulties in differentiating between internally and externally generated stimuli [Freedman et al., 1996], meaning that they could encounter difficulties when discerning between tasks 1 (where the stimulus – the tone, is a direct consequence of their own action) and 2 (where the stimulus is initiated externally, without direct action). During our experiment, we used the measurements pertaining to three of the healthy subjects, namely subjects 1, 2, 3, and three of the schizophrenia-suffering, namely 67, 68, 69. A total of $M = 5000$ measurements per task and per subject were selected and fed to Algorithm 2, in order: task 1, task 2 and task 3, as if executed one after the other. For both the healthy and unhealthy sets we used the exponential kernel with $\sigma_n = 1$, $\mu_n = 9 \cdot 10^{-2}$, $\xi_n = 0.1$. In order to obtain comparable results, the same manner of choosing parameters η_n and τ_n was used in both cases, and set as to obtain weakly-connected graphs. For each subject in the healthy group, 30 to 80 kernel functions were chosen for the dictionary, while for each subject in the unhealthy group between 20 and 60. These values are significantly lower than the possible maximum of 5000 dictionary elements for each patient.

The estimated topologies, averaged over the three subjects in the healthy and unhealthy groups respectively, are depicted in Fig. 4.7 for each of the three tasks. The topology was obtained by applying the threshold τ_n on the average of the estimated norms $\|\hat{a}_{nm}(M)\|$ for each group of three patients. For both the healthy subjects (first row) and the unhealthy schizophrenia-suffering (bottom row), no important changes appear in topology while moving from task to task. However, comparing the average healthy and unhealthy subjects, a rather different network structure arises for each case. Some graph metrics are given in Table 4.6. Interestingly, the graph-based analysis of schizophrenic patients conducted by the authors of [Olejarczyk and Jernajczyk, 2017] reports similar findings to ours, notably the reduced average path lengths exhibited by topologies pertaining to such patients.

See Fig. 4.8 for the evaluation of multiple centrality measures for each of the nodes, and Annex B for details on these metrics. An interesting remark is that nodes 28 and 61 have a relatively



(a) Task 1 (b) Task 2 (c) Task 3

Figure 4.7: Estimated topologies per task, averaged per group

high betweenness measure across all three tasks, for the case of the healthy group, indicative of their high importance in the flow of information and that their removal has the potential of fragmenting the global network. On the same line, nodes 2, 7, 41 and 61 exhibit a high *hub* centrality measure for the same group, for tasks 2 and 3. This measure, for the unhealthy group, is, however, relatively low on the same two tasks, indicating that there are no nodes of particularly high importance in the transit of information. However, for task 1, the unhealthy group shows a higher average betweenness and a lower average path length, signs that point towards a lesser connected network. This behavior is exposed in works such as [Venkataraman et al., 2012, Skåtun et al., 2016]. Moreover, according to the same works, patients with schizophrenia tend to have a higher connectivity in the frontal (approximately nodes $4-7$ and $39-42$) and parietal (approximately nodes $20-23$ and $57-60$) regions. This sort of higher regional connectivity is indeed visible in Fig. 4.7 across all tasks, as well as in the selection of local connectivity measures presented in Table 4.5.

4.6 Conclusion

An online, kernel-based, and distributed graph topology inference method was devised, which advocates the use of kernel dictionaries as a sparsification solution, under the coherence criterion.

Table 4.5: Local frontal and parietal metrics for the estimated topologies concerning the average healthy and unhealthy subject, for each of the three tasks

Local metric	Task	**Healthy**	**Unhealthy**
Frontal density	1	0.714	0.840
	2	0.697	0.643
	3	0.679	0.750
Parietal density	1	0.196	0.446
	2	0.482	0.500
	3	0.250	0.446
Average frontal betweenness	1	49.72	66.85
	2	50.70	61.46
	3	54.05	60.63
Average parietal betweenness	1	9.64	47.29
	2	42.33	32.75
	3	24.56	42.43

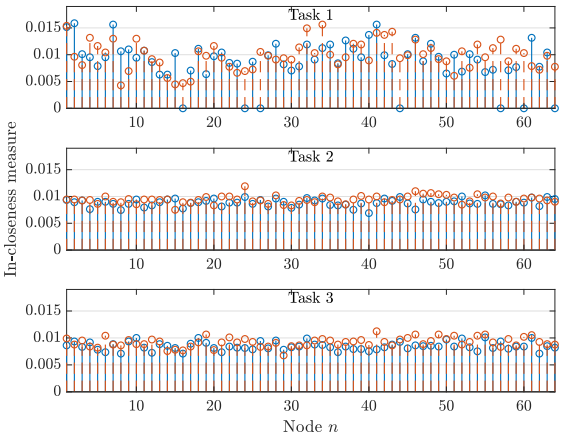
Table 4.6: Metrics for the estimated topologies concerning the average healthy and unhealthy subject, for each of the three tasks

Metric	Task	**Healthy**	**Unhealthy**
Network density	1	0.356	0.410
	2	0.286	0.451
	3	0.310	0.395
Average in- (and out-) degree	1	22.41	25.84
	2	17.98	28.44
	3	19.50	24.86
Average path length	1	1.705	1.753
	2	1.804	1.689
	3	1.888	1.739
Average in-closeness	1	$8.778 \cdot 10^{-3}$	$9.851 \cdot 10^{-3}$
	2	$8.857 \cdot 10^{-3}$	$9.449 \cdot 10^{-3}$
	3	$8.478 \cdot 10^{-3}$	$9.213 \cdot 10^{-3}$
Average out-closeness	1	$8.376 \cdot 10^{-3}$	$9.117 \cdot 10^{-3}$
	2	$9.231 \cdot 10^{-3}$	$9.758 \cdot 10^{-3}$
	3	$9.019 \cdot 10^{-3}$	$9.429 \cdot 10^{-3}$
Average betweenness	1	39.58	47.45
	2	50.62	43.42
	3	55.92	46.53

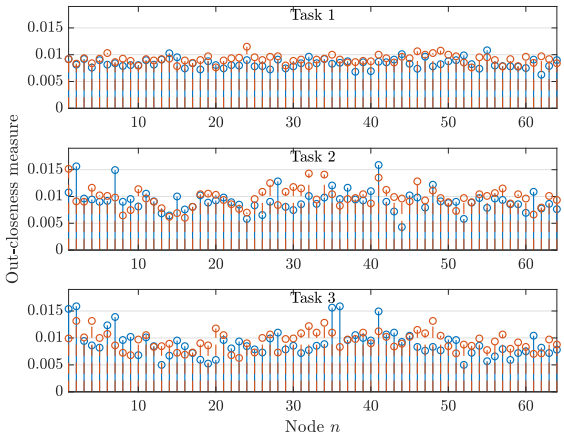
The use of kernels allows for the inference of connections when nonlinear links are presumed. While most state of the art methods rely on batch methods, which usually come with a high computational cost, the developed online algorithm comes with advantages such as adaptability. On the considered biomedical data, the method proved effective, paving the way to further work and research. Most importantly, behavior signaled by works in the study of epilepsy and schizophrenia was captured by the estimated networks and indicated through a set of both local

and global graph metrics.

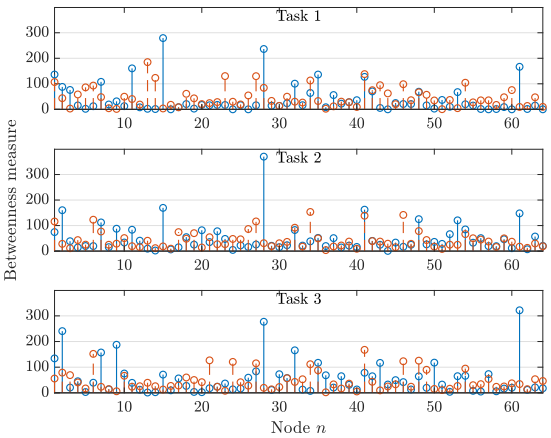
Chapter 5 proposes an alternative nonlinear model alongside a partial-derivative-based sparsity inducing method, under the same principles of online distributed and adaptive processing. In contrast with models such as the one employed in this current chapter, which do not consider the nonlinear interactions between multiple nodes and assume an additive nodal interaction, the next chapter introduces a general model accounting for any type of nodal interaction.



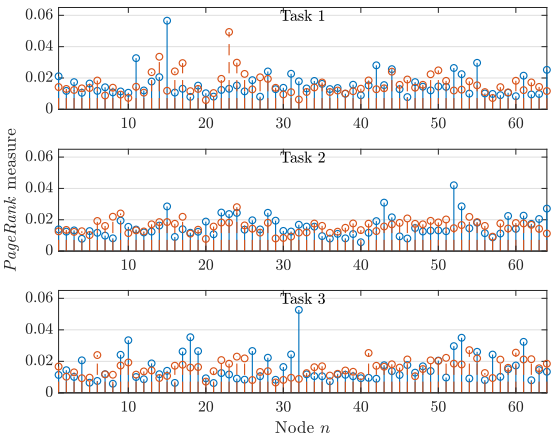
(a) The in-closeness centrality measure per node



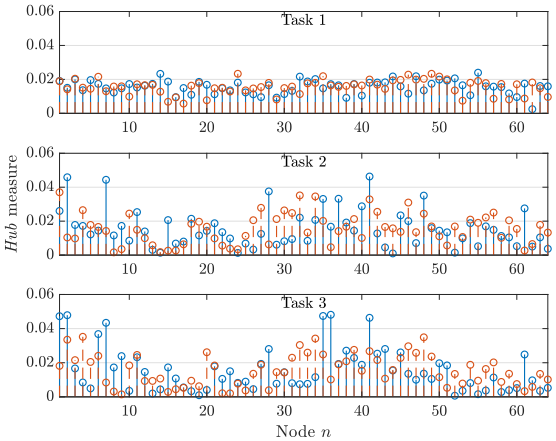
(b) The out-closeness centrality measure per node



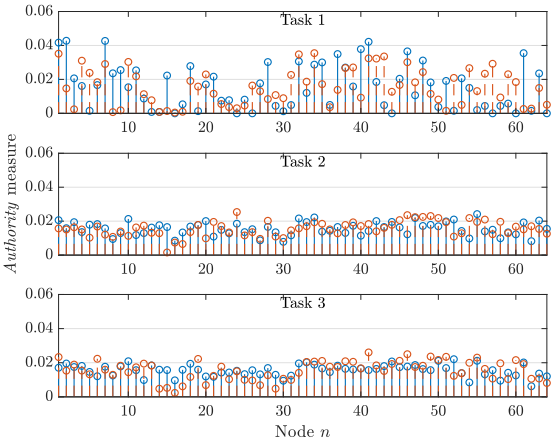
(c) The betweenness centrality measure per node



(d) The *PageRank* centrality measure per node



(e) The *hub* centrality measure per node



(f) The *authority* centrality measure per node

Figure 4.8: Various centrality measures per node, which are indicators of node importance within the graph. See Annex B for details on these measures. Blue continuous lines pertain to the group of healthy subjects, while red dashed lines pertain to the group of unhealthy subjects

CHAPTER 5

Topology inference with nonlinear dependencies: General model

Contents

In many real-world applications, such as brain network connectivity, gene networks or in shopping recommendations, the underlying graph explaining the different interactions between participating agents is not known. Moreover, many of these interactions may be based on nonlinear relationships, rendering the topology inference problem more complex. This chapter aims to develop a method of topology inference, under the form of a possibly non-symmetric adjacency matrix, able to explain nonlinear interactions between agents, in an online framework.

The proposed model is as general as possible, without assuming any nodal interchange based on, e.g., the additive model (4.3). Sparsity on the estimated matrices is imposed via partial derivatives, while kernel functions are used to model these nonlinear interactions. The impact of the increasing number of data points is alleviated by using dictionaries of kernel functions. A comparison with a previously developed method showcases the generality of the method. Furthermore, the interpretation of the estimated networks on real biomedical data is coherent with reports from the medical community.

The work presented in this chapter was published in:

- M. Moscu, R. Borsoi, and C. Richard. Online kernel-based graph topology identification with partial-derivative-imposed sparsity. In *28th European Signal Processing Conference (EUSIPCO)*, pages 2190–2194, 2021b. doi: 10.23919/Eusipco47968.2020.9287624

- M. Moscu, R. Borsoi, and C. Richard. Graph Topology Inference with Kernels and Partial-derivative-imposed Sparsity: Algorithm and Convergence Analysis. 2020b. submitted

5.1 Introduction

In the analysis and processing over networks such as gene regulation systems [The International HapMap Consortium et al., 2007], socio-economical interactions [Heiberger, 2018], or brain activity [Kramer et al., 2008], graphs have proven to be a useful tool, given their inherently distributed nature. Most graph signal processing algorithms, however, assume the graph topology as known beforehand. Recently, significant interest has been dedicated to the estimation of the graph topology from available data. Most of these works assume linear dependencies between the agents, e.g., brain regions, genes in a network, or sectors of a market economy. However, the presence of nonlinear interactions in real-world applications imposes the need of developing more general algorithms. As such, the ability of reproducing kernels to model nonlinear relationships between nodal signals makes them a powerful tool in the graph inference process.

In this chapter, we consider a setting where online nodal measurements are acquired and subsequently used in order to infer the topology of an underlying network. In the developed approach, the goal is to estimate a possibly directed adjacency matrix while accounting for general nonlinear dependencies between nodal signals, all in a distributed manner over the different agents of the network. Since many real-world examples, such as social graphs, show considerable edge sparsity, a sparsity-inducing framework based on partial derivatives is employed.

We propose an online approach able to estimate an adjacency matrix based on a general nonlinear model, thus improving upon the additive model (4.3) previously considered in Chapter 4. This model ensures a better representativity of nonlinear interactions, without assuming a particular manner on how agents in a network influence each other. Moreover, due to the online nature of the method, the data acquisition process can be stopped exactly when an estimate is

obtained, thus allowing for the acquisition of solely the needed data, a property which can prove useful in domains such as medical research.

5.2 General nonlinear problem and distributed problem statement

Table 5.1: List of notations and symbols present in Chapter 5

Symbol	Definition
$\boldsymbol{A}$	Adjacency matrix of a graph
$\mathcal{N}$	Set of nodes of the graph
$\mathcal{N}_n$	Set of nodes in the neighborhood of node n, excluding node n
$\mathcal{N}_{\setminus n}$	Set of all nodes, excluding node n
$J_n(\cdot)$	Local cost function
N	Total number of nodes in the graph
$\mathcal{H}_\kappa$	A Reproducing Kernel Hilbert Space associated to kernel κ
L_m	Time lag pertaining to the influence of node m
$\mathcal{D}_n$	The dictionary of node n
ξ_n	Dictionary admission threshold for node n
$\boldsymbol{\Omega}(\cdot)$	Sparsity-inducing regularizer

Consider an $(N+1)$-node graph with adjacency matrix $\boldsymbol{A}$ which models a system such as the brain network or a power grid. In this setting, the electrical activity of different brain-regions [Rubinov and Sporns, 2010, Shen et al., 2019], or the voltage angle per bus [Zhang et al., 2017], numbered from 1 through $(N+1)$, can be measured at different time instants $i \in \mathbb{N}_+$, leading to a dynamic graph signal $\boldsymbol{y}(i)$. The signal at each node influences and is influenced by the signals at the other nodes, with nonlinear relationships being reported in many applications such as, e.g., in the case of brain connectivity [Freeman, 1979, de Zwart et al., 2009]. The links between the signals at different nodes are then encoded in the matrix $\boldsymbol{A}$.

The distributed nature of graphs allows for a local problem formulation. As such, we focus on a single node n, while keeping in mind that the following reasoning can be applied for any other particular node. For ease of notation, we assume that $n \equiv (N+1)$ (i.e., we identify n with the $(N+1)^{\text{th}}$ node of the graph), which allows us to denote $\mathcal{N}_{\setminus n} = \{1, \ldots, N\}$. Recent methods have considered models of the form [Shen et al., 2017, Moscu et al., 2020a]:

$$y_n(i) = \sum_{m=1}^{N} a_{nm} g_m(\boldsymbol{y}_{L_m}(i)) + v_n(i)\,, \tag{5.1}$$

where $v_n(i)$ represents innovation noise, and a_{nm} is the $(n,m)^{\text{th}}$ entry of the graph adjacency matrix $\boldsymbol{A}$. This matrix models how each function g_m, $m = 1, \ldots, N$ influences the signal observed at node n. Let $g_m : \mathbb{R}^{L_m} \to \mathbb{R}$ be a nonlinear function whose possibly vector-valued argument is $\boldsymbol{y}_{L_m}(i) = [y_m(i), \ldots, y_m(i - L_m + 1)]^\top$, for $L_m \geq 1$. Therefore, the signal at each node depends

nonlinearly on the signals at all the other nodes, up to the past L_m samples. Given nodal measurements $\boldsymbol{y}(i)$ acquired online and model (5.1), the goal is to estimate the adjacency matrix $\boldsymbol{A}$ locally at each node:

$$\begin{aligned} &\underset{\boldsymbol{a}_n, g_1, \ldots, g_N}{\operatorname{argmin}} \frac{1}{2}\mathbb{E}\left\{\left|y_n(i) - \sum_{m=1}^{N} a_{nm} g_m(\boldsymbol{y}_{L_m}(i))\right|^2\right\} + \psi(\boldsymbol{a}_n) \\ &\text{subject to } a_{nm} \in \{0,1\}\,, \end{aligned} \tag{5.2}$$

where $\boldsymbol{a}_n$ is the n^{th} row of $\boldsymbol{A}$, and function $\psi(\cdot)$ is a sparsity promoting regularizer. However, models such as (5.1) do not consider the nonlinear interactions between multiple nodes, as they assume an additive model for $y_n(i)$ [Buja et al., 1989]. To overcome this issue, we propose to consider the following general nonlinear model:

$$y_n(i) = f_n(\boldsymbol{y}_{L_1}(i), \ldots, \boldsymbol{y}_{L_N}(i)) + v_n(i)\,. \tag{5.3}$$

Model (5.3), compared to (5.1), can capture and account for more complex relationships between the different nodes since it does not rely on a fixed presumed interaction model, therefore rendering it more general.

5.3 Introducing sparsity

5.3.1 Nonparametric sparsity

Although kernel-based and other nonlinear regression frameworks can be applied to estimate the function f_n described in equation (5.3), there remains a challenge to relate f_n to the underlying graph topology $\boldsymbol{A}$. Although the lack of an additive model precludes a straightforward relationship such as in (5.1), the influence of a certain variable can be quantified by the norm of the corresponding partial derivative, i.e.:

$$\text{node } m \text{ does not influence } n \iff \left\|\frac{\partial f_n}{\partial \boldsymbol{y}_{L_m}}\right\| = 0\,, \tag{5.4}$$

under the assumption that f_n is continuously differentiable.

If we do not assume the additive model, we can generalize problem (5.2) as:

$$\begin{aligned} &\underset{f_n}{\operatorname{argmin}} \frac{1}{2}\mathbb{E}\left|y_n(i) - f_n(\boldsymbol{y}_{L_1}(i), \ldots, \boldsymbol{y}_{L_N}(i))\right|^2 \\ &\text{subject to } \operatorname{col}\left\{\left\{\left\|\frac{\partial f_n}{\partial \boldsymbol{y}_{L_m}}\right\|\right\}_{m=1}^{N}\right\} \text{ being sparse}\,. \end{aligned} \tag{5.5}$$

Let us denote $\widetilde{\boldsymbol{y}}(i) = [\boldsymbol{y}_{L_1}(i)^\top, \ldots, \boldsymbol{y}_{L_N}(i)^\top]^\top$. As thoroughly detailed in [Rosasco et al., 2013], in order to define a nonparametric notion of sparsity that leads to a convex optimization problem, one can define the sparsity through the following functional:

$$\boldsymbol{\Omega}_{\mathbb{E}}(f_n) = \sum_{m=1}^{N} \left\|\frac{\partial f_n}{\partial \boldsymbol{y}_{L_m}}\right\|_{\mathbb{E}} = \sum_{m=1}^{N} \sqrt{\mathbb{E}_{\widetilde{\boldsymbol{y}}}\left\{\left\|\frac{\partial f_n(\widetilde{\boldsymbol{y}})}{\partial \boldsymbol{y}_{L_m}}\right\|^2\right\}}\,. \tag{5.6}$$

The expectation involving the derivatives in (5.6) can be approximated by the empirical average on all the data samples available up to instant i. Employing the ℓ_2-norm, as proposed in the aforementioned paper, and approximating the expectation, we obtain the following sampled version functional:

$$\boldsymbol{\Omega}(f_n) = \sum_{m=1}^{N} \left\| \frac{\partial f_n}{\partial \boldsymbol{y}_{L_m}} \right\|_i = \sum_{m=1}^{N} \sqrt{\frac{1}{i} \sum_{p=1}^{i} \left\| \frac{\partial f_n(\widetilde{\boldsymbol{y}}(p))}{\partial \boldsymbol{y}_{L_m}} \right\|^2}. \tag{5.7}$$

By including the regularizer $\boldsymbol{\Omega}(f_n)$ as an additive term in the cost function of (5.5), we are able to obtain a convex optimization problem, which allows us to obtain more efficient algorithms, and cater to real-word graphs, which tend to be sparse [Danisch et al., 2018].

5.3.2 Sparsity in Reproducing Kernel Hilbert Spaces

The penalty term proposed in the previous section allows us to promote sparsity in the estimated topology without the restrictive constraint of an additive model. However, there remains a fundamental step to constrain f_n to an adequate class of functions that is flexible but allows for an efficient, finite dimensional implementation. Several solutions exist in the literature, including nonlinear and polynomial Structural Equation Models [Jöreskog et al., 1996] and function selection from function sets defined *a priori* [Song et al., 2013]. In this work we will consider kernel methods, which address the presence of nonlinearities in classification or regression problems by applying linear algorithms to a high-dimensional feature space obtained by mapping the input data to an RKHS $\mathcal{H}_\kappa$ endowed with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_\kappa}$ and associated with a positive definite reproducing kernel $\kappa(\cdot, \cdot)$. RKHS-based solutions have been applied in the context of nonlinear additive models for online topology estimation in, e.g., [Moscu et al., 2020a]. The desired reproducing kernel property is briefly introduced in section 4.3.

Using the sparsity penalty in (5.7), constraining f_n to belong to an RKHS $\mathcal{H}_\kappa$ and approximating the expectation in (5.5) by an empirical average leads to the formulation of the following optimization problem:

$$\begin{aligned} \underset{f_n \in \mathcal{H}_\kappa}{\operatorname{argmin}} & \frac{1}{2i} \sum_{\ell=1}^{i} \left| y_n(\ell) - f_n\left(\boldsymbol{y}(\ell)\right) \right|^2 \\ & + \eta_n \left(\sum_{m=1}^{N} \sqrt{\frac{1}{i} \sum_{p=1}^{i} \sum_{q=1}^{L_m} \left(\frac{\partial f_n(\boldsymbol{y}(p))}{\partial y_{m,q}} \right)^2} + \psi_{\mathcal{H}_\kappa}\left(\|f_n\|_{\mathcal{H}_\kappa} \right) \right), \end{aligned} \tag{5.8}$$

where $y_{m,q}$ represents the q^{th} entry of $\boldsymbol{y}_{L_m}$. In (5.8), parameter $\eta_n > 0$ controls the relative importance of respecting the constraint on the unknown f_n, and $\psi_{\mathcal{H}_\kappa} : \mathbb{R} \to [0, \infty[$ is a monotonically increasing function.

Despite allowing us to introduce sparsity in f_n without constraining it to an additive model, the cost function in (5.8) also contains a significant challenge to an RKHS-based solution: the

sparsity-promoting penalty term hinders the direct application of traditional representation theorems to obtain a finite dimensional representation, because of the presence of the derivatives of f_n. However, if we suppose that the kernel $\kappa(\cdot,\cdot)$ is at least twice differentiable, the following relation holds [Zhou, 2008]:

$$\mathcal{H}_\kappa \ni \frac{\partial f_n(\boldsymbol{y})}{\partial y_{m,q}} = \langle f_n, \kappa_{\partial_{m,q}}(\cdot,\boldsymbol{y})\rangle_{\mathcal{H}_\kappa}\,, \tag{5.9}$$

where:

$$\kappa_{\partial_{m,q}}(\cdot,\boldsymbol{y}(q)) = \left.\frac{\partial\kappa(\cdot,\boldsymbol{a})}{\partial a_{m,q}}\right|_{\boldsymbol{a}=\boldsymbol{y}(q)}\,. \tag{5.10}$$

This means that for sufficiently smooth kernels, the derivative of functions in $\mathcal{H}_\kappa$ also belong to $\mathcal{H}_\kappa$, and can be evaluated in the form of simple inner products. This makes it possible to obtain a finite dimensional representation of the solution of (5.8), similarly to what has been previously done in [Rosasco et al., 2013]. As such, we can obtain a representer theorem for our approach, by generalizing the one presented in the aforementioned paper. We formalize this result in the following theorem:

Theorem 5.1. *Suppose that $\mathcal{H}_\kappa$ is an RKHS endowed with the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}_\kappa}$, and whose associated reproducing kernel $\kappa(\cdot,\cdot)$ is at least twice differentiable. Then, the optimal solution to the optimization problem* (5.8) *can be written as:*

$$f_n^* = \sum_{p=1}^{i}\alpha_p\kappa\left(\cdot,\boldsymbol{y}(p)\right) + \sum_{m=1}^{N}\sum_{\ell=1}^{i}\sum_{q=1}^{L_m}\beta_{m,\ell,q}\kappa_{\partial_{m,q}}\left(\cdot,\boldsymbol{y}(\ell)\right)\,. \tag{5.11}$$

Proof. Using the orthogonal projection, we can decompose any $f_n \in \mathcal{H}_\kappa$ as:

$$f_n = f_n^{\parallel} + f_n^{\perp}\,, \tag{5.12}$$

where $f_n^{\perp}$ is orthogonal to $f_n^{\parallel}$ (i.e., $\langle f_n^{\parallel}, f_n^{\perp}\rangle_{\mathcal{H}_\kappa} = 0$) and $f_n^{\parallel}$ can be written as:

$$f_n^{\parallel} = \sum_{\ell=1}^{i}\alpha_\ell\kappa\left(\cdot,\boldsymbol{y}(\ell)\right) + \sum_{m=1}^{N}\sum_{p=1}^{i}\sum_{q=1}^{L_m}\beta_{m,p,q}\kappa_{\partial_{m,q}}\left(\cdot,\boldsymbol{y}(p)\right)\,, \tag{5.13}$$

for an arbitrary choice of coefficients α_ℓ and $\beta_{m,p,q}$. This means that $f_n^{\parallel}$ lies in the span of $\kappa\left(\cdot,\boldsymbol{y}(\ell)\right)$ and $\kappa_{\partial_{m,q}}\left(\cdot,\boldsymbol{y}(p)\right)$, from which the orthogonality condition implies $\langle f_n^{\perp}, \kappa(\cdot,\boldsymbol{y}(\ell))\rangle_{\mathcal{H}_\kappa} = 0$ and $\langle f_n^{\perp}, \kappa_{\partial_{m,q}}\left(\cdot,\boldsymbol{y}(p)\right)\rangle_{\mathcal{H}_\kappa} = 0$, $\forall \ell, m, p, q$.

We suppose that the kernel $\kappa(\cdot,\cdot)$ is at least twice differentiable. Then, the following relation holds Zhou [2008]:

$$\mathcal{H}_\kappa \ni \frac{\partial f_n(\boldsymbol{y})}{\partial y_{m,q}} = \langle f_n, \kappa_{\partial_{m,q}}(\cdot,\boldsymbol{y})\rangle_{\mathcal{H}_\kappa}\,, \tag{5.14}$$

where $y_{m,q}$ is the q^{th} entry of $\boldsymbol{y}_{L_m}$.

We plug this decomposition of f_n in (5.12) in the cost function (5.8). For the first term in the cost function, using the kernel reproducing property, we have:

$$f_n(\boldsymbol{y}(\ell)) = \langle f_n, \kappa(\cdot, \boldsymbol{y}(\ell))\rangle_{\mathcal{H}_\kappa} = \langle f_n^{\parallel} + f_n^{\perp}, \kappa(\cdot, \boldsymbol{y}(\ell))\rangle_{\mathcal{H}_\kappa} = \langle f_n^{\parallel}, \kappa(\cdot, \boldsymbol{y}(\ell))\rangle_{\mathcal{H}_\kappa}\,, \tag{5.15}$$

$\forall \ell = 1, \ldots, i$, where $\langle f_n^{\perp}, \kappa(\cdot, \boldsymbol{y}(\ell))\rangle_{\mathcal{H}_\kappa} = 0$ since $f_n^{\perp}$ is orthogonal to each term in (5.13).

For the second term of the cost function:

$$\frac{\partial f_n(\boldsymbol{y}(p))}{\partial y_{m,q}} = \frac{\partial f_n(\boldsymbol{y})}{\partial y_{m,q}}\bigg|_{\boldsymbol{y}=\boldsymbol{y}(p)} = \langle f_n^{\parallel} + f_n^{\perp}, \kappa_{\partial_{m,q}}(\cdot, \boldsymbol{y}(p))\rangle_{\mathcal{H}_\kappa} = \langle f_n^{\parallel}, \kappa_{\partial_{m,q}}(\cdot, \boldsymbol{y}(p))\rangle_{\mathcal{H}_\kappa}\,, \tag{5.16}$$

$\forall p = 1, \ldots, i$, where $\langle f_n^{\perp}, \kappa_{\partial_{m,q}}(\cdot, \boldsymbol{y}(p))\rangle_{\mathcal{H}_\kappa} = 0$ since $f_n^{\perp}$ is perpendicular to each term in (5.13).

For the last term of the cost function we have:

$$\psi_{\mathcal{H}_\kappa}\big(\|f_n\|_{\mathcal{H}_\kappa}\big) = \psi_{\mathcal{H}_\kappa}\big(\|f_n^{\parallel} + f_n^{\perp}\|_{\mathcal{H}_\kappa}\big) = \psi_{\mathcal{H}_\kappa}\big(\|f_n^{\parallel}\|_{\mathcal{H}_\kappa} + \|f_n^{\perp}\|_{\mathcal{H}_\kappa}\big)\,. \tag{5.17}$$

Then, since $\|f_n^{\perp}\|_{\mathcal{H}_\kappa}$ does not influence the two first terms in the cost function, if $\psi_{\mathcal{H}_\kappa}$ is monotonically increasing, then the solution to (5.8) will be such that $\|f_n^{\perp}\|_{\mathcal{H}_\kappa} = 0$. □

Using the results of Theorem 5.1, relation (5.11) can be substituted in (5.8) in order to obtain the finite dimensional optimization problem:

$$\begin{aligned}
&\underset{\substack{\{\alpha_j\},\{\beta_{o,j,s}\}\\ j=1,\ldots,i\\ o=1,\ldots,N\\ s=1,\ldots,L_m}}{\operatorname{argmin}} \frac{1}{2i}\sum_{\ell=1}^{i}\left| y_n(\ell) - \left(\sum_{j=1}^{i}\alpha_j\kappa\left(\boldsymbol{y}(j),\boldsymbol{y}(\ell)\right) + \sum_{o=1}^{N}\sum_{j=1}^{i}\sum_{s=1}^{L_m}\beta_{o,j,s}\kappa_{\partial_{o,s}}\left(\boldsymbol{y}(j),\boldsymbol{y}(\ell)\right)\right)\right|^2 \\
&+\eta_n\sum_{m=1}^{N}\sqrt{\frac{1}{i}\sum_{p=1}^{i}\sum_{q=1}^{L_m}\left(\sum_{j=1}^{i}\alpha_j\kappa_{\partial_{m,q}}\left(\boldsymbol{y}(j),\boldsymbol{y}(p)\right) + \sum_{o=1}^{N}\sum_{j=1}^{i}\sum_{s=1}^{L_m}\beta_{o,j,s}\frac{\partial^2\kappa\left(\boldsymbol{y}(j),\boldsymbol{y}(p)\right)}{\partial y_{m,q}(p)\partial y_{o,s}(j)}\right)^2}\,,
\end{aligned} \tag{5.18}$$

where now we optimize over scalar coefficients instead of functions.

Before concluding this subsection, we note that an alternative solution to introducing sparsity exists in the literature: instead of introducing sparsity on the partial derivatives of the estimate function, works such as [Mukherjee and Wu, 2006, Ye and Xie, 2012] focus on learning the gradient directly, thus circumventing the need of learning the function itself. However, the proposed solutions are not straightforward to compute online and distributively, especially because of the highly demanding singular value decomposition.

For the remainder of the chapter, for ease of comprehension and notational simplicity, we consider the case of an instantaneous model, i.e., $L_m = 1$, $\forall m \in \mathcal{N}_{\backslash n}$, which means that we can now note $\boldsymbol{y}_{L_m}(i) = y_{m,q}(i) = y_m(i)$, and $\kappa_{\partial_{m,q}}(\cdot, \boldsymbol{y}) = \kappa_{\partial_m}(\cdot, \boldsymbol{y})$.

5.4 An online algorithm

An immediate observation concerning solution (5.11) is that the number of coefficients α_p and $\beta_{m,q}$ can become prohibitive as i increases, since each acquired measurement increases the number of kernel functions. A solution to this problem is the use of kernel dictionaries which can be defined *a priori* [Chen et al., 2014] or which can admit a new candidate kernel function only if the candidate function passes a certain sparsification rule based on, e.g., the coherence criterion [Richard et al., 2009]. Other options exist, some of which were briefly introduced in subsection 4.3.3. Under the coherence criterion framework, each node n in the network creates, updates, and stores a dictionary of kernel functions and their derivatives, $\mathcal{D}_n = \{\{\kappa(\cdot, \widetilde{\boldsymbol{y}}(\omega_j)), \kappa_{\partial_1}(\cdot, \widetilde{\boldsymbol{y}}(\omega_j)), \ldots, \kappa_{\partial_N}(\cdot, \widetilde{\boldsymbol{y}}(\omega_j))\} : \omega_j \in \mathcal{I}_n^i \subset \{1, \ldots, i-1\}\}$, where $\mathcal{I}_n^i$ represents the set of time indices of elements selected for the dictionary, before instant i. This entails the fact that, after a sufficient number of samples i has been acquired, only a number $\operatorname{card}\{\mathcal{D}_n\} \ll i$ of coefficient couples will be needed. A candidate kernel function $\kappa(\cdot, \widetilde{\boldsymbol{y}}(i))$ is added to $\mathcal{D}_n$ if the following sparsification condition holds [Richard et al., 2009]:

$$\max_{\omega_j \in \mathcal{I}_n^i} |\kappa(\widetilde{\boldsymbol{y}}(i), \widetilde{\boldsymbol{y}}(\omega_j))| \leq \xi_n \,, \tag{5.19}$$

where $\xi_n \in [0, 1[$ determines the level of sparsity and coherence of the dictionary. The number of entries in the dictionary satisfies $\operatorname{card}\{\mathcal{D}_n\} < \infty$ when $i \to \infty$ [Richard et al., 2009]. We rewrite relation (5.11) as:

$$f_n^* = \sum_{p=1}^{\operatorname{card}\{\mathcal{D}_n\}} \alpha_p \kappa(\cdot, \widetilde{\boldsymbol{y}}(\omega_p)) + \sum_{m=1}^{N} \sum_{q=1}^{\operatorname{card}\{\mathcal{D}_n\}} \beta_{m,q} \kappa_{\partial_m}(\cdot, \widetilde{\boldsymbol{y}}(\omega_q)) \,. \tag{5.20}$$

Let vectors $\boldsymbol{\alpha} = \operatorname{col}\left\{\{\alpha_p\}_{p=1}^{\operatorname{card}\{\mathcal{D}_n\}}\right\}$, $\boldsymbol{\beta} = \operatorname{col}\left\{\{\boldsymbol{\beta}_m\}_{m=1}^{N}\right\}$, with $\boldsymbol{\beta}_m = \operatorname{col}\left\{\{\beta_{m,q}\}_{q=1}^{\operatorname{card}\{\mathcal{D}_n\}}\right\}$, group the coefficients in (5.20). Considering the online version of the batch cost function (5.8) with the instantaneous MSD estimate (measured only at instant i), and using the dictionary-based representation of f_n^* in (5.20), we obtain the following finite-dimensional optimization problem:

$$\operatorname*{argmin}_{\boldsymbol{\gamma}} \frac{1}{2} \left| y_n(i) - \boldsymbol{\gamma}^\top \boldsymbol{s}(i) \right|^2 + \eta_n \sum_{m=1}^{N} \sqrt{\frac{1}{i} \sum_{p=1}^{i} \left(\boldsymbol{\gamma}^\top \boldsymbol{t}_m(p) \right)^2} \,, \tag{5.21}$$

with $\boldsymbol{s}(i) = \begin{bmatrix} \boldsymbol{z}(i) \\ \boldsymbol{k}(i) \end{bmatrix}$, $\boldsymbol{\gamma} = \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\alpha} \end{bmatrix}$, $\boldsymbol{t}_m(p) = \begin{bmatrix} \boldsymbol{\ell}_m(p) \\ \boldsymbol{z}_m(p) \end{bmatrix}$, and:

$$\boldsymbol{k}(i) = \operatorname{col}\Big\{ \kappa(\widetilde{\boldsymbol{y}}(i), \widetilde{\boldsymbol{y}}(\omega_q)) \Big\}_{q=1}^{\operatorname{card}\{\mathcal{D}_n\}} \,, \tag{5.22}$$

$$\boldsymbol{z}(i) = \left[\boldsymbol{z}_1^\top(i), \ldots, \boldsymbol{z}_N^\top(i) \right]^\top, \quad [\boldsymbol{z}_m(i)]_q = \left. \frac{\partial \kappa(\widetilde{\boldsymbol{y}}(i), \widetilde{\boldsymbol{y}}(\omega_q))}{\partial y_m(\omega_q)} \right|_{q=1,\ldots,\operatorname{card}\{\mathcal{D}_n\}} \,, \tag{5.23}$$

$$\boldsymbol{\ell}_m(i) = \left[\boldsymbol{\ell}_{1,m}^\top(i), \ldots, \boldsymbol{\ell}_{N,m}^\top(i) \right]^\top, \quad [\boldsymbol{\ell}_{m_1,m_2}(i)]_q = \left. \frac{\partial^2 \kappa(\widetilde{\boldsymbol{y}}(i), \widetilde{\boldsymbol{y}}(\omega_q))}{\partial y_{m_1}(\omega_q) \partial y_{m_2}(i)} \right|_{q=1,\ldots,\operatorname{card}\{\mathcal{D}_n\}} \,. \tag{5.24}$$

The quantities (5.22), (5.23), and (5.24) can be computed in closed form when an explicit expression of the continuously differentiable kernel $\kappa(\cdot,\cdot)$ is chosen.

One difficulty with problem (5.21) is that the summation of $\boldsymbol{t}_m(p)$, $p = 1,\dots,i$ in the regularization term grows linearly with i, and is thus not scalable in this form. Note that we can write the finite-dimensional regularizer in (5.21) as:

$$\boldsymbol{\Omega}(f_n) = \sum_{m=1}^{N} \sqrt{\frac{1}{i}\sum_{p=1}^{i}\left(\boldsymbol{\gamma}^\top \boldsymbol{t}_m(p)\right)^2} = \sum_{m=1}^{N} \sqrt{\boldsymbol{\gamma}^\top \left(\frac{1}{i}\sum_{p=1}^{i}\boldsymbol{T}_m(p)\right)\boldsymbol{\gamma}} = \sum_{m=1}^{N} \sqrt{\boldsymbol{\gamma}^\top \overline{\boldsymbol{T}}_m(i)\boldsymbol{\gamma}}\,, \tag{5.25}$$

with $\boldsymbol{T}_m(p) = \boldsymbol{t}_m(p)\boldsymbol{t}_m^\top(p)$ and $\overline{\boldsymbol{T}}_m(i) = \frac{1}{i}\sum_{p=1}^{i}\boldsymbol{T}_m(p)$, for $m = 1,\dots,N$. Since the following relation is satisfied:

$$\overline{\boldsymbol{T}}_m(i) = \frac{1}{i}\boldsymbol{T}_m(i) + \frac{i-1}{i}\overline{\boldsymbol{T}}_m(i-1)\,, \quad \forall i \geq 2\,, \tag{5.26}$$

we can compute $\overline{\boldsymbol{T}}_m(i)$ recursively for all i with a fixed complexity. In terms of an efficient initialization of the recursive average $\overline{\boldsymbol{T}}_m(1)$, implementing methods such as the Ledoit-Wolf shrinkage estimator [Ledoit and Wolf, 2004] may improve the algorhitm's convergence speed.

Optimization problem (5.21) now becomes:

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}}\, \frac{1}{2}\left|y_n(i) - \boldsymbol{\gamma}^\top \boldsymbol{s}(i)\right|^2 + \eta_n \sum_{m=1}^{N} \sqrt{\boldsymbol{\gamma}^\top \overline{\boldsymbol{T}}_m(i)\boldsymbol{\gamma}}\,. \tag{5.27}$$

Cost function (5.27) can now be optimized iteratively using the subgradient descent update:

$$\hat{\boldsymbol{\gamma}}(i+1) = \hat{\boldsymbol{\gamma}}_{(i)} + \mu_n \boldsymbol{s}(i)\big(y_n(i) - \boldsymbol{s}^\top(i)\hat{\boldsymbol{\gamma}}_{(i)}\big) - \mu_n\eta_n \sum_{m=1}^{N} \frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\,, \tag{5.28}$$

with $\hat{\Lambda}_m(i) = \sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top \overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}$. In (5.27) and (5.28), each Λ_m represents the estimate of the partial derivative of f_n with respect to y_m. The proposed method, summarized in Algorithm 3, has a per-iteration complexity of $\mathcal{O}(N^2)$. Approximate strategies can be considered to obtain a scalable implementation. Parameter τ_n acts as an edge identification threshold. It is used to identify the topology from the estimated coefficients $\hat{\Lambda}_m(i)$, determining whether there exist links from each node $m \in \mathcal{N}_{\setminus n}$ towards n. When processing real data, τ_n can be set as to obtain an estimated topology which realistically explains the studied process, method already successfully applied in works such as [Shen et al., 2019]. Its value can also adjusted as to obtain a connected graph, i.e., a graph in which there exist a path between any node couple.

Using the approximation $\overline{\boldsymbol{T}}_m(i) \approx \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}$: Should $\overline{\boldsymbol{T}}_m(i)$ be computed as the cumulative average of $\boldsymbol{T}_m(i)$, as per (5.26), then we can write:

$$\overline{\boldsymbol{T}}_m(i) = \frac{1}{i}\sum_{\ell=1}^{i}\boldsymbol{T}_m(i) \approx \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}, \quad \text{for large enough } i\,, \tag{5.29}$$

Algorithm 3: Kernel-based online graph inference with partial-derivative-imposed sparsity

Inputs: For every node n: μ, η, $\kappa(\cdot,\cdot)$, ξ_n, and τ_n
Initialization: Set all entries of $\hat{\boldsymbol{\gamma}}(0)$ to 0
Algorithm: At each time instant $i \geq 1$
Update $\mathcal{D}_n$ if $\kappa(\cdot, \widetilde{\boldsymbol{y}}(i))$ satisfies condition (5.19)
Compute $\boldsymbol{s}(i)$ and $\overline{\boldsymbol{T}}_m(i)$ with (5.22), (5.23), (5.24), (5.26)
Update $\hat{\boldsymbol{\gamma}}_{(i)}$ using (5.28)
Set $\hat{a}_{nm}(i)$ to 1 if $\hat{\Lambda}_m(i) \geq \tau_n$, to 0 otherwise

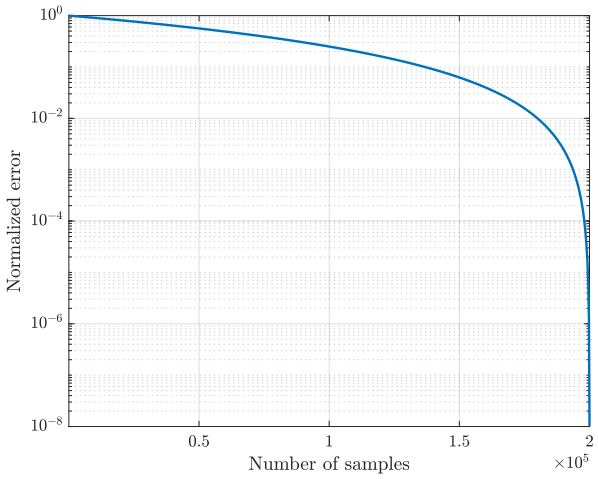


Figure 5.1: Normalized deviation between the empirical average $\overline{\boldsymbol{T}}_m(i)$ and its expectation (i.e., $\left\|\overline{\boldsymbol{T}}_m(i) - \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\right\|_{\mathrm{F}}^2 / \left\|\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\right\|_{\mathrm{F}}^2$) as a function of the number of samples i. Results are averaged for 100 Monte-Carlo runs, under the conditions defined in 5.6.1.1

which is an empirical average of $\boldsymbol{T}_m(i)$. Indeed, if the elements of $\overline{\boldsymbol{T}}_m(i)$ are i.i.d. (with respect to i) and have variance ϑ^2, Chebyshev's inequality implies that [Boucheron et al., 2003]:

$$\mathbb{P}\left(\left|[\overline{\boldsymbol{T}}_m(i)]_{u,v} - [\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}]_{u,v}\right| \geq \epsilon\right) \leq \frac{\vartheta^2}{\epsilon^2 i}\,, \tag{5.30}$$

meaning that the decay of the estimation error is of the order of i^{-1}. An empirical simulation is presented in Fig. 5.1.

5.5 Algorithm analysis

The distributed problem is analyzed in the current section. The dictionary elements are considered as chosen and set beforehand. Consider the local cost function, given dictionary $\mathcal{D}_n$:

$$J_n(\boldsymbol{\gamma}) = \frac{1}{2}\mathbb{E}\left\{\left|y_n(i) - \boldsymbol{\gamma}^\top \boldsymbol{s}(i)\right|^2 + \eta \sum_{m=1}^{N} \sqrt{\frac{1}{i}\sum_{p=1}^{i}\left(\boldsymbol{\gamma}^\top \boldsymbol{t}_m(p)\right)^2}\,\middle|\,\mathcal{D}_n\right\}. \tag{5.31}$$

Its minimum value is:

$$J_{n,\min} \triangleq J_n(\boldsymbol{\gamma}^*) = \mathbb{E}\left\{y_n^2(i)\right\} - \boldsymbol{r}_{sy}^\top \boldsymbol{R}_{ss}^{-1} \boldsymbol{r}_{sy}\,, \tag{5.32}$$

Table 5.2: List of notations and symbols employed throughout the analysis in Chapter 5

Symbol	Equation
$\boldsymbol{\gamma}_0^* = \boldsymbol{R}_{ss}^{-1}\boldsymbol{r}_{sy}$	(5.34)
$\varepsilon_0(i) = y_n(i) - \boldsymbol{s}^\top(i)\boldsymbol{\gamma}^*$	(5.37)
$\boldsymbol{d}_{(i)} \triangleq \hat{\boldsymbol{\gamma}}_{(i)} - \boldsymbol{\gamma}^*$	(5.45)
$\varepsilon(i) = y_n(i) - \boldsymbol{s}^\top(i)\boldsymbol{d}(i) - \boldsymbol{s}^\top(i)\boldsymbol{\gamma}^*$	(5.46)
$J_{n,\min} \triangleq J_n(\boldsymbol{\gamma}^*) = \mathbb{E}\left\{y_n^2(i)\right\} - \boldsymbol{r}_{sy}^\top\boldsymbol{R}_{ss}^{-1}\boldsymbol{r}_{sy}$	(5.32)
$\left[\boldsymbol{K}^{(u,v)}\right]_{ab} = \mathbb{E}\left\{[\boldsymbol{s}(i)]_u[\boldsymbol{s}(i)]_v[\boldsymbol{s}(i)]_a[\boldsymbol{s}(i)]_b\right\}$	(5.70)
$[\boldsymbol{Q}(i)]_{uv} = \mathrm{Tr}\left\{\boldsymbol{K}^{(u,v)}\boldsymbol{D}_{(i)}\right\}$	(5.71)
$\boldsymbol{F}_0 = \boldsymbol{I}_2 - \mu(\boldsymbol{I}\otimes\boldsymbol{R}_{ss} + \boldsymbol{R}_{ss}\otimes\boldsymbol{I}) + \mu^2\boldsymbol{F}_1$	(5.84)
$J_{n,\mathrm{MSE}}(\infty) = J_{n,\min} + \mathrm{Tr}\left\{\boldsymbol{R}_{ss}\boldsymbol{D}_{(\infty)}\right\}$	(5.87)

while its corresponding gradient w.r.t. $\boldsymbol{\gamma}$ is:

$$\nabla_{\boldsymbol{\gamma}} J_n(\boldsymbol{\gamma}) = \mathbb{E}\left\{-y_n(i)\boldsymbol{s}(i) + \boldsymbol{s}(i)\boldsymbol{s}^\top(i)\boldsymbol{\gamma} + \eta\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\boldsymbol{\gamma}}{\Lambda_m(i)}\right\}. \tag{5.33}$$

For the non-regularized case, i.e., $\eta = 0$, we have:

$$\begin{aligned}\nabla_{\boldsymbol{\gamma}} J(\boldsymbol{\gamma}) = 0 &\iff \mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\right\}\boldsymbol{\gamma} = \mathbb{E}\left\{y_n(i)\boldsymbol{s}(i)\right\} \\ &\iff \boldsymbol{\gamma}_0^* = \boldsymbol{R}_{ss}^{-1}\boldsymbol{r}_{sy}\,,\end{aligned} \tag{5.34}$$

with $\boldsymbol{R}_{ss} \triangleq \mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\right\}, \boldsymbol{r}_{sy} \triangleq \mathbb{E}\left\{y_n(i)\boldsymbol{s}(i)\right\}$, and $\boldsymbol{\gamma}_0^*$ representing the optimal value which minimizes cost function (5.31) with $\eta = 0$. Further in the analysis, we consider $\boldsymbol{\gamma}^*$ as being the optimal solution to the same cost function, for $\eta > 0$.

Before proceeding, we establish a set of simplifying hypotheses.

Assumption 5.1. *We assume that vector $\boldsymbol{y}(i)$ is zero-mean and Gaussian with covariance matrix $\boldsymbol{R}_y$, thus its probability density function is:*

$$\theta\big(\boldsymbol{y}(i)\big) = (2\pi)^{-\frac{N}{2}}\det\{\boldsymbol{R}_y\}^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\boldsymbol{y}^\top(i)\boldsymbol{R}_y^{-1}\boldsymbol{y}(i)\right). \tag{5.35}$$

Assumption 5.2. *For reasons of simplicity, we consider $L_m = 1$.*

Assumption 5.3. *We assume the use of the Gaussian kernel.*

It is defined as:

$$\kappa^{\mathrm{G}}(\boldsymbol{a},\boldsymbol{b}) = \exp\left(-\frac{\|\boldsymbol{a}-\boldsymbol{b}\|^2}{2\sigma^2}\right). \tag{5.36}$$

This kernel choice is made due to its capacities as an universal approximator [Liu et al., 2010].

Assumption 5.4. *We assume that $\boldsymbol{s}(i)\boldsymbol{s}^\top(i)$ is statistically independent of $\boldsymbol{d}_{(i)}$.*

This assumption has been successfully used in the analysis of various adaptive filtering algorithms [Parreira et al., 2012], and has been shown in [Minkoff, 2001] to be less restrictive when compared to the classical independence assumption.

We define the optimal estimation error ε_0 as:

$$\varepsilon_0(i) = y_n(i) - \boldsymbol{s}^\top(i)\boldsymbol{\gamma}^* \,. \tag{5.37}$$

Assumption 5.5. *We assume that $\varepsilon_0(i)$ and $\boldsymbol{s}(i)\boldsymbol{s}^\top(i)$ are uncorrelated.*

This assumption is closely related to the Assumption 5.4.

Assumption 5.6. *The elements $\boldsymbol{y}(\omega_p), \forall \omega_p \in \mathcal{I}_n^i$ that compose the dictionary are set a priori, and thus independent from $\boldsymbol{y}(i)$.*

Relating the error in the derivatives to the error in the coefficients: One difficulty related to the update (5.28) is that it considers the evolution of the coefficients $\boldsymbol{\gamma}$, instead of the partial derivatives of the function $\dfrac{\partial \hat{f}_n(\boldsymbol{y})}{\partial y_m}$. Nevertheless, we can study the convergence of the derivatives indirectly by means of filter coefficients. We denote the estimated function f_n by $\hat{f}_n$. Using (5.9) and the Cauchy-Schwarz inequality:

$$\begin{aligned}
\frac{\partial \hat{f}_n(\boldsymbol{y})}{\partial y_m} - \frac{\partial f_n^*(\boldsymbol{y})}{\partial y_m} &= \langle \hat{f}_n, \kappa_{\partial_m}(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}_\kappa} - \langle f_n^*, \kappa_{\partial_m}(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}_\kappa} \\
&\leq \|\kappa_{\partial_m}(\cdot, \boldsymbol{y})\|_{\mathcal{H}_\kappa} \left(\|\hat{f}_n\|_{\mathcal{H}_\kappa} - \|f_n^*\|_{\mathcal{H}_\kappa} \right) ,
\end{aligned} \tag{5.38}$$

leading to:

$$\begin{aligned}
\left| \frac{\partial \hat{f}_n(\boldsymbol{y})}{\partial y_m} - \frac{\partial f_n^*(\boldsymbol{y})}{\partial y_m} \right| &= \left| \langle \hat{f}_n, \kappa_{\partial_m}(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}_\kappa} - \langle f_n^*, \kappa_{\partial_m}(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}_\kappa} \right| \\
&= \left| \langle \hat{f}_n - f_n^*, \kappa_{\partial_m}(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}_\kappa} \right| \\
&\leq \|\kappa_{\partial_m}(\cdot, \boldsymbol{y})\|_{\mathcal{H}_\kappa} \|\hat{f}_n - f_n^*\|_{\mathcal{H}_\kappa} \,.
\end{aligned} \tag{5.39}$$

Moreover, we also have that:

$$\begin{aligned}
\|\hat{f}_n - f_n^*\|_{\mathcal{H}_\kappa} &= \|\boldsymbol{s}^\top \hat{\boldsymbol{\gamma}} - \boldsymbol{s}^\top \boldsymbol{\gamma}^*\| \\
&= \|\boldsymbol{s}^\top (\hat{\boldsymbol{\gamma}} - \boldsymbol{\gamma}^*)\| \\
&\leq \|\boldsymbol{s}\| \, \|\hat{\boldsymbol{\gamma}} - \boldsymbol{\gamma}^*\| \,,
\end{aligned} \tag{5.40}$$

which gives:

$$\left| \frac{\partial \hat{f}_n(\boldsymbol{y})}{\partial y_m} - \frac{\partial f_n^*(\boldsymbol{y})}{\partial y_m} \right| \leq \|\kappa_{\partial_m}(\cdot, \boldsymbol{y})\|_{\mathcal{H}_\kappa} \|\boldsymbol{s}\| \, \|\hat{\boldsymbol{\gamma}} - \boldsymbol{\gamma}^*\| \,. \tag{5.41}$$

Thus, we can bound the error in the estimated derivatives by the error in the coefficients $\boldsymbol{\gamma}$. This allows us to study the convergence behavior of the derivatives indirectly by means of the coefficients $\hat{\boldsymbol{\gamma}}$.

5.5.1 Weight error recursion

Let $\hat{\boldsymbol{\gamma}}_{(i)}$ denote the estimate at step i of $\boldsymbol{\gamma}$. The classic gradient descent step is:

$$\hat{\boldsymbol{\gamma}}_{(i+1)} = \hat{\boldsymbol{\gamma}}_{(i)} - \mu \nabla_{\boldsymbol{\gamma}} J_n(\boldsymbol{\gamma}) \,, \tag{5.42}$$

where μ is a small enough step size. Replacing the gradient via relation (5.33), we obtain the gradient step:

$$\hat{\boldsymbol{\gamma}}_{(i+1)} = \hat{\boldsymbol{\gamma}}_{(i)} + \mu \left(\boldsymbol{r}_{sy} - \boldsymbol{R}_{ss} \hat{\boldsymbol{\gamma}}_{(i)} - \eta \sum_{m=1}^{N} \mathbb{E} \left\{ \frac{\overline{\boldsymbol{T}}_m(i) \hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)} \right\} \right) \,, \tag{5.43}$$

with its stochastic variant:

$$\hat{\boldsymbol{\gamma}}_{(i+1)} = \hat{\boldsymbol{\gamma}}_{(i)} + \mu \boldsymbol{s}(i) \varepsilon(i) - \mu \eta \sum_{m=1}^{N} \frac{\overline{\boldsymbol{T}}_m(i) \hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)} \,, \tag{5.44}$$

where $\varepsilon(i) \triangleq y_n(i) - \boldsymbol{s}^\top(i) \hat{\boldsymbol{\gamma}}_{(i)}$ represents the instantaneous error. We recall that $\hat{\Lambda}_m(i) = \sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top \overline{\boldsymbol{T}}_m(i) \hat{\boldsymbol{\gamma}}_{(i)}}$. Let us denote the difference between the current available estimate and the optimal solution (5.34) by:

$$\boldsymbol{d}_{(i)} \triangleq \hat{\boldsymbol{\gamma}}_{(i)} - \boldsymbol{\gamma}^* \,. \tag{5.45}$$

Error $\varepsilon(i)$ can be expressed in terms of the error vector $\boldsymbol{d}_{(i)}$:

$$\varepsilon(i) = y_n(i) - \boldsymbol{s}^\top(i) \boldsymbol{d}_{(i)} - \boldsymbol{s}^\top(i) \boldsymbol{\gamma}^* \,. \tag{5.46}$$

Replacing (5.46) into relation (5.44) and using (5.37) leads to the following error vector recursion:

$$\boldsymbol{d}_{(i+1)} = \boldsymbol{d}_{(i)} - \mu \boldsymbol{s}(i) \boldsymbol{s}^\top(i) \boldsymbol{d}_{(i)} + \mu \boldsymbol{s}(i) \varepsilon_0(i) - \mu \eta \sum_{m=1}^{N} \frac{\overline{\boldsymbol{T}}_m(i) \left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)}{\sqrt{\left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)^\top \overline{\boldsymbol{T}}_m(i) \left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)}} \,. \tag{5.47}$$

5.5.2 Mean error behavior

The goal of the analysis in the mean is to determine the stability conditions of the algorithm, i.e., the conditions in which the algorithm converges in the mean. As such, we take the expectation of relation (5.47) and employ Assumption 5.4, leading to:

$$\begin{aligned} \mathbb{E}\left\{ \boldsymbol{d}_{(i+1)} \right\} = & \left(\boldsymbol{I} - \mu \boldsymbol{R}_{ss} \right) \mathbb{E}\left\{ \boldsymbol{d}_{(i)} \right\} + \mu \left(\mathbb{E}\left\{ \boldsymbol{s}(i) y_n(i) \right\} - \boldsymbol{R}_{ss} \boldsymbol{\gamma}^* \right) \\ & - \mu \eta \sum_{m=1}^{N} \mathbb{E} \left\{ \frac{\overline{\boldsymbol{T}}_m(i) \left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)}{\sqrt{\left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)^\top \overline{\boldsymbol{T}}_m(i) \left(\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^* \right)}} \right\} \,, \end{aligned} \tag{5.48}$$

with $\boldsymbol{R}_{ss} \triangleq \mathbb{E}\left\{ \boldsymbol{s}(i) \boldsymbol{s}^\top(i) \right\} = \begin{bmatrix} \boldsymbol{R}_{zz} & \boldsymbol{R}_{kz}^\top \\ \boldsymbol{R}_{kz} & \boldsymbol{R}_{kk} \end{bmatrix}$, where $\boldsymbol{R}_{zz} \triangleq \mathbb{E}\left\{ \boldsymbol{z}(i) \boldsymbol{z}^\top(i) \right\}$, $\boldsymbol{R}_{kk} \triangleq \mathbb{E}\left\{ \boldsymbol{k}(i) \boldsymbol{k}^\top(i) \right\}$, and $\boldsymbol{R}_{kz} \triangleq \mathbb{E}\left\{ \boldsymbol{k}(i) \boldsymbol{z}^\top(i) \right\}$. Due do their complexity, the explicit form of block matrix $\boldsymbol{R}_{ss}$ is present in Annex C.1, while the entries of term $\mathbb{E}\left\{ \boldsymbol{s} y_n(i) \right\}$ are developed in Annex C.5.

Approximating the regularization term: For the third term on the r.h.s. of (5.48), we first employ the approximation $\overline{\boldsymbol{T}}_m(i) \approx \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}$, followed by:

$$\mathbb{E}\left\{\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^{\top}\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \approx \frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\mu}}{\sqrt{\mathrm{Tr}\{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\Sigma}\} + \boldsymbol{\mu}^{\top}\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\mu}}}\,, \tag{5.49}$$

with $\boldsymbol{\mu} = \mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\right\}$ and $\boldsymbol{\Sigma} = \mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^{\top}\right\} - \boldsymbol{\mu}\boldsymbol{\mu}^{\top}$. In this case, we successively approximated the expectation of the ratio by the ratio of the expectations, and the expectation of the square root by the square root of the expectation [Elandt-Johnson and Johnson, 1999, p. 70]. Note that $\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^{\top}\right\}$ can be computed as:

$$\begin{aligned}
\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^{\top}\right\} &= \mathbb{E}\left\{[\hat{\boldsymbol{\gamma}}_{(i)} + \boldsymbol{\gamma}^* - \boldsymbol{\gamma}^*][\hat{\boldsymbol{\gamma}}_{(i)} + \boldsymbol{\gamma}^* - \boldsymbol{\gamma}^*]^{\top}\right\} \\
&= \mathbb{E}\left\{[\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^*][\boldsymbol{d}_{(i)} + \boldsymbol{\gamma}^*]^{\top}\right\} \\
&= \mathbb{E}\left\{\boldsymbol{d}_{(i)}\boldsymbol{d}_{(i)}^{\top}\right\} + \mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}(\boldsymbol{\gamma}^*)^{\top} + \boldsymbol{\gamma}^*\mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}^{\top} + \boldsymbol{\gamma}^*(\boldsymbol{\gamma}^*)^{\top}\,.
\end{aligned} \tag{5.50}$$

We remark that approximation (5.49) is successfully used in the theoretical validation, present further in this chapter, in subsection 5.6.1.

Algorithm stability: From (5.48), we can obtain an expression for the error at instant $i+1$:

$$\begin{aligned}
\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\} =& (\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i+1}\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\} + \mu\sum_{\ell=0}^{i}(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\left(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\} - \boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right) \\
&- \mu\eta\sum_{\ell=0}^{i}(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\mathbb{E}\left\{\frac{\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}{\sqrt{\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)^{\top}\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}}\right\}\,.
\end{aligned} \tag{5.51}$$

Taking the norm of both sides of (5.48) and using the triangle and Cauchy-Schwarz inequalities leads to:

$$\begin{aligned}
\left\|\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\}\right\| \leq& \left\|(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i+1}\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\left\|\sum_{\ell=0}^{i}(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\left(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\} - \boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)\right\| \\
&+ \mu\eta\left\|\sum_{\ell=0}^{i}(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\mathbb{E}\left\{\frac{\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}{\sqrt{\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)^{\top}\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}}\right\}\right\| \\
\leq& \left\|(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i+1}\right\|\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\right\|\left\|\left(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\} - \boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)\right\| \\
&+ \mu\eta\sum_{\ell=0}^{i}\left\|(\boldsymbol{I} - \mu\boldsymbol{R}_{ss})^{i-\ell}\right\|\left\|\mathbb{E}\left\{\frac{\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}{\sqrt{\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)^{\top}\overline{\boldsymbol{T}}_m(\ell)\left(\boldsymbol{d}_{(\ell)} + \boldsymbol{\gamma}^*\right)}}\right\}\right\|\,.
\end{aligned} \tag{5.52}$$

The Cholesky decomposition of $\overline{\boldsymbol{T}}_m(\ell)$ is $\overline{\boldsymbol{T}}_m(\ell) = \boldsymbol{C}_m^\top(\ell)\boldsymbol{C}_m(\ell)$. Using it, alongside certain properties of the spectral norm and Jensen's inequality [Jensen, 1906], leads to:

$$
\begin{aligned}
\left\|\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\}\right\| &\leq \left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i+1}\right\| \left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*)\right\| \\
&\quad + \mu\eta\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|\mathbb{E}\left\{\frac{\boldsymbol{C}_m^\top(\ell)\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)}{\sqrt{\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)^\top\boldsymbol{C}_m^\top(\ell)\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)}}\right\}\right\| \\
&= \left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*)\right\| \\
&\quad + \mu\eta\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\left\|\mathbb{E}\left\{\boldsymbol{C}_m^\top(\ell)\frac{\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)}{\left\|\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)\right\|}\right\}\right\| \\
&\leq \left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*)\right\| \\
&\quad + \mu\eta\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\mathbb{E}\left\{\left\|\boldsymbol{C}_m^\top(\ell)\right\|\left\|\frac{\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)}{\left\|\boldsymbol{C}_m(\ell)\left(\boldsymbol{d}_{(\ell)}+\boldsymbol{\gamma}^*\right)\right\|}\right\|\right\} \\
&= \left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*)\right\| \\
&\quad + \mu\eta\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\mathbb{E}\left\{\left\|\boldsymbol{C}_m^\top(\ell)\right\|\right\} \\
&= \left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\sum_{\ell=0}^{i}\left\|(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\right\| \left\|(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*)\right\| \\
&\quad + \mu\eta\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\mathbb{E}\left\{\sqrt{\left\|\boldsymbol{T}_m(\ell)\right\|}\right\}. \qquad (5.53)
\end{aligned}
$$

Since $\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\| \geq 0$, the third term in (5.53) can be upper bounded as:

$$
\begin{aligned}
\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\mathbb{E}\left\{\sqrt{\left\|\boldsymbol{T}_m(\ell)\right\|}\right\} &\leq \sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\max_{0\leq\ell\leq i}\mathbb{E}\left\{\sqrt{\left\|\boldsymbol{T}_m(\ell)\right\|}\right\} \\
&= \frac{\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}-1}{\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|-1}\max_{0\leq\ell\leq i}\mathbb{E}\left\{\sqrt{\left\|\boldsymbol{T}_m(\ell)\right\|}\right\}, \qquad (5.54)
\end{aligned}
$$

while, since we assume that the signals are stationary, for the second term we have:

$$
\begin{aligned}
&\sum_{\ell=0}^{i}\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i-\ell}\left\|\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right\| \\
&= \frac{\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|^{i+1}-1}{\left\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\right\|-1}\left\|\mathbb{E}\left\{\boldsymbol{s}(i)y_n(i)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right\|, \qquad (5.55)
\end{aligned}
$$

leading to the following upper bound for $\left\|\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\}\right\|$:

$$\begin{aligned}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(i+1)}\right\}\right\| \leq \|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\|^{i+1}\left\|\mathbb{E}\left\{\boldsymbol{d}_{(0)}\right\}\right\| + \mu\frac{\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\|^{i+1}-1}{\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\|-1}\left\|\mathbb{E}\left\{\boldsymbol{s}(i)y_n(i)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right\| \\ + \mu\eta\frac{\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\|^{i+1}-1}{\|\boldsymbol{I}-\mu\boldsymbol{R}_{ss}\|-1}\max_{0\leq\ell\leq i}\mathbb{E}\left\{\sqrt{\|\boldsymbol{T}_m(\ell)\|}\right\}. \end{aligned} \tag{5.56}$$

All terms above converge if $\lambda_{\max}(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})<1$, thus, stability in the mean is attained if the stepsize satisfies the following condition:

$$0<\mu<\frac{2}{\lambda_{\max}(\boldsymbol{R}_{ss})}. \tag{5.57}$$

5.5.3 Mean square error behavior

The mean square error analysis allows for the prediction of the behavior of the algorithm under a deterministic model, thus removing the reliance on Monte-Carlo simulations. Moreover, parameters such as the step-size μ and regularization parameter η can be then selected in accordance with the needs of the considered application.

As a first step in the mean square analysis, let us denote $\boldsymbol{D}_{(i)} \triangleq \mathbb{E}\left\{\boldsymbol{d}_{(i)}\boldsymbol{d}_{(i)}^\top\right\}$. Thus, by using (5.47) and Assumption 5.4, we obtain:

$$\begin{aligned}\boldsymbol{D}_{(i+1)} =& \boldsymbol{D}_{(i)} - \mu\left(\boldsymbol{D}_{(i)}\boldsymbol{R}_{ss}+\boldsymbol{R}_{ss}\boldsymbol{D}_{(i)}\right)+\mu^2\boldsymbol{Q}+\mu^2\boldsymbol{N}-\mu^2\,\mathrm{sym}\left\{\boldsymbol{M}\right\}+\mu\,\mathrm{sym}\left\{\boldsymbol{O}\right\} \\ & -\mu\eta\,\mathrm{sym}\left\{\boldsymbol{P}_1\right\}+\mu^2\eta\,\mathrm{sym}\left\{\boldsymbol{P}_2\right\}+\mu^2\eta^2\boldsymbol{P}_3-\mu^2\eta\,\mathrm{sym}\left\{\boldsymbol{P}_4\right\}, \end{aligned} \tag{5.58}$$

with:

$$\boldsymbol{Q}=\mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\boldsymbol{d}_{(i)}\boldsymbol{d}_{(i)}^\top\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\right\}, \tag{5.59}$$

$$\begin{aligned}[\boldsymbol{N}]_{u,v} =& \left[\mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\varepsilon_0^2(i)\right\}\right]_{u,v}=\mathbb{E}\{s_u(i)s_v^\top(i)(y_n(i)-\boldsymbol{s}^\top(i)\boldsymbol{\gamma}^*)^2\} \\ =&\mathbb{E}\left\{s_u(i)s_v(i)y_n^2(i)\right\}-2\sum_p\gamma_p^*\mathbb{E}\left\{s_u(i)s_v(i)s_p(i)y_n(i)\right\} \\ &+\sum_\ell\sum_m\gamma_\ell^*\gamma_m^*\mathbb{E}\left\{s_u(i)s_v(i)s_\ell(i)s_m(i)\right\}, \end{aligned} \tag{5.60}$$

$$\begin{aligned}[\boldsymbol{M}(i)]_{u,v} =& \left[\mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\boldsymbol{d}_{(i)}\boldsymbol{s}^\top(i)\varepsilon_0(i)\right\}\right]_{u,v}=\sum_m\mathbb{E}\left\{s_u(i)s_m(i)v_m(i)s_v(i)\varepsilon_0(i)\right\} \\ =&\sum_m\mathbb{E}\left\{\left[\boldsymbol{d}_{(i)}\right]_m\right\}\mathbb{E}\left\{s_u(i)s_m(i)s_v(i)(y_n(i)-\boldsymbol{s}^\top(i)\boldsymbol{\gamma}^*)\right\} \\ =&\sum_m\mathbb{E}\left\{\left[\boldsymbol{d}_{(i)}\right]_m\right\}\mathbb{E}\left\{s_u(i)s_m(i)s_v(i)y_n(i)\right\} \\ &-\sum_\ell\sum_m\mathbb{E}\left\{\left[\boldsymbol{d}_{(i)}\right]_m\right\}\mathbb{E}\left\{s_u(i)s_m(i)s_v(i)s_\ell(i)\right\}\gamma_b^*, \end{aligned} \tag{5.61}$$

$$\begin{aligned}\boldsymbol{O}(i) &= \mathbb{E}\left\{\boldsymbol{d}_{(i)}\boldsymbol{s}^{\top}(i)\varepsilon_0(i)\right\} = \mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}\mathbb{E}\left\{\boldsymbol{s}^{\top}(i)\varepsilon_0(i)\right\} \\ &= \mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}\left(\mathbb{E}\left\{\boldsymbol{s}^{\top}(i)y_n(i)\right\} - \left(\boldsymbol{\gamma}^*\right)^{\top}\boldsymbol{R}_{ss}\right) \\ &= \mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}\left(\mathbb{E}\left\{\boldsymbol{s}(i)y_n(i)\right\} - \boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)^{\top}, \end{aligned} \tag{5.62}$$

$$\boldsymbol{P}_1 = \mathbb{E}\left\{\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^{\top}\right\}, \tag{5.63}$$

$$\boldsymbol{P}_2 = \mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^{\top}(i)\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^{\top}\right\}, \tag{5.64}$$

$$\boldsymbol{P}_3 = \mathbb{E}\left\{\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]\left[\sum_{p=1}^{N}\frac{\overline{\boldsymbol{T}}_p(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_p(i)}\right]^{\top}\right\}, \tag{5.65}$$

$$\boldsymbol{P}_4 = \mathbb{E}\left\{\boldsymbol{s}(i)\varepsilon_0(i)\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^{\top}\right\}. \tag{5.66}$$

We remark upon the fact that terms (5.61) – (5.62) depend on the mean error term $\mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}$, which is computed using either (5.48) or (5.51). We also note that the third term on the r.h.s. of relation (5.60) is related to term (5.70), which is detailed in Annex C.2. The first two terms on the r.h.s. of the same relation are computed in Annexes C.4 and C.3, respectively.

We now define the MSE at instant i:

$$J_{n,\mathrm{MSE}}(i) \triangleq \mathbb{E}\left\{\left|y_n(i) - \boldsymbol{s}^{\top}(i)\hat{\boldsymbol{\gamma}}_{n(i)}\right|^2\right\}, \tag{5.67}$$

and the MSD at instant i:

$$\mathrm{MSD}(i) \triangleq \mathbb{E}\left\{\left\|\hat{\boldsymbol{\gamma}}_{(i)} - \boldsymbol{\gamma}^*\right\|^2\right\} = \mathbb{E}\left\{\left\|\boldsymbol{d}_{(i)}\right\|^2\right\}. \tag{5.68}$$

Computing any of these performance metrics requires explicit knowledge of quantities (5.59) and (5.63) – (5.66). The remainder of this section focuses on analyzing these terms.

5.5.3.1 Computing matrix $\boldsymbol{Q}$

Let us note $k_s = (N+1)\mathrm{card}\left\{\mathcal{D}_n\right\}$, the total number of entries in block vector $\boldsymbol{s}$. We make use of Assumption 5.4, leading to the writing of the $(u,v)^{\mathrm{th}}$ entry of $\boldsymbol{Q}$ as:

$$[\boldsymbol{Q}]_{uv} = \sum_{a=1}^{k_s}\sum_{b=1}^{k_s}\mathbb{E}\left\{[\boldsymbol{s}(i)]_u[\boldsymbol{s}(i)]_v[\boldsymbol{s}(i)]_a[\boldsymbol{s}(i)]_b\right\}\left[\boldsymbol{D}_{(i)}\right]_{ab}. \tag{5.69}$$

For alleviating the notation, we introduce matrix $\boldsymbol{K}^{(u,v)}$, whose $(a,b)^{\mathrm{th}}$ entry is:

$$\left[\boldsymbol{K}^{(u,v)}\right]_{ab} = \mathbb{E}\left\{[\boldsymbol{s}(i)]_u[\boldsymbol{s}(i)]_v[\boldsymbol{s}(i)]_a[\boldsymbol{s}(i)]_b\right\}. \tag{5.70}$$

Now we can write relation (5.69) as:

$$[\boldsymbol{Q}(i)]_{uv} = \mathrm{Tr}\left\{\boldsymbol{K}^{(u,v)}\boldsymbol{D}_{(i)}\right\} . \tag{5.71}$$

It is important to note that indexes u, v, a, b in relations (5.69) and (5.70) act upon the whole block-vector $\boldsymbol{s}(i)$. As such, it is necessary to identify which particular block $m = 1, \ldots, N$ and specific dictionary entry j – helping in determining ω_j – any of these indexes point to. Knowing the number of entries in the dictionary $\mathcal{D}_n$, this identification is straightforward: let $h = \{u, v, a, b\}$ be a generic index, able to replace any and all of the other indexes. The identification process is then trivially done as:

$$m = \left\lceil \frac{h}{\mathrm{card}\,\{\mathcal{D}_n\}} \right\rceil , \quad j = \mathrm{mod}(h-1, \mathrm{card}\,\{\mathcal{D}_n\}) + 1 . \tag{5.72}$$

After having identified the concerned indexes, each entry of $\boldsymbol{K}^{(u,v)}$ can be explicitly written. Due to their complexity, these derivations are found in Annex C.2.

5.5.3.2 Computing matrix $\boldsymbol{P}_1$

We need to compute:

$$\mathbb{E}\left\{\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} = \sum_{m=1}^{N}\mathbb{E}\left\{\frac{\boldsymbol{d}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} . \tag{5.73}$$

We present it alternatively as:

$$\begin{aligned}\mathbb{E}\left\{\frac{\boldsymbol{d}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} &= \mathbb{E}\left\{\frac{\left[\boldsymbol{d}_{(i)}+\boldsymbol{\gamma}^*-\boldsymbol{\gamma}^*\right]\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \\ &= \mathbb{E}\left\{\frac{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} - \boldsymbol{\gamma}^*\mathbb{E}\left\{\frac{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} .\end{aligned} \tag{5.74}$$

The second term on the r.h.s. of (5.74) can be computed using approximation (5.49). For the first term in the same relation, we employ the following approximation:

$$\begin{aligned}\mathbb{E}\left\{\frac{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} &\approx \mathbb{E}\left\{\frac{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \\ &\approx \frac{\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\right\}\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}^\top}{\sqrt{\mathrm{Tr}\{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\mathbb{E}\{\,\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\}\}}} .\end{aligned} \tag{5.75}$$

In this case, we again successively approximated the expectation of the ratio by the ratio of the expectations, and the expectation of the square root by the square root of the expectation. Note that $\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\right\}$ can be computed using (5.50).

5.5.3.3 Computing matrix $\boldsymbol{P}_2$

By approximating $\overline{\boldsymbol{T}}_m(i) \approx \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}$ and using some algebraic manipulations alongside Assumption 5.4, we obtain:

$$\begin{aligned}\mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} &\approx \mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} \\ &= \mathbb{E}\left\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\right\}\mathbb{E}\left\{\boldsymbol{d}_{(i)}\left[\sum_{m=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} \\ &= \boldsymbol{R}_{ss}\boldsymbol{P}_1\,. \end{aligned} \tag{5.76}$$

5.5.3.4 Computing matrix $\boldsymbol{P}_3$

We successively have:

$$\begin{aligned}\mathbb{E}\left\{\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]\left[\sum_{p=1}^{N}\frac{\overline{\boldsymbol{T}}_p(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_p(i)}\right]^\top\right\} &= \mathbb{E}\left\{\sum_{m=1}^{N}\sum_{p=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\frac{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_p^\top(i)}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_p(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \\ &= \sum_{m=1}^{N}\sum_{p=1}^{N}\mathbb{E}\left\{\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\frac{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_p^\top(i)}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\overline{\boldsymbol{T}}_p(i)\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \\ &\approx \sum_{m=1}^{N}\sum_{p=1}^{N}\mathbb{E}\left\{\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}^\top}{\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}\sqrt{\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}}\right\} \\ &\approx \sum_{m=1}^{N}\sum_{p=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\right\}\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}^\top}{\sqrt{\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}\hat{\boldsymbol{\gamma}}_{(i)}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}\right\}}}\,, \end{aligned} \tag{5.77}$$

for which we successively approximated the expectation of the ratio by the ratio of the expectations, and the expectation of the square root by the square root of the expectation. The numerator is straightforward to compute. The denominator is the expectation of a product of quadratic forms in Gaussian random variables, equal to [Kumar, 1973, Bao and Ullah, 2010]:

$$\begin{aligned}&\mathbb{E}\left\{\hat{\boldsymbol{\gamma}}^\top(i)\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}(i)\hat{\boldsymbol{\gamma}}^\top(i)\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\hat{\boldsymbol{\gamma}}(i)\right\} \\ &\quad= \left(\boldsymbol{\mu}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\mu} + \mathrm{Tr}\left\{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\Sigma}\right\}\right)\left(\boldsymbol{\mu}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\boldsymbol{\mu} + \mathrm{Tr}\left\{\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\boldsymbol{\Sigma}\right\}\right) \\ &\qquad+ 4\boldsymbol{\mu}^\top\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\Sigma}\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\boldsymbol{\mu} + 2\mathrm{Tr}\left\{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\boldsymbol{\Sigma}\mathbb{E}\left\{\overline{\boldsymbol{T}}_p(i)\right\}\boldsymbol{\Sigma}\right\}\,, \end{aligned} \tag{5.78}$$

where $\boldsymbol{\mu} = \mathbb{E}\left\{\hat{\boldsymbol{\gamma}}(i)\right\}$ and $\boldsymbol{\Sigma} = \mathbb{E}\left\{\hat{\boldsymbol{\gamma}}(i)\hat{\boldsymbol{\gamma}}^\top(i)\right\} - \boldsymbol{\mu}\boldsymbol{\mu}^\top$.

5.5.3.5 Computing matrix $\boldsymbol{P}_4$

By approximating $\overline{\boldsymbol{T}}_m(i) \approx \mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}$ and using Assumption 5.4, we obtain:

$$\begin{aligned}\mathbb{E}\left\{\boldsymbol{s}(i)\varepsilon_0(i)\left[\sum_{m=1}^{N}\frac{\overline{\boldsymbol{T}}_m(i)\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} &\approx \mathbb{E}\left\{\boldsymbol{s}(i)\varepsilon_0(i)\left[\sum_{m=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} \\ &= \mathbb{E}\left\{\boldsymbol{s}(i)\varepsilon_0(i)\right\}\mathbb{E}\left\{\left[\sum_{m=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\} \\ &= \left(\mathbb{E}\left\{\boldsymbol{s}(i)y_n(i)\right\} - \boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)\mathbb{E}\left\{\left[\sum_{m=1}^{N}\frac{\mathbb{E}\left\{\overline{\boldsymbol{T}}_m(i)\right\}\hat{\boldsymbol{\gamma}}_{(i)}}{\hat{\Lambda}_m(i)}\right]^\top\right\}.\end{aligned} \tag{5.79}$$

Entries of the quantity $\mathbb{E}\left\{\boldsymbol{T}_m(i)\right\}$ are expressed in Annex C.6.

With results (5.71) – (5.79), all the terms in the recursive equation for $\boldsymbol{D}_{(i)}$ (5.58) are now explicited, allowing the computation of both the MSE and MSD.

5.5.3.6 Case when $\eta = 0$

When there is no regularization, the recursion is:

$$\boldsymbol{D}_{(i+1)} = \boldsymbol{D}_{(i)} - \mu\left(\boldsymbol{D}_{(i)}\boldsymbol{R}_{ss} + \boldsymbol{R}_{ss}\boldsymbol{D}_{(i)}\right) + \mu^2\boldsymbol{Q} + \mu^2\boldsymbol{N} - \mu^2\,\mathrm{sym}\left\{\boldsymbol{M}\right\} + \mu\,\mathrm{sym}\left\{\boldsymbol{O}\right\}. \tag{5.80}$$

In this case, the second-order weight moments relate to the MSE through [Haykin, 2002, p. 268]:

$$J_{n,\mathrm{MSE}}(i) = J_{n,\min} + \mathrm{Tr}\left\{\boldsymbol{R}_{ss}\boldsymbol{D}_{(i)}\right\}, \tag{5.81}$$

and to the MSD through:

$$\mathrm{MSD}(i) = \mathrm{Tr}\left\{\boldsymbol{D}_{(i)}\right\}. \tag{5.82}$$

In order to compute the steady-state MSD, we stack columns of $\boldsymbol{D}_{(i)}$ on top of each other, i.e., $\bar{\boldsymbol{d}}_{(i)} = \mathrm{vec}\left\{\boldsymbol{D}_{(i)}\right\}$. Making use of the properties of the vectorization operator, we obtain the following for recursion (5.80):

$$\bar{\boldsymbol{d}}_{(i+1)} = \boldsymbol{F}_0\bar{\boldsymbol{d}}_{(i)} + \mu^2\bar{\boldsymbol{p}}(i), \tag{5.83}$$

where $\bar{\boldsymbol{p}}(i) = \mathrm{vec}\left\{\boldsymbol{N} - \mathrm{sym}\left\{\boldsymbol{M}(i)\right\} + \mu^{-1}\,\mathrm{sym}\left\{\boldsymbol{O}(i)\right\}\right\}$, and:

$$\boldsymbol{F}_0 = \boldsymbol{I}_2 - \mu(\boldsymbol{I}\otimes\boldsymbol{R}_{ss} + \boldsymbol{R}_{ss}\otimes\boldsymbol{I}) + \mu^2\boldsymbol{F}_1. \tag{5.84}$$

We remark upon the fact that the identity matrix $\boldsymbol{I}_2$ is of size $k_s^2 \times k_s^2$, while $\boldsymbol{I}$ is if size $k_s \times k_s$. Also, entries of the matrix $\boldsymbol{F}_1$ are $[\boldsymbol{F}_1]_{u+(v-1)k_s,a+(b-1)k_s} = \left[\boldsymbol{K}^{(u,v)}\right]_{ab}$.

Steady-state performance: Assuming a small enough step size μ, i.e., one which verifies condition (5.57), then we have, via relation (5.51), that:

$$\begin{aligned}\lim_{i\to\infty}\mathbb{E}\left\{\boldsymbol{d}_{(i)}\right\}=\boldsymbol{d}_{(\infty)}=&\mathbf{0}+\mu\sum_{\ell=0}^{i}(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})^{i-\ell}\left(\mathbb{E}\left\{\boldsymbol{s}(\ell)y_n(\ell)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)\\ =&\mu\frac{\boldsymbol{I}}{\boldsymbol{I}-(\boldsymbol{I}-\mu\boldsymbol{R}_{ss})}\left(\mathbb{E}\left\{\boldsymbol{s}(i)y_n(i)\right\}-\boldsymbol{R}_{ss}\boldsymbol{\gamma}^*\right)\\ =&\mathbf{0}\left(\boldsymbol{R}_{ss}^{-1}\boldsymbol{r}_{sy}-\boldsymbol{\gamma}_0^*\right)\overset{(5.34)}{=}\mathbf{0}\,.\end{aligned}\tag{5.85}$$

Thus, replacing $\boldsymbol{d}_{(\infty)}$ into $\boldsymbol{M}(i)$ and $\boldsymbol{O}(i)$ yields $\boldsymbol{M}(\infty)$ and $\boldsymbol{O}(\infty)$, respectively. Since $\boldsymbol{d}_{(\infty)}=\mathbf{0}$, given (5.61) – (5.62), we have that $\boldsymbol{M}(\infty)=\boldsymbol{O}(\infty)=\mathbf{0}$. In turn, these quantities give $\bar{\boldsymbol{p}}(\infty)=\operatorname{vec}\left\{\boldsymbol{N}-\operatorname{sym}\left\{\boldsymbol{M}(\infty)\right\}+\mu^{-1}\operatorname{sym}\left\{\boldsymbol{O}(\infty)\right\}\right\}=\operatorname{vec}\left\{\boldsymbol{N}\right\}$. The algorithm is mean-square stable as $i\to\infty$, and converges towards:

$$\lim_{i\to\infty}\bar{\boldsymbol{d}}_{(i)}=\mu^2\left(\boldsymbol{I}_2-\boldsymbol{F}_0\right)^{-1}\bar{\boldsymbol{p}}(\infty)=\bar{\boldsymbol{d}}_{(\infty)}\,.\tag{5.86}$$

Using relations (5.81) – (5.82) and the matrix form $\boldsymbol{D}_{(\infty)}$ of $\bar{\boldsymbol{d}}_{(\infty)}$, i.e., $\boldsymbol{D}_{(\infty)}=\operatorname{vec}^{-1}\left\{\bar{\boldsymbol{d}}_{(\infty)}\right\}$, the steady-state MSE is given by:

$$J_{n,\mathrm{MSE}}(\infty)=J_{n,\min}+\operatorname{Tr}\left\{\boldsymbol{R}_{ss}\boldsymbol{D}_{(\infty)}\right\}\,,\tag{5.87}$$

while the steady-state MSD is:

$$\mathrm{MSD}(\infty)=\operatorname{Tr}\left\{\boldsymbol{D}_{(\infty)}\right\}\,.\tag{5.88}$$

5.6 Theoretical validation and experimental results

5.6.1 Theoretical validation

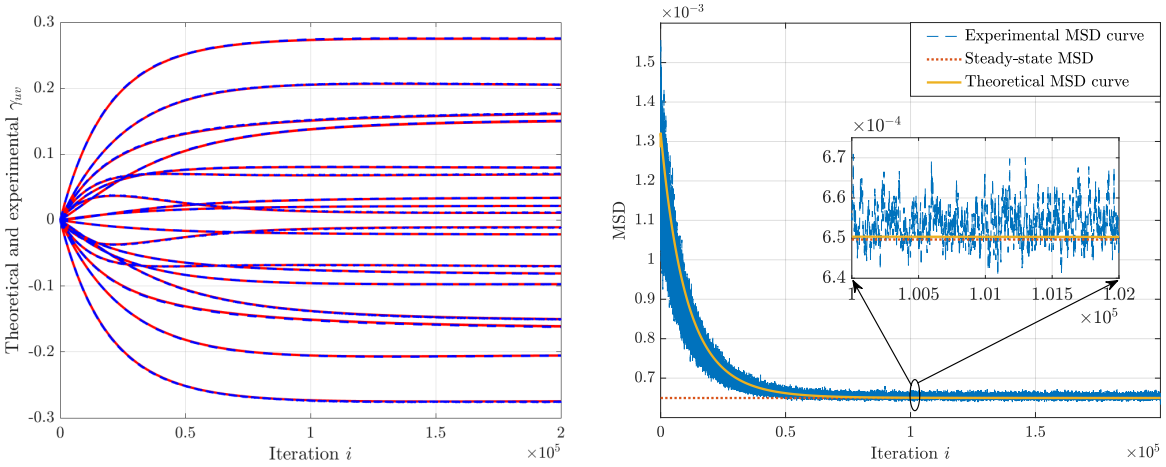
The conditions of the validation and the data generation methods are the same as those enumerated in subsection 4.5.1, except for the dictionary $\mathcal{D}_n$, whose six elements were chosen uniformly from the grid $[-1,1]\times[-1,1]$. We are splitting this subsection in two parts, one with the results where there is no regularization, i.e., $\eta=0$, and one where the regularization term is present, i.e., $\eta>0$.

5.6.1.1 Parameter $\eta=0$

The optimal solution in the non-regularized case is $\boldsymbol{\gamma}_0^*$, computed via (5.34). Fig. 5.2a shows both the theoretical and experimental values for coefficients γ_{uv}. Fig. 5.2b shows both the theoretical and experimental MSD curves, as well as the steady-state MSD. The experimental MSD was computed using:

$$\mathrm{MSD}(i)=\mathbb{E}\left\{\left\|\hat{\boldsymbol{\gamma}}_{(i)}-\boldsymbol{\gamma}^*\right\|^2\right\}\,.\tag{5.89}$$

The theoretical curves are shown to be closely following the theoretical ones, in both the mean and mean square sense. They also validate the approximation (5.49) of the regularization term present in update (5.47).



(a) Theoretical and experimental entries of $\boldsymbol{\gamma}$. Blue dashed lines represent the experimental curves, while red lines represent the theoretical ones

(b) Experimental, steady-state, and theoretical MSD

Figure 5.2: Validation for the analysis in both the mean and mean square sense, for $\eta = 0$

5.6.1.2 Parameter $\eta > 0$

The conditions of the experiment are the same as for the previous case, with $\eta = 1 \cdot 10^{-4}$. The quantity $\overline{\boldsymbol{T}}_m$ is computed *a priori*, using the cumulative average (5.26). As depicted in Fig. 5.1, it converges towards a satisfying average of $\boldsymbol{T}_m$. The simulation curves are presented in Fig. 5.3. The optimal solution for the regularized case $\boldsymbol{\gamma}^*$ was computed using the Matlab CVX package [Grant and Boyd, 2014, 2008]. The experimental MSD curve was computed with (5.89). As for the previous case, the theoretical curves fit the experimental ones. This serves to validate the approximations employed in the analysis, for terms such as $\boldsymbol{P}_1$ (5.74) – (5.75) and $\boldsymbol{P}_3$ (5.77). Moreover, these curves also validate the simplifying hypotheses made before proceeding with the analysis.

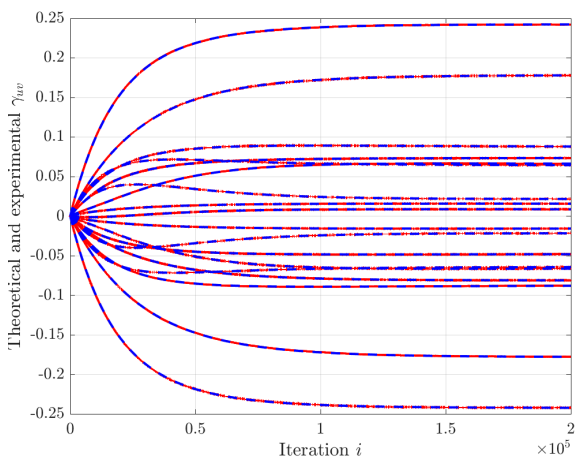
5.6.2 Experimental results

Experimental setup: In this section, the performance of the proposed method is evaluated by two experiments: one considering synthetic data, and another considering real biomedical epilepsy data. The Gaussian kernel is defined as:

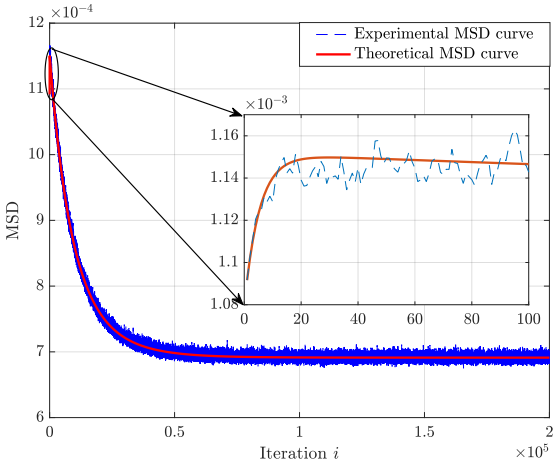
$$\kappa^{\mathrm{G}}(\boldsymbol{a}, \boldsymbol{b}) = \exp\left(-\frac{\|\boldsymbol{a} - \boldsymbol{b}\|^2}{2\sigma_n^2}\right), \quad \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^N, \tag{5.90}$$

where σ_n represents the kernels' band-width, and was used in both cases.

Discretized Lorenz Attractor: Let us consider the Discretized Lorenz Attractor experiment from section 4.5. The additive-model-based algorithm developed in Chapter 4 and employed in the aforementioned section is hereafter called RA for *reference algorithm*. The Lorenz system



(a) Theoretical and experimental entries of $\boldsymbol{\gamma}$. Blue dashed lines represent the experimental curves, while red lines represent the theoretical ones

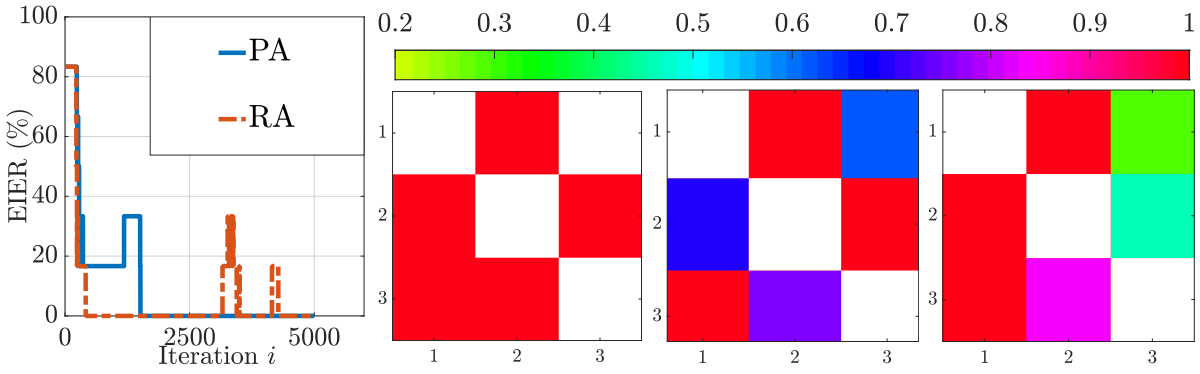


(b) Experimental and theoretical MSD

Figure 5.3: Validation for the analysis in both the mean and mean square sense, for $\eta = 1 \cdot 10^{-4}$

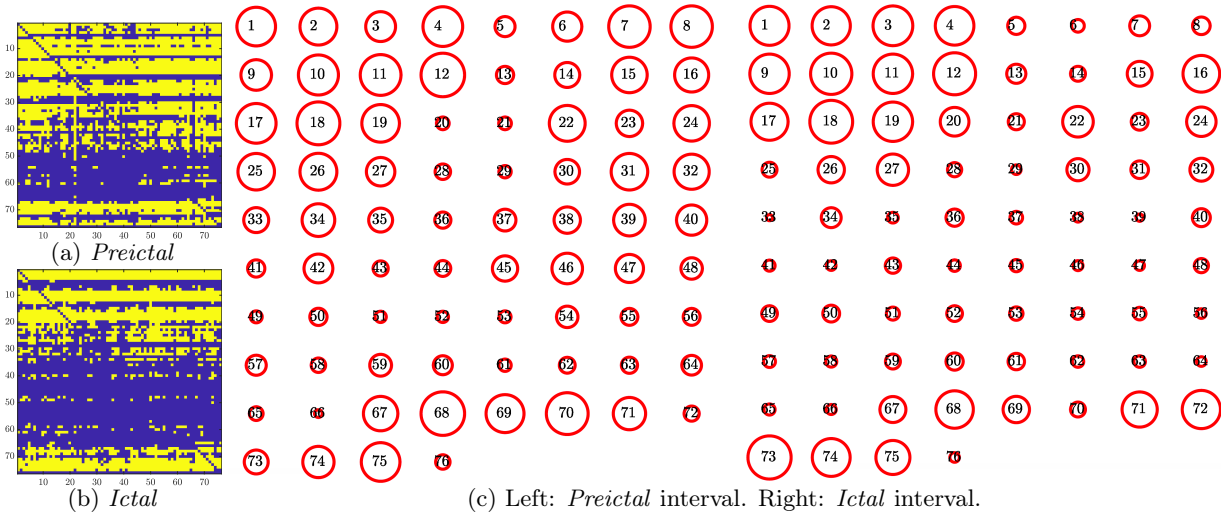
parameters were set to the same values as in (4.61), with initial conditions $[y_1(0), y_2(0), y_3(0)]^\top = [10^{-10}, 10^{-10}, 10^{-10}]^\top$ for both RA and the proposed Algorithm 3 , called PA. This system contains nonlinear interactions which cannot be completely characterized by additive models. We set μ_n and τ_n as to achieve the fastest convergence. Parameters σ_n and ξ_n were set as to obtain the same number of dictionary entries per node, nine, in order to achieve a meaningful comparison between the two methods. Finally, τ_n were set as to achieve the best performance in terms of EIER, previously defined in (4.62), where $\boldsymbol{A}_{\text{gt}}$ is the ground truth depicted in Fig. 5.4b. Fig. 5.4a depicts the EIER of both algorithms as a function of the iterations. In Fig. 5.4c and Fig. 5.4d, each entry represents the mean of the normalized $\hat{\Lambda}_m(i)$, per n, over $i = 5000$ iterations, which encodes the strength of a link from node m towards n before thresholding. It is desired that the amplitude of the elements for which $a_{nm} = 1$ be larger and as separated as possible from the amplitude of the elements for which $a_{nm} = 0$, since this makes distinguishing the active and inactive links easier. On the first row, the more general model of the current method is able to better differentiate between the absence and presence of a link. This is seen through the larger difference between $\hat{\Lambda}_2$ and $\hat{\Lambda}_3$ corresponding to $\hat{a}_{12} = 1$ and $\hat{a}_{13} = 0$, respectively, in the case of PA, while for the RA the strength of the link corresponding to $\hat{a}_{13}$ is much closer to that of the active link. The lower performance of RA was expected since it constrains the interactions to obey a more constraining additive model.

Tests on epilepsy seizure data: The data for this experiment come from a 39-year-old female subject suffering from intractable epilepsy. The data acquisition and pre-processing information is provided in [Kramer et al., 2008]. The data set contains 8 instances of electrocorticography (ECoG) time series, each instance representing one seizure and contains voltage measurements



(a) EIER (b) $\boldsymbol{A}_{\text{gt}}$ (c) RA (d) PA

Figure 5.4: EIER, ground truth and estimates. White represents 0



(a) *Preictal* (b) *Ictal* (c) Left: *Preictal* interval. Right: *Ictal* interval.

Figure 5.5: Estimated adjacency matrices (left). Summed in- and out-degrees for the estimated graphs (right). The larger the radius corresponding to n, the larger the summed degree of n

from 76 different regions on and inside the brain, during the 10 seconds before the epilepsy seizure (*preictal* interval) and the first 10 seconds during the seizure (*ictal* interval). Further information on these data and on epilepsy can be found in subsection 4.5.2. We set $\mu_n = 10^{-3}, \eta_n = 10^2, \xi_n = 0.8$ for each n. Since there was no ground truth, parameters σ_n and τ_n were set as to obtain results coherent with previous works and existing medical studies.

Fig. 5.5a and Fig. 5.5b show the estimated connectivity of the brain, for each interval, averaged over the 8 instances. Fig. 5.5c depicts the degree (sum of in- and out-degree), encoded in the radii of the circles, relative to each interval. Interestingly, our online estimate reveals roughly the same behavior before and during the seizure as the estimate obtained using the method batch developed in [Shen et al., 2019] and the online method in [Moscu et al., 2020a]. More precisely, the number of total connections decreases from one interval to the other, especially due to the variation of in-degrees for nodes 30 to 50. Further analyzing the connections, nodes 75 and 76

Table 5.3: Metrics for the estimated topologies concerning the *preictal* and *ictal* intervals

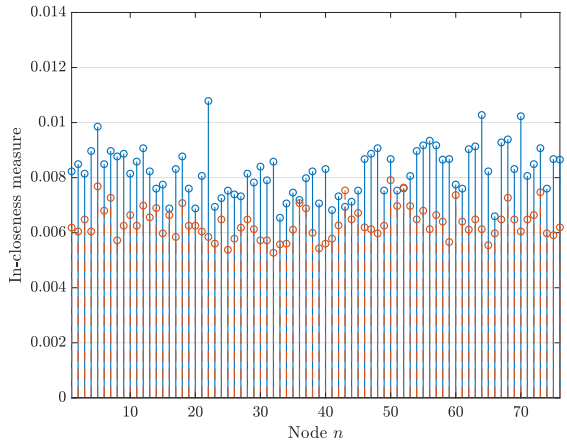
Metric	*Preictal* interval	*Ictal* interval
Network density	0.498	0.370
Average in- (and out-) degree	37.37	27.71
Average path length	1.467	1.559
Average in-closeness	$8.219 \cdot 10^{-3}$	$6.350 \cdot 10^{-3}$
Average out-closeness	$8.710 \cdot 10^{-3}$	$6.895 \cdot 10^{-3}$
Average betweenness	31.39	30.91

have a small in-degree, however they present a more important out-degree. Observe the decrease of the degree of node 26 or the major increase for node 73. This behavior is consistent with the findings of the aforementioned papers. Moreover, as stated in subsection 4.5.2, the average path length tends to be greater in the *ictal* interval, behavior illustrated by our estimates as well. See Table 5.3 and Fig. 5.6 for more metrics and details. An interesting observation we can make concerning this figure is that the low betweenness measure for nodes 50 – 65 indicates their relatively decreased importance in the flow of information in the network. The same behavior has been showcased in Fig. 4.5 by nodes 56 – 65, where the topology was estimated by the algorithm developed in the previous chapter. The same nodes 50 – 65 also exhibit a low *hub* authority measure, fact which only serves to emphasize that their influence in the network is relatively low. We pay particular attention to nodes 22 and 73: the first displays unusually high betweenness and *PageRank* measures in the *preictal* interval, while the same measures are high for the second in the *ictal* interval. This indicates that these two nodes are important transit points for the information in the network, each in one of the two intervals.

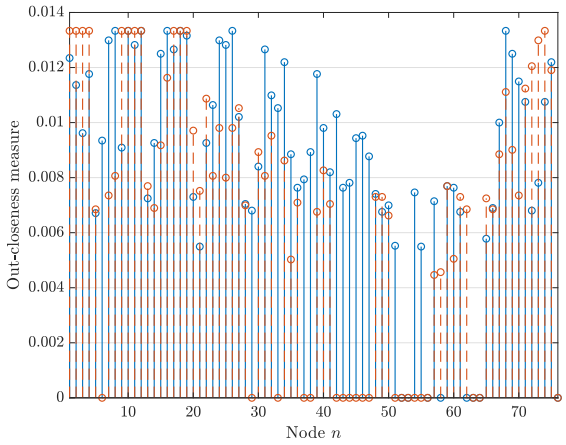
The proposed algorithm is therefore able to obtain results similar to those obtained in previous works, as well as results reported in works of the medical field. This goes to show that the general nonlinear data model proposed in this chapter is pertinent and is able to capture hidden interactions in both synthetic and real data. For reference, the number of kernel functions inserted in the dictionaries was at most 9, after 4000 samples. These results show how a small number of kernel functions are actually needed in order to obtain a satisfactory topology estimate. This fact, alongside the online approach and general model, can translate in reduced computational complexity due to the drastically reduced number of necessary kernel functions.

5.7 Conclusion

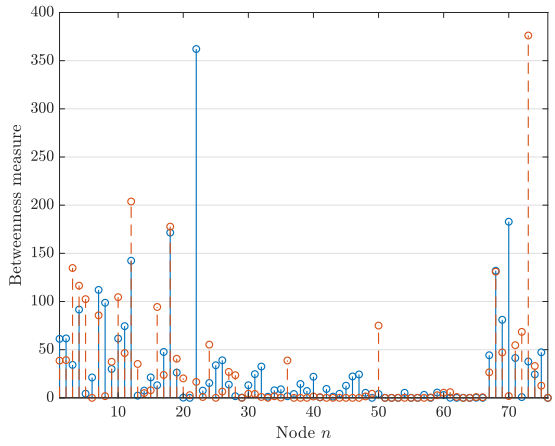
In this chapter, a new kernel-based online topology estimation method was proposed accounting for general nonlinear interactions between the agents in a network. While previous works and the previous chapter only considered models based on additive interactions between the signals at the different nodes, the proposed method surpasses this possibly limiting aspect. Also, such a simplifying additive model is not entirely justified in many practical applications. Following



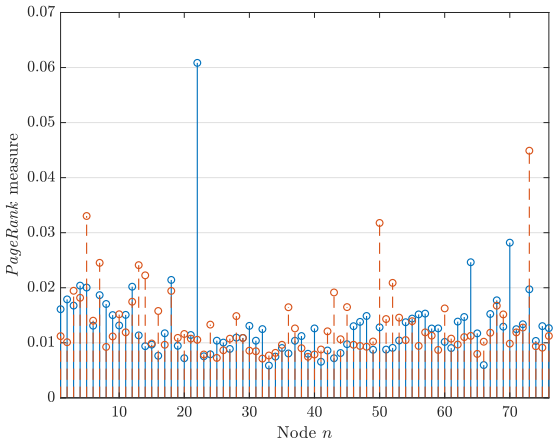
(a) The in-closeness centrality measure per node



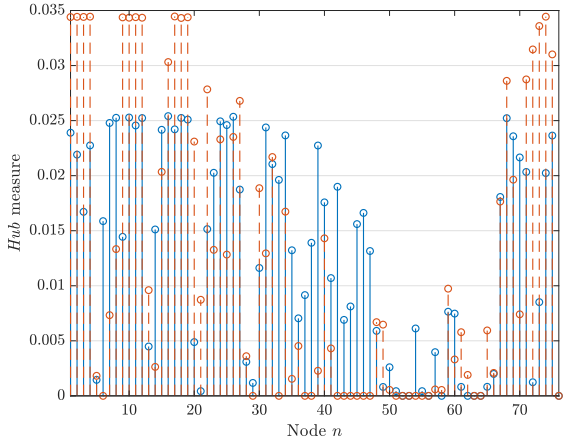
(b) The out-closeness centrality measure per node



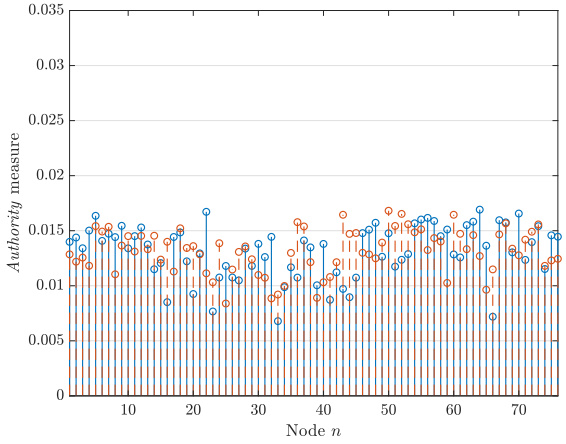
(c) The betweenness centrality measure per node



(d) The *PageRank* centrality measure per node



(e) The *hub* centrality measure per node



(f) The *authority* centrality measure per node

Figure 5.6: Various centrality measures per node, which are indicators of node importance within the graph. See Annex B for details on these measures. Blue continuous lines pertain to the *preictal*, while red dashed lines pertain to the *ictal* interval

a more general approach, we consider arbitrary nonlinear interactions between the nodes, which render our model much more general. By encoding links as partial derivatives of the nonlinear functions, we are able to benefit from the kernel *machinery* framework to estimate a possibly directed, sparse adjacency matrix. An online algorithm is proposed, using kernel dictionaries and recursive computations of the regularization terms to operate with bounded complexity. A complex analysis of the algorithm is provided, as well as performance bounds and conditions. The experimental results, as well as the algorithm analysis, indicate the proposed method can lead to more accurate estimates for more general nonlinear systems.

CHAPTER 6

Conclusion and possible research directions

Contents

As acquiring, processing and storing data become a more and more challenging, algorithms develop new capacities in order to keep up. The quantity and variety of data has only increased, with studies showing that, for the year 2013, 90% of the data had been generated only over the past two years [SINTEF, 2013]. Moreover, recent estimates place the data generation rate at over $2.5 \cdot 10^{18}$ bytes per day [Marr, 2018], data stemming from areas such as blogs, shopping sites, media streaming, just to name a few. This rhythm is only increasing, given the rise and development of the Internet of Things. In medicine, data processing tools such as clustering or classification are commonplace among efforts towards cost-effective disease prevention [Razzak et al., 2020]. Increased ownership of personal devices such as smartphones and smartwatches, able to measure and relay different metrics about the bodies of their wearers, brings the problem of data processing to adapt to reduced resource usage in terms of storage space and energy. These examples represent only a few of the driving forces in the recent developments in the field of Graph Signal Processing, alongside the different tools and solutions that it proposes.

Graphs are versatile and easy to employ in data processing and analysis, usable in different fields of human activity, such as medicine, transportation, and economics. As motivated in Chapter 2, however, many of these applications and tools need knowledge of the actual graph in order to be feasible. As such, the main goal throughout this work was developing algorithms for graph topology estimation, constrained by conditions such as distributivity, adaptability, and online processing. In this final chapter, we summarize the methods and solutions developed for topology inference, and propose new and captivating possible directions for future work.

6.1 Results summary

Pursuing the development of algorithms for graph topology detection led to tackling different approaches and investigating multiple options. These were further constrained by the three main characteristics we considered for our algorithms. The first one was the focus on the online setting, motivated by the continuous flow of data occurring in the real world: medical parameters, road traffic, internet activity. Stemming directly from this setting is the second characteristic, the adaptive capacity. Being able to process every data point quickly and on-the-fly endows the algorithms with the ability to adapt to changes and evolution in the studied network. Thirdly, the distributed aspect: thanks to the intrinsic nature of graphs, processing data can take advantage of distributed computation. In turn, there is a gain in enforcing data privacy [Harrane et al., 2016] and robustness to any local failure of an agent [Mi et al., 2011].

Chapter 3 focused on linear interactions between the agents of a network. The proposed method has the versatility and simplicity necessary for quick data processing. Departing from the fact that each node only needs to exchange data with its one-hop neighbors, the proposed model is based on the principle of causality between signals at different nodes. Moreover, it also gives the possibility to employ any desired regularization, in order to account for any prior knowledge on the estimated graph. The proposed algorithm is also analyzed in both the mean and mean square sense. A successful comparison with another method of the literature emphasizes its qualities, especially concerning the quantity of data necessary to attain a satisfying performance. Other synthetic experiments were run, notably in order to prove the usability of the obtained estimates in a post-processing scenario, such as clustering.

In order to take into account the fact that certain networks interact based on nonlinear relationships, Chapter 4 presents an algorithm able to infer the topology of such a network. The introduced model is based on a presumed additive interaction between nodes. We make use of reproducing kernels and their properties, especially the *kernel trick*. In order to mitigate the impact of the continuously increasing number of data points, due to the online setting, we employ kernel dictionaries, based on the coherence criterion. One of the advantages of such a solution is represented by the extent of the existing literature on the subject. The proposed algorithm is then analyzed in both the mean and mean square sense. Applied on real biomedical data, the method proves itself capable of inferring brain networks which portray the same behavior uncovered by studies in neurology and neuropsychology.

Chapter 5 takes the same nonlinear connectivity premise of the previous chapter, but without presuming any additive behavior. The proposed method is thus conceived to be general and able to encompass any type of interaction. While the same solutions were used for modeling nonlinearities – reproducing kernels – and mitigating the increase in data points – kernel dictionaries, a new solution is employed for introducing sparsity. Based on partial derivatives, it acts as a natural means of quantifying if and how much one node influences another. This novel

algorithm as also analyzed in the mean and mean square sense, including the influence fo the aforementioned sparsity-enforcing regularizer. When compared to the additive model, it displays a better ability of separating links which exist in the network from those which do not. As a final confirmation, it yields satisfactory results on real data as well, verifiable both by the previously established additive model and medical studies.

6.2 Future research directions

One of the most interesting and straightforward focuses for the future is represented by the use of the presented algorithms in a more diversified selection of applications. Indeed, as seen throughout the manuscript, the field of GSP has seen widespread use in many fields of interest for human activity and it would be only natural to tackle more applications than the medical one, which was the main focus of the work.

In order to further observe and analyze the proposed methods, more detailed performance comparisons can be ran. While, undoubtedly, they offer certain advantages, quantifying them in relation with other works remains an aspect to be taken into consideration. In the same vein, an analysis in terms of complexity can also prove beneficial.

In terms of foreseeable improvements to the node-dependent algorithm developed in Chapter 4, one lead is the introduction of a method of automatically selecting the threshold τ_n by making use of the mean square analysis. If there is no link between node m towards n, then the optimal coefficients related to f_m satisfy $\tilde{\boldsymbol{\alpha}}_{nm}^* = \mathbf{0}$. In this case, we have that $\mathbb{E}\left\{\|\widehat{\tilde{\boldsymbol{\alpha}}}_{nm(\infty)} - \tilde{\boldsymbol{\alpha}}_{nm}^*\|^2\right\} = \mathrm{Tr}\left\{\left[\boldsymbol{D}_{(\infty)}\right]_{a\to b,\, a\to b}\right\}$, where $a = 1 + \sum_{\ell=1}^{m-1} L_{m_\ell}$, $b = \sum_{\ell=1}^{m} L_{m_\ell}$. Thus, if the weights corresponding to the linked nodes are not exceedingly small when compared to the coefficient MSD, it is possible to use the model information to identify the links. Assuming that i is sufficiently large such that the coefficients have already converged (i.e., $\boldsymbol{D}_{(i)} \approx \boldsymbol{D}_{(\infty)}$), an automatic decision rule can be established.

Regarding the general algorithm developed in Chapter 5, one improvement is to consider a low-rank decomposition of matrix $\overline{\boldsymbol{T}}_m(i)$, such as $\overline{\boldsymbol{T}}_m(i) = \boldsymbol{U}_m(i)\boldsymbol{U}_m^\top(i)$, with $\boldsymbol{U}_m \in \mathbb{R}^{N\mathrm{card}\{\mathcal{D}\}\times K_{\mathrm{rank}}}$. The reduced size of $\boldsymbol{U}_m$ translates into reduced computational stress, with complexity of the order of $\mathcal{O}(N\mathrm{card}\,\{\mathcal{D}\}\,K_{\mathrm{rank}})$. Indeed, preliminary tests show that the first two largest eigenvalues of matrices $\overline{\boldsymbol{T}}_m(i)$ are three to four orders of magnitudes larger than the rest, indicating that a rank-2 decomposition is indeed possible.

The use of a multi-kernel framework akin to the method presented in [Garrigos et al., 2018, Shen et al., 2019] can endow both nonlinear algorithms with better inference capacities, alongside flexibility in the choice of kernels. Using a selection of predefined kernels, the goal is to learn a combination of these in order to solve the problem at hand. An introduction and analysis into this topic is given in [Jin et al., 2010, Kloft et al., 2011].

On the general line of topology inference through estimating a nonlinear function, the field of

deep learning can provide an alternative [Schwab and Karlen, 2019, Nauta et al., 2019, Lachapelle et al., 2019]. For a given neural network of $J > 2$ layers, the network paths, starting from any input representing node m which is not connected to n, increase exponentially with the number of neurons per layer. A solution to this is *isolating* the decision of node influence to the input layer, while allowing for the rest of the layers to account for the nonlinearity aspect. In doing so, any node m without influence on n, intuitively, gets *cut* from the first layer, without influencing the networks behavior in deeper layers. With these remarks, note that a function $f_n(\boldsymbol{x})$ can be rewritten as the composition of two other functions $f_n(\boldsymbol{x}) = f_n^{\text{deep}} \circ \sigma(\boldsymbol{W}_1\boldsymbol{x} + \boldsymbol{b}_1)$ where $\circ$ denotes the function composition, and $f_n^{\text{deep}}(\cdot)$ models the behavior of layers $\{2, \ldots, J\}$ of the neural network. Matrix $\boldsymbol{W}_1$ and vector $\boldsymbol{b}_1$ collect the weights and biases of the first layer, respectively, while $\sigma(\cdot)$ is the activation function, e.g., ReLU [Hanin, 2019]. The identification can then be done by checking if the norm of the m^{th} column of $\boldsymbol{W}_1$ approaches zero, fact indicative of a lack of influence from node m.

* * *

In line with current research tendencies, the direction of the undertaken work aimed in developing online, distributed, and adaptive topology inference algorithms. Both the comprehensive theoretical analyses and their corresponding experiments showcase the capacities for each proposed method. Moreover, in many cases, the obtained results are also supported by previous research in other fields of study. With many directions to tackle in the future, the field of Graph Signal Processing represents a growing and developing domain, boundless in opportunity.

APPENDIX

A

Complete form of matrix $\boldsymbol{R}_{yy}^{(m_{1\to4})}$

The full form of matrix $\boldsymbol{R}_{yy}^{(m_{1\to4})}$, which intervenes in relation (4.46), is:

$$\boldsymbol{R}_{yy}^{(m_{1\to4})} = \begin{bmatrix} [\boldsymbol{R}_{yy}]_{m_1m_1}\,\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_1}}^\top & [\boldsymbol{R}_{yy}]_{m_1m_2}\,\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_2}}^\top & [\boldsymbol{R}_{yy}]_{m_1m_3}\,\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_3}}^\top & [\boldsymbol{R}_{yy}]_{m_1m_4}\,\mathbb{1}_{L_{m_1}}\mathbb{1}_{L_{m_4}}^\top \\ [\boldsymbol{R}_{yy}]_{m_2m_1}\,\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_1}}^\top & [\boldsymbol{R}_{yy}]_{m_2m_2}\,\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_2}}^\top & [\boldsymbol{R}_{yy}]_{m_2m_3}\,\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_3}}^\top & [\boldsymbol{R}_{yy}]_{m_2m_4}\,\mathbb{1}_{L_{m_2}}\mathbb{1}_{L_{m_4}}^\top \\ [\boldsymbol{R}_{yy}]_{m_3m_1}\,\mathbb{1}_{L_{m_3}}\mathbb{1}_{L_{m_1}}^\top & [\boldsymbol{R}_{yy}]_{m_3m_2}\,\mathbb{1}_{L_{m_3}}\mathbb{1}_{L_{m_2}}^\top & [\boldsymbol{R}_{yy}]_{m_3m_3}\,\mathbb{1}_{L_{m_3}}\mathbb{1}_{L_{m_3}}^\top & [\boldsymbol{R}_{yy}]_{m_3m_4}\,\mathbb{1}_{L_{m_3}}\mathbb{1}_{L_{m_4}}^\top \\ [\boldsymbol{R}_{yy}]_{m_4m_1}\,\mathbb{1}_{L_{m_4}}\mathbb{1}_{L_{m_1}}^\top & [\boldsymbol{R}_{yy}]_{m_4m_2}\,\mathbb{1}_{L_{m_4}}\mathbb{1}_{L_{m_2}}^\top & [\boldsymbol{R}_{yy}]_{m_4m_3}\,\mathbb{1}_{L_{m_4}}\mathbb{1}_{L_{m_3}}^\top & [\boldsymbol{R}_{yy}]_{m_4m_4}\,\mathbb{1}_{L_{m_4}}\mathbb{1}_{L_{m_4}}^\top \end{bmatrix}. \tag{A.1}$$

We note that $\boldsymbol{R}_{yy}^{(m_{1\to4})}$ exhibits a sparser structure should $\boldsymbol{y}(i)$ be i.i.d.. Blocks on the main diagonal then become $[\boldsymbol{R}_{yy}]_{m_jm_j}\,\boldsymbol{I}$, where $\boldsymbol{I}$ is of size $m_j \times m_j$, for $j = 1, 2, 3, 4$. The remaining blocks, which are not necessarily square, are formed similarly, with entries $[\boldsymbol{R}_{yy}]_{m_jm_k}$, $j, k = 1, 2, 3, 4, j \neq k$, on their main diagonals and zero otherwise.

APPENDIX B

Metrics pertaining to directed graphs

We present several graph metrics which can be useful in interpreting the estimated topologies obtained throughout the experiments. For ease of reading, we recall some useful notations in Table B.1. Consider a directed graph $\mathfrak{G}$. We define the following:

Network density: The ratio between the existent number of edges and the total possible number of edges and it is equal to:

$$\rho_{\mathfrak{G}} = \frac{\text{card}\,\{\mathcal{E}\}}{N(N-1)}\,. \tag{B.1}$$

Average in- (out-) degree: It represents the average number of edges going in (out) of the networks' nodes. It is equal to:

$$\sigma_{\mathfrak{G}} = \frac{\text{card}\,\{\mathcal{E}\}}{N}\,. \tag{B.2}$$

Average path length: Applies to an un-weighted graph and is one of the most robust graph metrics [Albert and Barabási, 2002]. It is equal to:

$$\bar{\ell}_{\mathfrak{G}} = \frac{1}{N(N-1)} \sum_{m_1 \neq m_2} d(m_1, m_2)\,, \tag{B.3}$$

where $d(m_1, m_2)$ outputs the shortest path between nodes m_1 and m_2.

In- (out-) closeness centrality: It is computed at every node n and is the inverse of the sum of the distances to (from) the other reachable nodes in the graph, on all paths arriving to (going from) node n. It is computed as:

$$c^{\text{in}}_{\mathfrak{G}}, c^{\text{out}}_{\mathfrak{G}} = \left(\frac{\text{card}\,\{\mathcal{N}_n\}}{N-1}\right)^2 \frac{1}{d_n}\,, \tag{B.4}$$

Table B.1: List of graph-related notations and symbols present in the quantities defined throughout Annex B

Symbol	Definition
$\mathfrak{G}$	A directed graph
$\boldsymbol{A}$	Adjacency matrix of a graph
$\mathcal{E}$	Set of edges of a graph
$\mathcal{N}$	Set of nodes of the graph
$\mathcal{N}_n$	Set of nodes in the neighborhood of node n, excluding node n
N	Total number of nodes in the graph

where d_n is the sum of the number of edges on the shortest paths between n and nodes $m \in \mathcal{N}_n$. These paths can be computed using Dijkstra's algorithm [Dijkstra, 1959].

Betweenness centrality: Computed at a certain node n, it is the ratio between the shortest paths that pass through n and the total number of shortest paths. Thus, it is a measure of importance as an intermediary on paths between other node pairs. A high node betweenness can mean that its removal will sever or isolate other nodes. It is computed as:

$$b_{\mathfrak{G}} = \sum_{m_1, m_2 \neq n} \frac{s^n_{m_1 m_2}}{s_{m_1 m_2}} \,, \tag{B.5}$$

where $s_{m_1 m_2}$ is the total number of shortest paths from m_1 to m_2, while $s^n_{m_1 m_2}$ is the number of shortest paths from m_1 to m_2 which pass through n.

***PageRank* centrality:** Measures the average time spent on a certain node when applying a random walk on the graph [Brin and Page, 1998]. When at a certain node, a successor is chosen randomly with a preset probability. The *PageRank* centrality measure of a node n is:

$$\mathrm{PR}_{\mathfrak{G}} = \lim_{i \to \infty} \frac{1}{i} \sum_{j=1}^{i} \iota_n \left(c(i) \right) \,, \tag{B.6}$$

where $\iota_n(a) = \begin{cases} 0 \,, n = a \\ 1 \,, n \neq a \end{cases}$ is an indicator function, and $c(i)$ is a function whose output is the index of the current node at time i. The supporting principle of this metric is that the rank of a certain node should be high if the rank of other nodes linking towards it are high, and low if the rank of these nodes are low.

A random walk, in accordance with [Grady and Polimeni, 2010, p. 106], is an iterative process which follows a random walker located at a certain node as it moves from node to node, along the edges. At each step, the random walker located at node m_1 will move to a node $m_2 \in \mathcal{N}_{m_1}$ with a probability $q_{m_1 m_2} = \dfrac{a_{m_1 m_2}}{w_{m_1}}$, where w_{m_1} is a possibly weighted degree of node m_1. If p_{m_1} is the probability that the walker is present at node m_1, then we can write the process as:

$$\boldsymbol{p}(i+1) = \boldsymbol{W}^{-1} \boldsymbol{A} \boldsymbol{p}(i) \,, \tag{B.7}$$

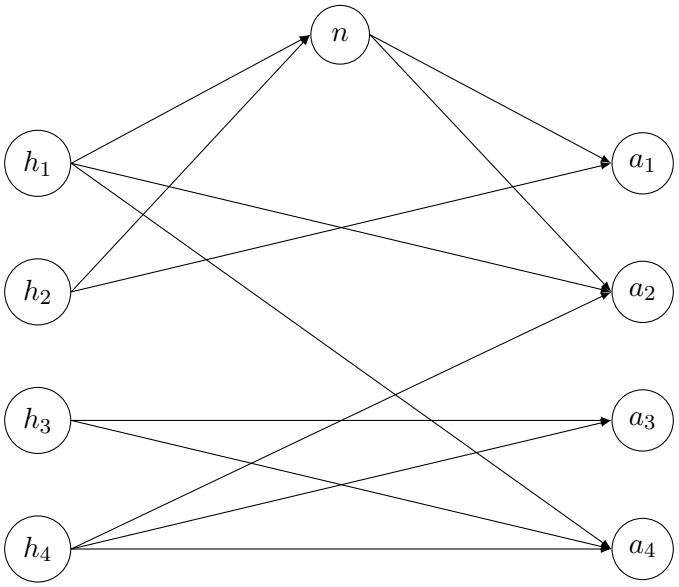


Figure B.1: Example of a *hub-authority* interaction, with *hub* nodes on the left and *authority* nodes on the right. Notice how n is both a *hub* and an *authority*

where $\boldsymbol{p} = \mathrm{col}\,\{\{p_m\}_{m\in\mathcal{N}}\}$.

***Hub-authority* centrality:** Also known as Kleinberg centrality [Kleinberg, 1999], it is a linked measure between a *hub* – node which *points* to multiple other nodes (i.e., the *authorities*), and an *authority* – node towards which many other nodes *point* (i.e., the *hubs*). One node can be both a *hub* and an *authority*. A visual example of such an interaction is in Fig. B.1. The *hub* centrality $H_{(n)}$ of node n is given by:

$$H_{(n)} = c_H \sum_{m\in\mathcal{N}_{\setminus n}} [\boldsymbol{A}]_{mn}\, A_{(n)}\,, \tag{B.8}$$

while the *authority* centrality $A_{(n)}$ of node n is given by:

$$A_{(n)} = c_A \sum_{m\in\mathcal{N}_{\setminus n}} [\boldsymbol{A}]_{nm}\, H_{(n)}\,, \tag{B.9}$$

where c_H and c_A are chosen constants.

We remark upon the fact that, except for the first, all the other listed metrics represent different measures of centrality, which is a node property quantifying its relative importance in the network.

APPENDIX C

Quantities involved in the algorithm analysis

We recall that $\widetilde{\boldsymbol{y}}(i) = [\boldsymbol{y}^\top(i),\, y_n^\top(i)]^\top$. The expectations involved in computing $\boldsymbol{R}_{ss}$, matrices $\boldsymbol{K}^{(u,v)}$ and $\mathbb{E}\{\boldsymbol{T}_m(i)\}$, alongside quantities $\mathbb{E}\left\{s_a(i)s_b(i)y_n^2(i)\right\}$, $\mathbb{E}\left\{s_u(i)s_a(i)s_b(i)y_n(i)\right\}$ and $\mathbb{E}\left\{s_b(i)y_n(i)\right\}$ are expressed generically as[1]:

$$\mathbb{E}\left\{ f(\widetilde{\boldsymbol{y}}) \triangleq (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1}(\widetilde{y}_{h_3} - x_{h_4})^{\iota_2}(\widetilde{y}_{h_5} - x_{h_6})^{\iota_3}(\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \exp\left(-\frac{1}{2}c_4\widetilde{\boldsymbol{y}}^\top\boldsymbol{B}_0\widetilde{\boldsymbol{y}} - \boldsymbol{x}_4^\top\boldsymbol{B}_4^\top\widetilde{\boldsymbol{y}} + \frac{1}{2}\boldsymbol{x}_4^\top\boldsymbol{Q}_4\boldsymbol{x}_4\right)\right\}, \tag{C.1}$$

where $\boldsymbol{x}_4 = [\boldsymbol{y}^\top(\omega_p), \boldsymbol{y}^\top(\omega_q), \boldsymbol{y}^\top(\omega_r), \boldsymbol{y}^\top(\omega_s)]^\top$ represents the fixed dictionary elements. Also, $\boldsymbol{B}_0 = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix}$, $c_4 = \dfrac{1+\sum_{i=1}^3 c_i}{\sigma^2}$, $\boldsymbol{B}_4 = -\dfrac{1}{\sigma^2}\begin{bmatrix} \boldsymbol{I} & c_3\boldsymbol{I} & c_3c_2\boldsymbol{I} & c_3c_2c_1\boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}$, and $\boldsymbol{Q}_4 = -\dfrac{1}{\sigma^2}\begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & c_3\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & c_3c_2\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & c_3c_2c_1\boldsymbol{I} \end{pmatrix}$. In the previous quantities, c_1, c_2, c_3 represent binary selection

[1] Time instants i are dropped in order to alleviate notation.

variables, accounting for the following possible cases:

$$\begin{cases} c_1 = c_2 = c_3 = 1, \text{ for term } \boldsymbol{K}^{(u,v)} \\ c_1 = 0, c_2 = c_3 = 1, \text{ for term } \mathbb{E}\{s_u(i)s_a(i)s_b(i)y_n(i)\} \\ c_1 = c_2 = 0, c_3 = 1, \text{ for terms } \mathbb{E}\{\boldsymbol{T}_m(i)\}, \boldsymbol{R}_{ss} \text{ and } \mathbb{E}\{s_a(i)s_b(i)y_n^2(i)\} \\ c_1 = c_2 = c_3 = 0, \text{ for term } \mathbb{E}\{s_b(i)y_n(i)\} \end{cases} . \tag{C.2}$$

Note that $h_1, h_2, \ldots, h_8$ are different indexes of $\widetilde{\boldsymbol{y}}(i)$ and $\boldsymbol{x}_4$ (not necessarily distinct), and the binary variables $\iota_i \in \{0,1\}$, for $i = 1, \ldots, 4$, allow us to accommodate lower-order cases and be more flexible.

We now have, successively:

$$\begin{aligned} &\mathbb{E}\left\{ f(\widetilde{\boldsymbol{y}}) \triangleq (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \exp\left(-\frac{1}{2} c_4 \widetilde{\boldsymbol{y}}^\top \boldsymbol{B}_0 \widetilde{\boldsymbol{y}} - \boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \widetilde{\boldsymbol{y}} + \frac{1}{2} \boldsymbol{x}_4^\top \boldsymbol{Q}_4 \boldsymbol{x}_4 \right) \right\} \\ &= (2\pi)^{-\frac{N+1}{2}} \det\{\boldsymbol{R}_{\widetilde{y}}\}^{-\frac{1}{2}} \exp\left(\frac{1}{2} \boldsymbol{x}_4^\top \boldsymbol{Q}_4 \boldsymbol{x}_4 \right) \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \\ &\qquad \times \exp\left(-\frac{1}{2} c_4 \widetilde{\boldsymbol{y}}^\top \boldsymbol{B}_0 \widetilde{\boldsymbol{y}} \right) \exp\left(-\frac{1}{2} \widetilde{\boldsymbol{y}}^\top \boldsymbol{R}_{\widetilde{y}}^{-1} \widetilde{\boldsymbol{y}} \right) \exp\left(-\boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \widetilde{\boldsymbol{y}} \right) d\widetilde{y}_1 \cdots d\widetilde{y}_{N+1} \qquad \text{(C.3)} \\ &= (2\pi)^{-\frac{N+1}{2}} \det\{\boldsymbol{R}_{\widetilde{y}}\}^{-\frac{1}{2}} \exp\left(\frac{1}{2} \boldsymbol{x}_4^\top \boldsymbol{Q}_4 \boldsymbol{x}_4 \right) \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \\ &\qquad \times \exp\left(-\frac{1}{2} \widetilde{\boldsymbol{y}}^\top \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right] \widetilde{\boldsymbol{y}} - \boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \widetilde{\boldsymbol{y}} \right) d\widetilde{y}_1 \cdots d\widetilde{y}_{N+1} . \qquad \text{(C.4)} \end{aligned}$$

Note that we can write:

$$\begin{aligned} -\frac{1}{2} \widetilde{\boldsymbol{y}}^\top \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right] \widetilde{\boldsymbol{y}} - \boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \widetilde{\boldsymbol{y}} &= -\frac{1}{2} \left(\widetilde{\boldsymbol{y}}^\top \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right] \widetilde{\boldsymbol{y}} + \boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \widetilde{\boldsymbol{y}} + \widetilde{\boldsymbol{y}}^\top \boldsymbol{B}_4 \boldsymbol{x}_4 \right) \\ = -\frac{1}{2} \left(\widetilde{\boldsymbol{y}} + \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right]^{-1} \boldsymbol{B}_4 \boldsymbol{x}_4 \right)^\top \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right] \left(\widetilde{\boldsymbol{y}} + \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right]^{-1} \boldsymbol{B}_4 \boldsymbol{x}_4 \right) &+ \frac{1}{2} \boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \left[c_4 \boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1} \right]^{-1} \boldsymbol{B}_4 \boldsymbol{x}_4 . \end{aligned} \tag{C.5}$$

Thus, relation (C.4) becomes:

$$
\begin{aligned}
&(2\pi)^{-\frac{N+1}{2}} \det\{\boldsymbol{R}_{\widetilde{y}}\}^{-\frac{1}{2}} \exp\left(\frac{1}{2}\boldsymbol{x}_4^\top \boldsymbol{Q}_4 \boldsymbol{x}_4\right) \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \\
&\quad \times \exp\left(-\frac{1}{2}\left(\widetilde{\boldsymbol{y}} + \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right)^\top \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right] \left(\widetilde{\boldsymbol{y}} + \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right)\right) \\
&\quad \times \exp\left(\frac{1}{2}\boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right) d\widetilde{y}_1 \cdots d\widetilde{y}_{N+1} \\
&= (2\pi)^{-\frac{N+1}{2}} \det\{\boldsymbol{R}_{\widetilde{y}}\}^{-\frac{1}{2}} \exp\left(\frac{1}{2}\boldsymbol{x}_4^\top \boldsymbol{Q}_4 \boldsymbol{x}_4\right) \det\left\{\left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1}\right\}^{\frac{1}{2}} \det\left\{\left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1}\right\}^{-\frac{1}{2}} \\
&\quad \times \exp\left(\frac{1}{2}\boldsymbol{x}_4^\top \boldsymbol{B}_4^\top \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right) \\
&\qquad \int_{\mathbb{R}} \cdots \int_{\mathbb{R}} (\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4} \\
&\qquad \times \exp\left(-\frac{1}{2}\left(\widetilde{\boldsymbol{y}} + \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right)^\top \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right] \left(\widetilde{\boldsymbol{y}} + \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4\right)\right) d\widetilde{y}_1 \cdots d\widetilde{y}_{N+1} \\
&= \det\{\boldsymbol{R}_{\widetilde{y}}\}^{-\frac{1}{2}} \det\left\{\left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1}\right\}^{\frac{1}{2}} \exp\left(\frac{1}{2}\boldsymbol{x}_4^\top \left(\boldsymbol{Q}_4 + \boldsymbol{B}_4^\top \left[c_4\boldsymbol{B}_0 + \boldsymbol{R}_{\widetilde{y}}^{-1}\right]^{-1} \boldsymbol{B}_4\right)\boldsymbol{x}_4\right) \\
&\quad \times \mathbb{E}_{\theta'(\widetilde{\boldsymbol{y}})}\left\{(\widetilde{y}_{h_1} - x_{h_2})^{\iota_1} (\widetilde{y}_{h_3} - x_{h_4})^{\iota_2} (\widetilde{y}_{h_5} - x_{h_6})^{\iota_3} (\widetilde{y}_{h_7} - x_{h_8})^{\iota_4}\right\} = \nu(\{h_i\}_{i=1}^8)\,,
\end{aligned}
\tag{C.6}
$$

where

$$
\theta'(\boldsymbol{y}) = \mathfrak{N}\left(\boldsymbol{\mu} = -\left[c_4\boldsymbol{I} + \boldsymbol{R}_y^{-1}\right]^{-1} \boldsymbol{B}_4\boldsymbol{x}_4, \boldsymbol{\Sigma} = \left[c_4\boldsymbol{I} + \boldsymbol{R}_y^{-1}\right]^{-1}\right)\,. \tag{C.7}
$$

Let us focus on the expectation. We have, for $\iota_i = 1, i = \{1,2,3,4\}$:

$$\begin{aligned}
&\mathbb{E}_{\theta'(\boldsymbol{y})}\left\{(\widetilde{y}_{h_1} - x_{h_2})^{\iota_1}(\widetilde{y}_{h_3} - x_{h_4})^{\iota_2}(\widetilde{y}_{h_5} - x_{h_6})^{\iota_3}(\widetilde{y}_{h_7} - x_{h_8})^{\iota_4}\right\} \\
&= \mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\widetilde{y}_{h_5}\widetilde{y}_{h_7}\right\} - x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\widetilde{y}_{h_5}\right\} - x_{h_6}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\widetilde{y}_{h_7}\right\} + x_{h_6}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\right\} \\
&- x_{h_4}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_5}\widetilde{y}_{h_7}\right\} + x_{h_4}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_5}\right\} + x_{h_4}x_{h_6}\mathbb{E}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_7}\right\} - x_{h_4}x_{h_6}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_1}\right\} \\
&- x_{h_2}\mathbb{E}\left\{\widetilde{y}_{h_3}\widetilde{y}_{h_5}\widetilde{y}_{h_7}\right\} + x_{h_2}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_3}\widetilde{y}_{h_5}\right\} + x_{h_2}x_{h_6}\mathbb{E}\left\{\widetilde{y}_{h_3}\widetilde{y}_{h_7}\right\} - x_{h_2}x_{h_6}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_3}\right\} \\
&+ x_{h_2}x_{h_4}\mathbb{E}\left\{\widetilde{y}_{h_5}\widetilde{y}_{h_7}\right\} - x_{h_2}x_{h_4}x_{h_8}\mathbb{E}\left\{\widetilde{y}_{h_5}\right\} - x_{h_2}x_{h_4}x_{h_6}\mathbb{E}\left\{\widetilde{y}_{h_7}\right\} + x_{h_2}x_{h_4}x_{h_6}x_{h_8}\,. \qquad \text{(C.8)}
\end{aligned}$$

Since $h_i, i \in \{1,3,5,7\}$ represent solely generic placeholders for any actual index, it suffices to compute only a few expectations from the previous relation, which can then be used for any other combination. We have:

$$\begin{aligned}
&\mathbb{E}_{\theta'(\widetilde{\boldsymbol{y}})}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\widetilde{y}_{h_5}\widetilde{y}_{h_7}\right\} = \mu_{h_1}\mu_{h_3}\mu_{h_5}\mu_{h_7} + \Sigma_{h_1,h_3}\Sigma_{h_5,h_7} + \Sigma_{h_1,h_5}\Sigma_{h_3,h_7} + \Sigma_{h_1,h_7}\Sigma_{h_3,h_5} \\
&+\mu_{h_1}\mu_{h_3}\Sigma_{h_5,h_7} + \mu_{h_1}\mu_{h_5}\Sigma_{h_3,h_7} + \mu_{h_1}\mu_{h_7}\Sigma_{h_3,h_5} + \mu_{h_3}\mu_{h_5}\Sigma_{h_1,h_7} + \mu_{h_3}\mu_{h_7}\Sigma_{h_1,h_5} + \mu_{h_5}\mu_{h_7}\Sigma_{h_1,h_3}\,, \qquad \text{(C.9)}
\end{aligned}$$

$$\mathbb{E}_{\theta'(\boldsymbol{y})}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\widetilde{y}_{h_5}\right\} = \mu_{h_1}\mu_{h_3}\mu_{h_5} + \mu_{h_5}\Sigma_{h_1,h_3} + \mu_{h_3}\Sigma_{h_1,h_5} + \mu_{h_1}\Sigma_{h_3,h_5}\,, \qquad \text{(C.10)}$$

$$\mathbb{E}_{\theta'(\boldsymbol{y})}\left\{\widetilde{y}_{h_1}\widetilde{y}_{h_3}\right\} = \mu_{h_1}\mu_{h_3} + \Sigma_{h_1,h_3}\,, \qquad \text{(C.11)}$$

$$\mathbb{E}_{\theta'(\boldsymbol{y})}\left\{\widetilde{y}_{h_1}\right\} = \mu_{h_1}\,. \qquad \text{(C.12)}$$

Before proceeding, for ease of notation, we denote $d_n = \operatorname{card}\{\mathcal{D}_n\}$. Indexes $p, q, r, s = 1, \ldots, d_n$ correspond to entries of the dictionary $\mathcal{D}_n$. Also, the bullet $\bullet$ is used to mean that the respective index is irrelevant, due to its corresponding term not being present in the generic form (C.1).

C.1 Cases corresponding to $\boldsymbol{R}_{ss}$ ($c_1 = c_2 = 0, c_3 = 1$)

We recall that $\boldsymbol{R}_{ss} \triangleq \mathbb{E}\{\boldsymbol{s}(i)\boldsymbol{s}^\top(i)\} = \begin{bmatrix} \boldsymbol{R}_{zz} & \boldsymbol{R}_{kz}^\top \\ \boldsymbol{R}_{kz} & \boldsymbol{R}_{kk} \end{bmatrix}$, where $\boldsymbol{R}_{zz} \triangleq \mathbb{E}\left\{\boldsymbol{z}(i)\boldsymbol{z}^\top(i)\right\}$, $\boldsymbol{R}_{kk} \triangleq \mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{k}^\top(i)\right\}$, and $\boldsymbol{R}_{kz} \triangleq \mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{z}^\top(i)\right\}$.

We also recall (5.23), (5.24), as well as their explicit forms for the particular case of the Gaussian kernel:

$$\boldsymbol{z}(i) = \left[\boldsymbol{z}_1^\top(i), \ldots, \boldsymbol{z}_N^\top(i)\right]^\top, \quad [\boldsymbol{z}_m(i)]_q = \left.\frac{\partial \kappa(\boldsymbol{y}(i), \boldsymbol{y}(\omega_q))}{\partial y_m(\omega_q)}\right|_{q=1,\ldots,d_n},$$

$$\boldsymbol{\ell}_m(i) = \left[\boldsymbol{\ell}_{1,m}^\top(i), \ldots, \boldsymbol{\ell}_{N,m}^\top(i)\right]^\top, \quad [\boldsymbol{\ell}_{m_1,m_2}(i)]_q = \left.\frac{\partial^2 \kappa(\boldsymbol{y}(i), \boldsymbol{y}(\omega_q))}{\partial y_{m_1}(\omega_q) \partial y_{m_2}(\omega_q)}\right|_{q=1,\ldots,d_n},$$

$$[\boldsymbol{z}_m(i)]_q = \exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2\right)\left(\frac{[\boldsymbol{y}(i)]_m - [\boldsymbol{y}(\omega_q)]_m}{\sigma^2}\right),$$

$$[\boldsymbol{\ell}_{m_1,m_2}]_q = \begin{cases} -\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2\right)\frac{[\boldsymbol{y}(i)]_{m_1} - [\boldsymbol{y}(\omega_q)]_{m_1}}{\sigma^2}\frac{[\boldsymbol{y}(i)]_{m_2} - [\boldsymbol{y}(\omega_q)]_{m_2}}{\sigma^2} & m_1 \neq m_2 \\ -\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2\right)\left(\frac{\left([\boldsymbol{y}(i)]_{m_1} - [\boldsymbol{y}(\omega_q)]_{m_1}\right)^2}{\sigma^4} - \frac{1}{\sigma^2}\right) & m_1 = m_2 \end{cases}.$$

We are now able compute every block of $\boldsymbol{R}_{ss}$ separately.

C.1.1 Block $\boldsymbol{R}_{zz}$

Due to the nature of vector $\boldsymbol{z}$, matrix $\boldsymbol{R}_{zz}$ is, in turn, a block matrix, whose blocks are $[\boldsymbol{R}_{zz,m_1,m_2}]$, $m_1, m_2 = 1, \ldots, N$. Their $(p,q)^{\text{th}}$ entry, $p, q = 1, \ldots, d_n$, is given by:

$$\begin{aligned}[\boldsymbol{R}_{zz,m_1,m_2}]_{p,q} &= \left[\mathbb{E}\left\{\boldsymbol{z}_{m_1}(i)\boldsymbol{z}_{m_2}^\top(i)\right\}\right]_{p,q} = \mathbb{E}\left\{[\boldsymbol{z}_{m_1}]_p[\boldsymbol{z}_{m_2}]_q\right\} \\ &= \mathbb{E}\left\{\frac{(y_{m_1}(i) - y_{m_1}(\omega_p))}{\sigma^2}\frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2} \times \exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 - \frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2\right)\right\}. \end{aligned} \tag{C.13}$$

We relate these entries to $\nu(\{h_i\}_{i=1}^8)$ through:

$$[\boldsymbol{R}_{zz,m_1,m_2}]_{p,q} = \frac{1}{\sigma^4}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} \left.\begin{array}{l} h_1 = \bullet \\ h_2 = \bullet \end{array}\right\} \iota_1 = 0 \\ h_3 = m_1 \\ h_4 = m_1 \\ \left.\begin{array}{l} h_5 = \bullet \\ h_6 = \bullet \end{array}\right\} \iota_3 = 0 \\ h_7 = m_2 \\ h_8 = N + m_2 \end{cases} . \tag{C.14}$$

C.1.2 Block $\boldsymbol{R}_{kz}$

Once again, due to the nature of vector $\boldsymbol{z}$, matrix $\boldsymbol{R}_{kz}$ is, in turn, a block matrix, whose blocks are $[\boldsymbol{R}_{kz,m}]$. Their $(p,q)^{\text{th}}$ entry, $p, q = 1, \ldots, d_n$, is given by:

$$[\boldsymbol{R}_{kz,m}]_{p,q} = \left[\mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{z}_m^\top(i)\right\}\right]_{p,q} = \mathbb{E}\left\{[\boldsymbol{k}]_p[\boldsymbol{z}_m]_q\right\} = \mathbb{E}\left\{\frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2}\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 - \frac{1}{2\sigma^2}\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2\right)\right\} . \tag{C.15}$$

We relate these entries to $\nu(\{h_i\}_{i=1}^8)$ through:

$$[\boldsymbol{R}_{kz,m}]_{p,q} = \frac{1}{\sigma^2}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} \left.\begin{array}{l} h_1 = \bullet \\ h_2 = \bullet \end{array}\right\} \iota_1 = 0 \\ \left.\begin{array}{l} h_3 = \bullet \\ h_4 = \bullet \end{array}\right\} \iota_2 = 0 \\ \left.\begin{array}{l} h_5 = \bullet \\ h_6 = \bullet \end{array}\right\} \iota_3 = 0 \\ h_7 = m \\ h_8 = N + m \end{cases} . \tag{C.16}$$

C.1.3 Block $\boldsymbol{R}_{kk}$

Its $(p,q)^{\text{th}}$ entry, $p,q=1,\ldots,d_n$, is computed trivially and is given by:

$$[\boldsymbol{R}_{kk}]_{p,q}=\left[\mathbb{E}\left\{\boldsymbol{k}(i)\boldsymbol{k}^{\top}(i)\right\}\right]_{p,q}=\mathbb{E}\left\{[\boldsymbol{k}]_p[\boldsymbol{k}]_q\right\}=\mathbb{E}\left\{\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\right\}. \tag{C.17}$$

We relate these entries to $\nu(\{h_i\}_{i=1}^8)$ through:

$$[\boldsymbol{R}_{kk}]_{p,q}=\nu(\{h_i\}_{i=1}^8),\text{ with }\begin{cases}\left.\begin{matrix}h_1=\bullet\\h_2=\bullet\end{matrix}\right\}\iota_1=0\\\left.\begin{matrix}h_3=\bullet\\h_4=\bullet\end{matrix}\right\}\iota_2=0\\\left.\begin{matrix}h_5=\bullet\\h_6=\bullet\end{matrix}\right\}\iota_3=0\\\left.\begin{matrix}h_7=\bullet\\h_8=\bullet\end{matrix}\right\}\iota_4=0\end{cases}. \tag{C.18}$$

C.2 Cases corresponding to $\boldsymbol{K}^{(u,v)}$ ($c_1=c_2=c_3=1$)

Recall that $\boldsymbol{s}(i)=\begin{bmatrix}\boldsymbol{z}(i)\\\boldsymbol{k}(i)\end{bmatrix}$, and:

$$\left[\boldsymbol{K}^{(u,v)}\right]_{ab}=\mathbb{E}\left\{[\boldsymbol{s}(i)]_u[\boldsymbol{s}(i)]_v[\boldsymbol{s}(i)]_a[\boldsymbol{s}(i)]_b\right\}. \tag{C.19}$$

We can divide this relation in the following cases:

C.2.1 Term $\mathbb{E}\{z_u(i)z_a(i)z_b(i)z_v(i)\}$

Consider indexes $u, v = 1, \ldots, Nd_n$, $a, b = 1, \ldots, Nd_n$.

$$\begin{aligned}
&\mathbb{E}\{z_u(i)z_a(i)z_b(i)z_v(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_1}(i) - y_{m_1}(\omega_p))}{\sigma^2} \frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2} \frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2} \frac{(y_{m_4}(i) - y_{m_4}(\omega_s))}{\sigma^2} \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_s)\|^2 \right) \right) \Bigg\} \qquad \text{(C.20)} \\
&= \frac{1}{\sigma^8} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \left\lceil \frac{u}{d_n} \right\rceil \\ h_3 = h_4 - N = m_2 = \left\lceil \frac{a}{d_n} \right\rceil \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = \left\lceil \frac{v}{d_n} \right\rceil \end{cases}, \begin{cases} \omega_p = \mathrm{mod}(u-1, d_n) + 1 \\ \omega_q = \mathrm{mod}(a-1, d_n) + 1 \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \mathrm{mod}(v-1, d_n) + 1 \end{cases} . \qquad \text{(C.21)}
\end{aligned}$$

C.2.2 Term $\mathbb{E}\{k_u(i)z_a(i)z_b(i)z_v(i)\}$

Consider indexes $u = 1, \ldots, d_n, v = 1, \ldots, Nd_n$, $a, b = 1, \ldots, Nd_n$.

$$\begin{aligned}
&\mathbb{E}\{k_u(i)z_a(i)z_b(i)z_v(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2} \frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2} \frac{(y_{m_4}(i) - y_{m_4}(\omega_s))}{\sigma^2} \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_s)\|^2 \right) \right) \Bigg\} \qquad \text{(C.22)} \\
&= \frac{1}{\sigma^6} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \bullet \\ h_3 = h_4 - N = m_2 = \left\lceil \frac{a}{d_n} \right\rceil \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = \left\lceil \frac{v}{d_n} \right\rceil \end{cases}, \begin{cases} \omega_p = u \\ \omega_q = \mathrm{mod}(a-1, d_n) + 1 \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \mathrm{mod}(v-1, d_n) + 1 \end{cases} . \qquad \text{(C.23)}
\end{aligned}$$

C.2.3 Term $\mathbb{E}\{k_u(i)k_a(i)z_b(i)z_v(i)\}$

Consider indexes $u = 1, \ldots, d_n, v = 1, \ldots, Nd_n$, $a = 1, \ldots, d_n$, $b = 1, \ldots, Nd_n$.

$$\begin{aligned}
&\mathbb{E}\{k_u(i)k_a(i)z_b(i)z_v(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2} \frac{(y_{m_4}(i) - y_{m_4}(\omega_s))}{\sigma^2} \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_s)\|^2 \right) \right) \Bigg\} \qquad \text{(C.24)} \\
&= \frac{1}{\sigma^4} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \bullet \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = \left\lceil \frac{v}{d_n} \right\rceil \end{cases}, \begin{cases} \omega_p = u \\ \omega_q = a \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \mathrm{mod}(v-1, d_n) + 1 \end{cases} . \qquad \text{(C.25)}
\end{aligned}$$

C.2.4 Term $\mathbb{E}\{k_u(i)k_a(i)k_b(i)z_v(i)\}$

Consider indexes $u = 1, \ldots, d_n, v = 1, \ldots, Nd_n$, $a, b = 1, \ldots, d_n$.

$$\begin{aligned}
&\mathbb{E}\{k_u(i)k_a(i)k_b(i)z_v(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_4}(i) - y_{m_4}(\omega_s))}{\sigma^2} \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_s)\|^2 \right) \right) \Bigg\} \qquad \text{(C.26)} \\
&= \frac{1}{\sigma^2} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \bullet \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \bullet \\ h_7 = h_8 - 3N = m_4 = \left\lceil \frac{v}{d_n} \right\rceil \end{cases}, \begin{cases} \omega_p = u \\ \omega_q = a \\ \omega_r = b \\ \omega_s = \mathrm{mod}(v-1, d_n) + 1 \end{cases} . \qquad \text{(C.27)}
\end{aligned}$$

C.2.5 Term $\mathbb{E}\left\{k_u(i)k_a(i)k_b(i)k_v(i)\right\}$

Consider indexes $u, v = 1, \ldots, d_n,\ a, b = 1, \ldots, d_n$.

$$\begin{aligned}
&\mathbb{E}\left\{k_u(i)k_a(i)k_b(i)k_v(i)\right\} \\
&\quad= \exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2+\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2+\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_r)\|^2+\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_s)\|^2\right)\right) \qquad \text{(C.28)} \\
&\quad=\nu(\{h_i\}_{i=1}^8),\ \text{with} \begin{cases} h_1 = h_2 = m_1 = \bullet \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \bullet \\ h_7 = h_8 - 3N = m_4 = \bullet \end{cases}, \begin{cases} \omega_p = u \\ \omega_q = a \\ \omega_r = b \\ \omega_s = v \end{cases}. \qquad \text{(C.29)}
\end{aligned}$$

C.3 Cases corresponding to $\mathbb{E}\{s_u(i)s_a(i)s_b(i)y_n(i)\}$ $(c_1 = 0, c_2 = c_3 = 1)$

These terms can easily be computed using the results in Annex C.2.

C.3.1 Term $\mathbb{E}\{z_u(i)z_a(i)z_b(i)y_n(i)\}$

Consider indexes $u = 1, \ldots, Nd_n$, $a, b = 1, \ldots, Nd_n$.

We adapt relations (C.20) – (C.21), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{z_u(i)z_a(i)z_b(i)y_n(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_1}(i) - y_{m_1}(\omega_p))}{\sigma^2} \frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2} \frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2} y_n(i) \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 \right) \right) \Bigg\} \qquad \text{(C.30)} \\
&= \frac{1}{\sigma^6} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \left\lceil \frac{u}{d_n} \right\rceil \\ h_3 = h_4 - N = m_2 = \left\lceil \frac{a}{d_n} \right\rceil \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = \mathrm{mod}(u-1, d_n) + 1 \\ \omega_q = \mathrm{mod}(a-1, d_n) + 1 \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \bullet \end{cases}. \qquad \text{(C.31)}
\end{aligned}$$

C.3.2 Term $\mathbb{E}\{z_u(i)k_a(i)z_b(i)y_n(i)\}$

Consider indexes $u = 1, \ldots, Nd_n$, $a = 1, \ldots, d_n$, $b = 1, \ldots, Nd_n$.

We adapt relations (C.22) – (C.23), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{z_u(i)k_a(i)z_b(i)y_n(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_1}(i) - y_{m_1}(\omega_p))}{\sigma^2} \frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2} y_n(i) \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 \right) \right) \Bigg\} \qquad \text{(C.32)} \\
&= \frac{1}{\sigma^4} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \left\lceil \frac{u}{d_n} \right\rceil \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = \mathrm{mod}(u-1, d_n) + 1 \\ \omega_q = a \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \bullet \end{cases}. \qquad \text{(C.33)}
\end{aligned}$$

C.3.3 Term $\mathbb{E}\{z_u(i)k_a(i)k_b(i)y_n(i)\}$

Consider indexes $u = 1, \ldots, Nd_n$, $a, b = 1, \ldots, d_n$.

We adapt relations (C.24) – (C.25), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{z_u(i)k_a(i)k_b(i)y_n(i)\} \\
&= \mathbb{E}\Bigg\{ \frac{(y_{m_1}(i) - y_{m_1}(\omega_p))}{\sigma^2} y_n(i) \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 \right) \right) \Bigg\} \qquad \text{(C.34)} \\
&= \frac{1}{\sigma^2} \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \left\lceil \frac{u}{d_n} \right\rceil \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \bullet \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = \mathrm{mod}(u-1, d_n) + 1 \\ \omega_q = a \\ \omega_r = b \\ \omega_s = \bullet \end{cases} . \qquad \text{(C.35)}
\end{aligned}$$

C.3.4 Term $\mathbb{E}\{k_u(i)k_a(i)k_b(i)y_n(i)\}$

Consider indexes $u = 1, \ldots, d_n$, $a, b = 1, \ldots, d_n$.

We adapt relations (C.26) – (C.27), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{k_u(i)k_a(i)k_b(i)k_v(i)\} \\
&\quad = \exp\left(-\frac{1}{2\sigma^2} \left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_p)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2 \right) \right) \qquad \text{(C.36)} \\
&= \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = \bullet \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \bullet \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = u \\ \omega_q = a \\ \omega_r = b \\ \omega_s = \bullet \end{cases} . \qquad \text{(C.37)}
\end{aligned}$$

C.4 Cases corresponding to $\mathbb{E}\{s_a(i)s_b(i)y_n^2(i)\}$ $(c_1 = c_2 = 0, c_3 = 1)$

These terms can easily be computed using the results in Annex C.3.

C.4.1 Term $\mathbb{E}\{y_n(i)z_a(i)z_b(i)y_n(i)\}$

Consider indexes $a, b = 1, \ldots, Nd_n$.

We adapt relations (C.30) – (C.31), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{y_n(i)z_a(i)z_b(i)y_n(i)\} \\
&= \mathbb{E}\Bigg\{ y_n(i)\frac{(y_{m_2}(i) - y_{m_2}(\omega_q))}{\sigma^2}\frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2}y_n(i) \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2\right)\right)\Bigg\}
\end{aligned} \tag{C.38}$$

$$= \frac{1}{\sigma^4}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = n \\ h_3 = h_4 - N = m_2 = \left\lceil \frac{a}{d_n} \right\rceil \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = \bullet \\ \omega_q = \mathrm{mod}(a-1, d_n) + 1 \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \bullet \end{cases}. \tag{C.39}$$

C.4.2 Term $\mathbb{E}\{y_n(i)k_a(i)z_b(i)y_n(i)\}$

Consider indexes $a = 1, \ldots, d_n$, $b = 1, \ldots, Nd_n$.

We adapt relations (C.32) – (C.33), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{y_n(i)k_a(i)z_b(i)y_n(i)\} \\
&= \mathbb{E}\Bigg\{ y_n(i)\frac{(y_{m_3}(i) - y_{m_3}(\omega_r))}{\sigma^2}y_n(i) \\
&\quad \times \exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_q)\|^2 + \|\boldsymbol{y}(i) - \boldsymbol{y}(\omega_r)\|^2\right)\right)\Bigg\}
\end{aligned} \tag{C.40}$$

$$= \frac{1}{\sigma^2}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = h_2 = m_1 = n \\ h_3 = h_4 - N = m_2 = \bullet \\ h_5 = h_6 - 2N = m_3 = \left\lceil \frac{b}{d_n} \right\rceil \\ h_7 = h_8 - 3N = m_4 = n \end{cases}, \begin{cases} \omega_p = \bullet \\ \omega_q = a \\ \omega_r = \mathrm{mod}(b-1, d_n) + 1 \\ \omega_s = \bullet \end{cases}. \tag{C.41}$$

C.4.3 Term $\mathbb{E}\{y_n(i)k_a(i)k_b(i)y_n(i)\}$

Consider indexes $a, b = 1, \ldots, d_n$.

We adapt relations (C.34) – (C.35), thus obtaining:

$$\begin{aligned}
&\mathbb{E}\{y_n(i)k_a(i)k_b(i)y_n(i)\} \\
&\quad = \mathbb{E}\left\{ y_n(i)y_n(i)\exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2+\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_r)\|^2\right)\right)\right\} \qquad \text{(C.42)} \\
&\quad = \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1=h_2=m_1=n \\ h_3=h_4-N=m_2=\bullet \\ h_5=h_6-2N=m_3=\bullet \\ h_7=h_8-3N=m_4=n \end{cases}, \begin{cases} \omega_p=\bullet \\ \omega_q=a \\ \omega_r=b \\ \omega_s=\bullet \end{cases}. \qquad \text{(C.43)}
\end{aligned}$$

C.5 Cases corresponding to $\mathbb{E}\{s_b(i)y_n(i)\}$ $(c_1=c_2=c_3=0)$

These terms can easily be computed using the results in Annex C.4.

C.5.1 Term $\mathbb{E}\{z_b(i)y_n(i)\}$

Consider index $b=1,\ldots,Nd_n$.

We adapt relations (C.38) – (C.39), thus obtaining:

$$\mathbb{E}\{z_b(i)y_n(i)\} = \mathbb{E}\left\{\frac{(y_{m_3}(i)-y_{m_3}(\omega_r))}{\sigma^2}y_n(i)\exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_r)\|^2\right)\right)\right\} \qquad \text{(C.44)}$$

$$= \frac{1}{\sigma^2}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1=h_2=m_1=\bullet \\ h_3=h_4-N=m_2=\bullet \\ h_5=h_6-2N=m_3=\left\lceil \frac{b}{|\mathcal{D}|}\right\rceil \\ h_7=h_8-3N=m_4=n \end{cases}, \begin{cases} \omega_p=\bullet \\ \omega_q=\bullet \\ \omega_r=\mathrm{mod}(b-1,|\mathcal{D}|)+1 \\ \omega_s=\bullet \end{cases}. \qquad \text{(C.45)}$$

C.5.2 Term $\mathbb{E}\{k_b(i)y_n(i)\}$

Consider index $b=1,\ldots,d_n$.

We adapt relations (C.42) – (C.43), thus obtaining:

$$\mathbb{E}\{k_b(i)y_n(i)\} = \mathbb{E}\left\{ y_n(i)\exp\left(-\frac{1}{2\sigma^2}\left(\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_r)\|^2\right)\right)\right\} \qquad \text{(C.46)}$$

$$= \nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1=h_2=m_1=\bullet \\ h_3=h_4-N=m_2=\bullet \\ h_5=h_6-2N=m_3=\bullet \\ h_7=h_8-3N=m_4=n \end{cases}, \begin{cases} \omega_p=\bullet \\ \omega_q=\bullet \\ \omega_r=b \\ \omega_s=\bullet \end{cases}. \qquad \text{(C.47)}$$

C.6 Cases corresponding to $\mathbb{E}\{\boldsymbol{T}_m(i)\}$ ($c_1 = c_2 = 0, c_3 = 1$)

Since $\boldsymbol{t}_m(p) = \begin{bmatrix} \boldsymbol{\ell}_m(p) \\ \boldsymbol{z}_m(p) \end{bmatrix}$, the matrix $\mathbb{E}\{\boldsymbol{T}_m(i)\}$ can be written as:

$$\mathbb{E}\{\boldsymbol{T}_m(i)\} = \mathbb{E}\left\{\boldsymbol{t}_m(i)\boldsymbol{t}_m^\top(i)\right\} = \mathbb{E}\left\{\begin{bmatrix} \boldsymbol{\ell}_m(i) \\ \boldsymbol{z}_m(i) \end{bmatrix}\begin{bmatrix} \boldsymbol{\ell}_m^\top(i) & \boldsymbol{z}_m^\top(i) \end{bmatrix}\right\} = \begin{bmatrix} \boldsymbol{R}_{\ell\ell,m} & \boldsymbol{R}_{z\ell,m}^\top \\ \boldsymbol{R}_{z\ell,m} & \boldsymbol{R}_{zz,m} \end{bmatrix}, \quad \text{(C.48)}$$

where in the last step we used the fact that the signals are i.i.d. (for different i).

In the following subsections, indexes $a, b = 1\ldots, N$ correspond to nodes in the graph and, implicitly, blocks in ℓ_m, when applicable.

C.6.1 Block $\boldsymbol{R}_{\ell\ell,m}$

$$\begin{aligned}
\left[\boldsymbol{R}_{\ell\ell,m}\right]_{(a-1)d_n+p,(b-1)d_n+q} &= \left[\mathbb{E}\left\{\boldsymbol{\ell}_m(i)\boldsymbol{\ell}_m^\top(i)\right\}\right]_{(a-1)d_n+p,(b-1)d_n+q} = \mathbb{E}\left\{[\boldsymbol{\ell}_{a,m}]_p[\boldsymbol{\ell}_{b,m}]_q\right\} \\
&= \begin{cases}
\mathbb{E}\Big\{ \dfrac{(y_a(i)-y_a(\omega_p))}{\sigma^2}\dfrac{(y_m(i)-y_m(\omega_p))}{\sigma^2}\dfrac{(y_b(i)-y_b(\omega_q))}{\sigma^2}\dfrac{(y_m(i)-y_m(\omega_q))}{\sigma^2} \\
\qquad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & a \neq m,\ b \neq m \\
\mathbb{E}\Big\{ \dfrac{(y_a(i)-y_a(\omega_p))}{\sigma^2}\dfrac{(y_m(i)-y_m(\omega_p))}{\sigma^2}\left(\dfrac{\left(y_b(i)-y_b(\omega_q)\right)^2}{\sigma^4} - \dfrac{1}{\sigma^2}\right) \\
\qquad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & a \neq m,\ b = m \\
\mathbb{E}\Big\{ \dfrac{(y_b(i)-y_b(\omega_q))}{\sigma^2}\dfrac{(y_m(i)-y_m(\omega_q))}{\sigma^2}\left(\dfrac{\left(y_a(i)-y_a(\omega_p)\right)^2}{\sigma^4} - \dfrac{1}{\sigma^2}\right) \\
\qquad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & a = m,\ b \neq m \\
\mathbb{E}\Big\{ \left(\dfrac{\left(y_a(i)-y_a(\omega_p)\right)^2}{\sigma^4} - \dfrac{1}{\sigma^2}\right)\left(\dfrac{\left(y_b(i)-y_b(\omega_q)\right)^2}{\sigma^4} - \dfrac{1}{\sigma^2}\right) \\
\quad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & a = m,\ b = m
\end{cases}
\end{aligned} \quad \text{(C.49)}$$

$$
\left[\boldsymbol{R}_{\ell\ell,m}\right]_{(a-1)d_n+p,(b-1)d_n+q}
$$

$$
= \begin{cases}
\dfrac{1}{\sigma^8}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} h_1 = a \\ h_2 = a \\ h_3 = m \\ h_4 = m \\ h_5 = b \\ h_6 = N+b \\ h_7 = m \\ h_8 = N+m \end{cases} & , \quad a \neq m,\, b \neq m \\
\dfrac{1}{\sigma^8}\nu(\{h_i\}_{i=1}^8) \\ -\dfrac{1}{\sigma^2}\left[\boldsymbol{R}_{zz}\right]_{(a-1)d_n+p,(m-1)d_n+p}, \text{ with } \begin{cases} h_1 = a \\ h_2 = a \\ h_3 = b \\ h_4 = b \\ h_5 = b \\ h_6 = N+b \\ h_7 = b \\ h_8 = N+b \end{cases} & , \quad a \neq m,\, b = m \\
\dfrac{1}{\sigma^8}\nu(\{h_i\}_{i=1}^8) \\ -\dfrac{1}{\sigma^2}\left[\boldsymbol{R}_{zz}\right]_{(b-1)d_n+q,(m-1)d_n+q}, \text{ with } \begin{cases} h_1 = b \\ h_2 = b \\ h_3 = a \\ h_4 = a \\ h_5 = a \\ h_6 = N+a \\ h_7 = a \\ h_8 = N+a \end{cases} & , \quad a = m,\, b \neq m \\
\dfrac{1}{\sigma^8}\nu(\{h_i\}_{i=1}^8) \\ -\dfrac{1}{\sigma^2}\left[\boldsymbol{R}_{zz}\right]_{(a-1)d_n+p,(a-1)d_n+p} \\ -\dfrac{1}{\sigma^2}\left[\boldsymbol{R}_{zz}\right]_{(b-1)d_n+q,(b-1)d_n+q} \\ +\dfrac{1}{\sigma^4}\left[\boldsymbol{R}_{kk}\right]_{p,q}, \text{ with } \begin{cases} h_1 = a \\ h_2 = a \\ h_3 = a \\ h_4 = a \\ h_5 = b \\ h_6 = N+b \\ h_7 = b \\ h_8 = N+b \end{cases} & , \quad a = m,\, b = m
\end{cases} . \tag{C.50}
$$

C.6.2 Block $\boldsymbol{R}_{z\ell,m}$

$$
\left[\boldsymbol{R}_{z\ell,m}\right]_{p,(b-1)d_n+q} = \left[\mathbb{E}\left\{\boldsymbol{z}_m(i)\boldsymbol{\ell}_m^\top(i)\right\}\right]_{p,(b-1)d_n+q} = \mathbb{E}\left\{[\boldsymbol{z}_m]_p[\boldsymbol{\ell}_{b,m}]_q\right\}
$$
$$
= \begin{cases} -\mathbb{E}\Big\{ \dfrac{(y_m(i)-y_m(\omega_p))}{\sigma^2}\dfrac{(y_b(i)-y_b(\omega_q))}{\sigma^2}\dfrac{(y_m(i)-y_m(\omega_q))}{\sigma^2} \\ \quad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & b \neq m \\ -\mathbb{E}\Big\{ \dfrac{(y_b(i)-y_b(\omega_p))}{\sigma^2}\left(\dfrac{\left(y_b(i)-y_b(\omega_q)\right)^2}{\sigma^4} - \dfrac{1}{\sigma^2}\right) \\ \quad \times \exp\left(-\dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \dfrac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\Big\}, & b = m \end{cases} . \tag{C.51}
$$

$$
[\boldsymbol{R}_{z\ell,m}]_{p,(b-1)d_n+q} = \begin{cases} -\dfrac{1}{\sigma^6}\nu(\{h_i\}_{i=1}^8), \text{ with } \begin{cases} \left.\begin{matrix} h_1 = \bullet \\ h_2 = \bullet \end{matrix}\right\} \iota_1 = 0 \\ h_3 = m \\ h_4 = m \\ h_5 = b \\ h_6 = N+b \\ h_7 = m \\ h_8 = N+m \end{cases}, & b \neq m \\ \begin{matrix} -\dfrac{1}{\sigma^6}\nu(\{h_i\}_{i=1}^8) \\ +\dfrac{1}{\sigma^2}\left[\boldsymbol{R}_{kz}\right]_{q,(b-1)d_n+p} \end{matrix}, \text{ with } \begin{cases} \left.\begin{matrix} h_1 = \bullet \\ h_2 = \bullet \end{matrix}\right\} \iota_1 = 0 \\ h_3 = b \\ h_4 = b \\ h_5 = b \\ h_6 = N+b \\ h_7 = b \\ h_8 = N+b \end{cases}, & b = m \end{cases} . \tag{C.52}
$$

C.6.3 Block $\boldsymbol{R}_{zz,m}$

$$
[\boldsymbol{R}_{zz,m}]_{p,q} = \left[\mathbb{E}\left\{\boldsymbol{z}_m(i)\boldsymbol{z}_m^\top(i)\right\}\right]_{p,q} = \mathbb{E}\left\{[\boldsymbol{z}_m]_p[\boldsymbol{z}_m]_q\right\}
$$
$$
= \mathbb{E}\left\{\frac{(y_m(i)-y_m(\omega_p))}{\sigma^2}\frac{(y_m(i)-y_m(\omega_q))}{\sigma^2}\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_p)\|^2 - \frac{1}{2\sigma^2}\|\boldsymbol{y}(i)-\boldsymbol{y}(\omega_q)\|^2\right)\right\} . \tag{C.53}
$$

$$[\boldsymbol{R}_{zz,m}]_{p,q} = \frac{1}{\sigma^4}\nu(\{h_i\}_{i=1}^{8}), \text{ with } \begin{cases} \left.\begin{array}{l} h_1 = \bullet \\ h_2 = \bullet \end{array}\right\} \iota_1 = 0 \\ h_3 = m \\ h_4 = m \\ \left.\begin{array}{l} h_5 = \bullet \\ h_6 = \bullet \end{array}\right\} \iota_3 = 0 \\ h_7 = m \\ h_8 = N + m \end{cases} . \tag{C.54}$$

Bibliography

R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74: 47–97, Jan 2002. doi: 10.1103/RevModPhys.74.47. URL https://link.aps.org/doi/10.1103/RevModPhys.74.47. (Cited on page 95.)

A. V. Alekseyenko, N. I. Lytkin, J. Ai, B. Ding, L. Padyukov, C. F. Aliferis, and A. Statnikov. Causal graph-based analysis of genome-wide association data in rheumatoid arthritis. *Biology direct*, 6(1):25, 2011. (Cited on page 2.)

B. Baingana, G. Mateos, and G. B. Giannakis. Dynamic structural equation models for tracking topologies of social networks. In *Proc. IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 292–295, St. Martin, France, 2013. doi: 10.1109/CAMSAP.2013.6714065. (Cited on page 14.)

Y. Bao and A. Ullah. Expectation of quadratic forms in normal and nonnormal variables with applications. *Journal of Statistical Planning and Inference*, 140(5):1193–1205, 2010. (Cited on page 79.)

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003. (Cited on page 2.)

Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in neural information processing systems*, pages 177–184, 2004. (Cited on page 31.)

N. Biggs. *Algebraic Graph Theory*. Cambridge University Press, 1993. (Cited on page 8.)

A. Bolstad, B. D. van Veen, and R. Nowak. Causal network inference via group sparse regularization. *IEEE Transactions on Signal Processing*, 59(6):2628–2641, 2011. (Cited on pages 14 and 21.)

P. Bouboulis, S. Pougkakiotis, and S. Theodoridis. Efficient KLMS and KRLS algorithms: A random Fourier feature perspective. In *2016 IEEE Statistical Signal Processing Workshop (SSP)*, pages 1–5, 2016. (Cited on page 42.)

S. Boucheron, G. Lugosi, and O. Bousquet. Concentration inequalities. In *Summer School on Machine Learning*, pages 208–240. Springer, 2003. (Cited on page 70.)

M. Breakspear. Dynamic models of large-scale brain activity. *Nature neuroscience*, 20(3):340–352, 2017. (Cited on page 38.)

S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998. (Cited on page 96.)

A. A. Bueno and M. T. M. Silva. Gram-Schmidt-based sparsification for kernel dictionary. *IEEE Signal Processing Letters*, 27:1130–1134, 2020. (Cited on page 42.)

A. Buja, T. Hastie, and R. Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, pages 453–510, 1989. (Cited on page 64.)

E. J. Candes, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008. (Cited on page 34.)

J. Chen, W. Gao, C. Richard, and J.-C. M. Bermudez. Convergence analysis of kernel LMS algorithm with pre-tuned dictionary. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014. (Cited on page 68.)

J. Chen, C. Richard, Y. Song, and D. Brie. Transient performance analysis of zero-attracting LMS. *Signal Processing Letters, IEEE*, 23(12):1786–1790, 2016. (Cited on page 26.)

Y. Chen, Y. Gu, and A. O. Hero. Sparse LMS for system identification. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3125–3128. IEEE, 2009. (Cited on pages 25 and 34.)

T. Cioacă, B. Dumitrescu, and M.-S. Stupariu. Graph-based wavelet multiresolution modeling of multivariate terrain data. In L. Stanković and E. Sejdić, editors, *Vertex-Frequency Analysis of Graph Signals*, pages 479–507. Springer International Publishing, Cham, 2019. ISBN 978-3-030-03574-7. doi: 10.1007/978-3-030-03574-7_15. URL https://doi.org/10.1007/978-3-030-03574-7_15. (Cited on page 2.)

M. Danisch, O. Balalau, and M. Sozio. Listing k-cliques in sparse real-world graphs. In *Proceedings of the 2018 World Wide Web Conference*, pages 589–598, 2018. (Cited on pages 25 and 65.)

J. A. de Zwart, P. van Gelderen, J. M. Jansma, M. Fukunaga, M. Bianciardi, and J. H. Duyn. Hemodynamic nonlinearities affect BOLD fMRI response timing and amplitude. *Neuroimage*, 47(4):1649–1658, 2009. (Cited on pages 38 and 63.)

B. S. Dees, L. Stanković, M. Daković, A. G. Constantinides, and D. P. Mandic. Unitary shift operators on a graph. *arXiv preprint arXiv:1909.05767*, 2019. (Cited on page 9.)

A. P. Dempster. Covariance selection. *Biometrics*, 28(1):157–175, 1972. (Cited on page 14.)

R. D. Dennerll. Cognitive deficits and lateral brain dysfunction in temporal lobe epilepsy. *Epilepsia*, 5(2):177–191, 1964. (Cited on pages 37 and 51.)

J. A. Deri and J. M. F. Moura. New York City taxi analysis with graph signal processing. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1275–1279. IEEE, 2016. (Cited on page 2.)

E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1 (1):269–271, 1959. (Cited on page 96.)

F. Ding, D.and Xia, X. Yang, and C. Tang. Joint dictionary and graph learning for unsupervised feature selection. *Applied Intelligence*, pages 1–19, 2020. (Cited on page 14.)

X. Dong, D. Thanou, M. Rabbat, and P. Frossard. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 36(3):44–63, 2019. ISSN 1558-0792. doi: 10.1109/MSP.2018.2887284. (Cited on page 14.)

R. C. Elandt-Johnson and N. L. Johnson. *Survival Models and Data Analysis*, volume 74. John Wiley & Sons, 1999. (Cited on page 74.)

Y. Engel, S. Mannor, and R. Meir. The kernel recursive least-squares algorithm. *IEEE Transactions on Signal Processing*, 52(8):2275–2285, 2004. (Cited on page 42.)

W. Holmes Finch. Modeling nonlinear structural equation models: A comparison of the two-stage generalized additive models and the finite mixture structural equation model. *Structural Equation Modeling: A Multidisciplinary Journal*, 22(1):60–75, 2015. doi: 10.1080/10705511.2014.935749. (Cited on page 14.)

J. M. Ford, V. A. Palzes, B. J. Roach, and D. H. Mathalon. Did I do that? Abnormal predictive processes in schizophrenia when button pressing to deliver a tone. *Schizophrenia bulletin*, 40 (4):804–812, 2013. (Cited on page 55.)

R. Freedman, L. E. Adler, M. Myles-Worsley, H. T. Nagamoto, C. Miller, M. Kisley, K. McRae, E. Cawthra, and M. Waldo. Inhibitory Gating of an Evoked Response to Repeated Auditory Stimuli in Schizophrenic and Normal Subjects: Human Recordings, Computer Simulation, and an Animal Model. *Archives of General Psychiatry*, 53(12):1114–1121, December 1996. ISSN 0003-990X. doi: 10.1001/archpsyc.1996.01830120052009. URL https://doi.org/10.1001/archpsyc.1996.01830120052009. (Cited on page 55.)

W. J. Freeman. EEG analysis gives model of neuronal template-matching mechanism for sensory search with olfactory bulb. *Biological cybernetics*, 35(4):221–234, 1979. (Cited on pages 38 and 63.)

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical Lasso. *Biostatistics*, 9(3):432–41, 2008. (Cited on page 14.)

G. Garrigos, L. Rosasco, and S. Villa. Sparse Multiple Kernel Learning: Support Identification via Mirror Stratifiability. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1077–1081. IEEE, 2018. (Cited on page 91.)

A. Gavili and X. Zhang. On the shift operator, graph frequency, and optimal filtering in graph signal processing. *IEEE Transactions on Signal Processing*, 65(23):6303–6318, 2017. (Cited on page 9.)

M. S. Gazzaniga. The split-brain: Rooting consciousness in biology. *Proceedings of the National Academy of Sciences*, 111(51):18093–18094, 2014. ISSN 0027-8424. doi: 10.1073/pnas.1417892111. URL https://www.pnas.org/content/111/51/18093. (Cited on page 37.)

G. George and S. M. Thampi. A graph-based security framework for securing industrial iot networks from vulnerability exploitations. *IEEE Access*, 6:43586–43601, 2018. (Cited on page 7.)

G. B. Giannakis, Y. Shen, and G. V. Karanikolas. Topology Identification and Learning over Graphs: Accounting for Nonlinearities and Dynamics. *Proceedings of the IEEE*, 106(5):787–807, 2018. ISSN 0018-9219. doi: 10.1109/JPROC.2018.2804318. (Cited on page 15.)

D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *Proceedings of the 31st international conference on Very large data bases*, pages 721–732, 2005. (Cited on page 25.)

L. J. Grady and J. R. Polimeni. *Discrete calculus: Applied analysis on graphs for computational science*. Springer Science & Business Media, 2010. (Cited on page 96.)

C. W. J. Granger. Some recent development in a concept of causality. *Journal of econometrics*, 39(1-2):199–211, 1988. (Cited on page 21.)

M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. http://stanford.edu/~boyd/graph_dcp.html. (Cited on page 82.)

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014. (Cited on page 82.)

B. Hanin. Universal function approximation by deep neural nets with bounded width and ReLU activations. *Mathematics*, 7(10):992, 2019. (Cited on page 92.)

C. T. Harbison, D. B. Gordon, T. I. Lee, N. J. Rinaldi, K. D. Macisaac, T. W. Danford, N. M. Hannett, J.-B. Tagne, D. B. Reynolds, J. Yoo, et al. Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431(7004):99–104, 2004. (Cited on page 25.)

I. E. K. Harrane, R. Flamary, and C. Richard. Toward privacy-preserving diffusion strategies for adaptation and learning over networks. In *2016 24th European Signal Processing Conference (EUSIPCO)*, pages 1513–1517, 2016. (Cited on page 90.)

J. Harring, B. Weiss, and J.-C. Hsu. A comparison of methods for estimating quadratic effects in nonlinear structural equation models. *Psychological methods*, 17:193–214, 2012. doi: 10.1037/a0027539. (Cited on page 14.)

S. Haykin. *Adaptive Filter Theory*. Prentice Hall, 4$^{\text{th}}$ edition, 2002. ISBN 9780130901262. (Cited on pages 46 and 80.)

R. H. Heiberger. Predicting economic growth with stock networks. *Physica A: Statistical Mechanics and its Applications*, 489:102–111, 2018. (Cited on pages 36 and 62.)

M. H. Histed, V. Bonin, and R. C. Reid. Direct activation of sparse, distributed populations of cortical neurons by electrical microstimulation. *Neuron*, 63(4):508–522, 2009. (Cited on page 20.)

F. Hua, R. Nassif, C. Richard, H. Wang, and A. H. Sayed. A preconditioned graph diffusion LMS for adaptive graph signal processing. In *Proc. European Conference on Signal Processing (EUSIPCO)*, pages 1–5, Rome, Italy, 2018. (Cited on page 21.)

W. Huang, T. A. W. Bolton, J. D. Medaglia, D. S. Bassett, A. Ribeiro, and D. van De Ville. A graph signal processing perspective on functional brain imaging. *Proceedings of the IEEE*, 106 (5):868–885, 2018. (Cited on page 10.)

I. Jabłoński. Graph signal processing in applications to sensor networks, smart grids, and smart cities. *IEEE Sensors Journal*, 17(23):7659–7666, 2017. (Cited on page 2.)

J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta mathematica*, 30:175–193, 1906. (Cited on page 75.)

D. Jin, J. Chen, C. Richard, and J. Chen. Adaptive parameters adjustement for group reweighted zero-attracting LMS. In *Acoustics, Speech and Signal Processing (ICASSP), Proc. 2018 IEEE International Conference on*, 2018a. (Cited on page 41.)

D. Jin, J. Chen, C. Richard, and J. Chen. Model-driven online parameter adjustment for zero-attracting LMS. *Signal Processing*, 152:373 – 383, 2018b. ISSN 0165-1684. doi: https://doi.org/10.1016/j.sigpro.2018.06.020. URL http://www.sciencedirect.com/science/article/pii/S0165168418302172. (Cited on pages 25 and 41.)

R. Jin, S. C. H. Hoi, and T. Yang. Online multiple kernel learning: Algorithms and mistake bounds. In *International conference on algorithmic learning theory*, pages 390–404. Springer, 2010. (Cited on page 91.)

K. G. Jöreskog, F. Yang, G. Marcoulides, and R. Schumacker. Nonlinear structural equation models: The Kenny-Judd model with interaction effects. *Advanced structural equation modeling: Issues and techniques*, pages 57–88, 1996. (Cited on pages 39 and 65.)

V. Kalofolias. How to learn a graph from smooth signals. In *Proc. International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 920–929, Cadiz, Spain, 09–11 May 2016. PMLR. (Cited on page 19.)

D. Kim, J. M. Paggi, C. Park, C. Bennett, and S. L. Salzberg. Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype. *Nature biotechnology*, 37(8):907–915, 2019. (Cited on page 2.)

G. Kitagawa. An algorithm for solving the matrix equation $X = FXF^\top + S$. *International Journal of Control*, 25(5):745–753, 1977. doi: 10.1080/00207177708922266. URL https://doi.org/10.1080/00207177708922266. (Cited on page 29.)

J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999. (Cited on page 97.)

M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. ℓ_p-norm multiple kernel learning. *The Journal of Machine Learning Research*, 12:953–997, 2011. (Cited on page 91.)

M. Kramer, E. D. Kolaczyk, and H. Kirsch. Emergent network topology at seizure onset in humans. *Epilepsy research*, 79:173–86, 2008. doi: 10.1016/j.eplepsyres.2008.02.002. (Cited on pages 36, 51, 62, and 83.)

A. Kumar. Expectation of product of quadratic forms. *Sankhyā: The Indian Journal of Statistics, Series B*, pages 359–362, 1973. (Cited on page 79.)

S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien. Gradient-based neural DAG learning. *arXiv preprint arXiv:1906.02226*, 2019. (Cited on page 92.)

O. Ledoit and M. Wolf. Honey, I shrunk the sample covariance matrix. *The Journal of Portfolio Management*, 30(4):110–119, 2004. ISSN 0095-4918. doi: 10.3905/jpm.2004.110. URL https://jpm.pm-research.com/content/30/4/110. (Cited on page 69.)

Y. C. Lee and A. Y. Zomaya. Energy conscious scheduling for distributed computing systems under different operating conditions. *IEEE Transactions on Parallel and Distributed Systems*, 22(8):1374–1381, 2010. (Cited on page 13.)

K. Lerman, R. Ghosh, and T. Surachawala. Social contagion: An empirical study of information spread on Digg and Twitter follower graphs. *arXiv preprint arXiv:1202.3162*, 2012. (Cited on page 20.)

M.-J. Lesot, F. Nel, T. Delavallade, P. Capet, and B. Bouchon-Meunier. Two methods for internet buzz detection exploiting the citation graph. In *2012 IEEE International Conference on Fuzzy Systems*, pages 1–8. IEEE, 2012. (Cited on page 2.)

N. Lim, F. D'Alché-Buc, C. Auliac, and G. Michailidis. Operator-valued kernel-based vector autoregressive models for network inference. *Machine learning*, 99(3):489–513, 2015. (Cited on page 14.)

C. Lippert, O. Stegle, Z. Ghahramani, and K. Borgwardt. A kernel method for unsupervised structured network inference. In *Artificial Intelligence and Statistics*, pages 368–375, 2009. (Cited on page 15.)

W. Liu, J. C. Príncipe, and S. Haykin. *Kernel adaptive filtering: a comprehensive introduction*, chapter 1, pages 1–26. John Wiley & Sons, Ltd, 2010. ISBN 9780470608593. doi: 10.1002/9780470608593.ch1. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470608593.ch1. (Cited on pages 44 and 71.)

E. N. Lorenz. Computational chaos-a prelude to computational instability. *Physica D: Nonlinear Phenomena*, 35(3):299 – 317, 1989. ISSN 0167-2789. doi: https://doi.org/10.1016/0167-2789(89)90072-9. (Cited on page 50.)

I. Lukovits. Correlation between components of the wiener index and partition coefficients of hydrocarbons. *International Journal of Quantum Chemistry*, 44(S19):217–223, 1992. (Cited on page 2.)

B. Marr. How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read, May 2018. URL https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/. Section: Innovation. (Cited on page 89.)

J. Mei and J. M. F. Moura. Signal processing on graphs: Causal modeling of unstructured data. *IEEE Transactions on Signal Processing*, 65(8):2077–2092, 2017. ISSN 1053-587X. doi: 10.1109/TSP.2016.2634543. (Cited on pages 22, 23, and 32.)

Z. Mi, Y. Yang, and G. Liu. HERO: A hybrid connectivity restoration framework for mobile multi-agent networks. In *2011 IEEE International Conference on Robotics and Automation*, pages 1702–1707, 2011. (Cited on page 90.)

J. Minkoff. Comment on the "Unnecessary assumption of statistical independence between reference signal and filter weights in feedforward adaptive systems". *IEEE Transactions on Signal Processing*, 49(5):1109–, 2001. (Cited on pages 44 and 72.)

M. Moscu, R. Nassif, F. Hua, and C. Richard. Apprentissage distribué de la topologie d'un graphe à partir de signaux temporels sur graphe. In *Actes du 27e Colloque GRETSI sur le Traitement du Signal et des Images*, 2019. (Not cited.)

M. Moscu, R. Nassif, F. Hua, and C. Richard. Learning causal networks topology from streaming graph signals. In *27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019. doi: 10.23919/EUSIPCO.2019.8902826. (Not cited.)

M. Moscu, R. Borsoi, and C. Richard. Online graph topology inference with kernels for brain connectivity estimation. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2020a. (Cited on pages 63, 65, and 84.)

M. Moscu, R. Borsoi, and C. Richard. Graph Topology Inference with Kernels and Partial-derivative-imposed Sparsity: Algorithm and Convergence Analysis. 2020b. submitted. (Not cited.)

M. Moscu, R. Borsoi, and C. Richard. Convergence analysis of the graph-topology-inference kernel LMS algorithm. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021a. (Not cited.)

M. Moscu, R. Borsoi, and C. Richard. Online kernel-based graph topology identification with partial-derivative-imposed sparsity. In *28th European Signal Processing Conference (EUSIPCO)*, pages 2190–2194, 2021b. doi: 10.23919/Eusipco47968.2020.9287624. (Not cited.)

S. Mukherjee and Q. Wu. Estimation of gradients and coordinate covariation in classification. *Journal of Machine Learning Research*, 7(Nov):2481–2514, 2006. (Cited on page 67.)

S. K. Narang, A. Gadde, and A. Ortega. Signal processing techniques for interpolation in graph structured data. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5445–5449. IEEE, 2013. (Cited on page 2.)

R. Nassif, C. Richard, J. Chen, R. Couillet, and P. Borgnat. Filtrage lms sur graphe. algorithme et analyse. In *Actes du 26e Colloque GRETSI sur le Traitement du Signal et des Images*, 2017a. (Cited on page 1.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Diffusion LMS for multitask problems with local linear equality constraints. *IEEE Transactions on Signal Processing*, 65(19):4979 – 4993, 2017b. (Cited on pages 1 and 24.)

R. Nassif, C. Richard, J. Chen, and A. H. Sayed. Distributed diffusion adaptation over graph signals. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4129–4133, Calgary, Canada, 2018. (Cited on page 21.)

A. Natali, M. Coutino, and G. Leus. Topology-aware joint graph filter and edge weight identification for network processes. *arXiv preprint arXiv:2007.03266*, 2020. (Cited on page 19.)

M. Nauta, D. Bucur, and C. Seifert. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1(1):312–340, 2019. (Cited on page 92.)

E. Olejarczyk and W. Jernajczyk. Graph-based analysis of brain connectivity in schizophrenia. *PLOS ONE*, 12(11):1–28, November 2017. doi: 10.1371/journal.pone.0188629. URL https://doi.org/10.1371/journal.pone.0188629. (Cited on page 55.)

J. Omura and T. Kailath. *Useful Probability Distributions*, volume 2. Stanford Electronics, September 1965. (Cited on page 44.)

W. D. Parreira, J.-C. M. Bermudez, C. Richard, and J.-Y. Tourneret. Stochastic behavior analysis of the Gaussian kernel least-mean-square algorithm. *IEEE Transactions on Signal Processing*, 60(5):2208–2222, 2012. (Cited on page 72.)

A. Paul. Graph based M2M optimization in an IoT environment. In *Proceedings of the 2013 Research in Adaptive and Convergent Systems*, pages 45–46, 2013. (Cited on page 7.)

V. I. Paulsen and M. Raghupathi. *An introduction to the theory of reproducing kernel Hilbert spaces*, volume 152. Cambridge University Press, 2016. (Cited on page 39.)

W. G. Penfield. Ferrier lecture-some observations on the cerebral cortex of man. *Proceedings of the Royal Society of London. Series B-Biological Sciences*, 134(876):329–347, 1947. (Cited on page 20.)

N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond. GSPBOX: A toolbox for signal processing on graphs. *ArXiv e-prints*, August 2014. (Cited on page 29.)

S. Petkoski and V. K. Jirsa. Transmission time delays organize the brain network synchronization. *A Philosophical Transactions of the Royal Society*, 377(2153), 2019. (Cited on page 38.)

S. C. Ponten, F. Bartolomei, and C. J. Stam. Small-world networks and epilepsy: graph theoretical analysis of intracerebrally recorded mesial temporal lobe seizures. *Clinical neurophysiology*, 118(4):918–927, 2007. (Cited on page 52.)

M. Ramezani-Mayiami and B. Beferull-Lozano. Graph recursive least squares filter for topology inference in causal data processes. In *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5, 2017. (Cited on page 14.)

M. I. Razzak, M. Imran, and G. Xu. Big data analytics for preventive medicine. *Neural Computing and Applications*, 32(9):4417–4451, 2020. (Cited on page 89.)

C. Richard, J.-C. M. Bermudez, and P. Honeine. Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57(3):1058–1067, 2009. (Cited on pages 42 and 68.)

L. Rosasco, S. Villa, S. Mosci, M. Santoro, and A. Verri. Nonparametric sparsity and regularization. *Journal of Machine Learning Research*, 14:1665–1714, 2013. (Cited on pages 64 and 66.)

M. Rubinov and O. Sporns. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, 52(3):1059–1069, 2010. (Cited on page 63.)

P. Salembier, S. Liesegang, and C. López-Martínez. Ship detection in SAR images based on maxtree representation and graph signal processing. *IEEE Transactions on Geoscience and Remote Sensing*, 57(5):2709–2724, 2018. (Cited on page 2.)

A. Sandryhaila and J. M. F. Moura. Discrete signal processing on graphs. *IEEE Transactions on Signal Processing*, 61(7):1644–1656, 2013a. ISSN 1053-587X. doi: 10.1109/TSP.2013.2238935. (Cited on pages 18 and 20.)

A. Sandryhaila and J. M. F. Moura. Discrete signal processing on graphs: Graph filters. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6163–6166, Vancouver, Canada, 2013b. (Cited on pages 10 and 20.)

A. Sandryhaila and J. M. F. Moura. Discrete signal processing on graphs: Graph Fourier transform. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6167–6170. IEEE, 2013c. (Cited on page 10.)

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. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014. ISSN 1053-5888. doi: 10.1109/MSP.2014.2329213. (Cited on page 1.)

A. Sanfeliu, R. Alquézar, J. Andrade, J. Climent, F. Serratosa, and J. Vergés. Graph-based representations and techniques for image processing and image analysis. *Pattern recognition*, 35(3):639–650, 2002. (Cited on page 1.)

S. Sardellitti, S. Barbarossa, and P. Di Lorenzo. Graph topology inference based on transform learning. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 356–360, 2016. (Cited on page 14.)

S. Sardellitti, S. Barbarossa, and P. Di Lorenzo. On the graph Fourier transform for directed graphs. *IEEE Journal of Selected Topics in Signal Processing*, 11(6):796–811, 2017. (Cited on page 9.)

S. Sardellitti, S. Barbarossa, and P. D. Lorenzo. Graph topology inference based on sparsifying transform learning. *IEEE Transactions on Signal Processing*, 67(7):1712–1727, 2019. (Cited on page 14.)

A. H. Sayed. *Adaptive Filters*. John Wiley & Sons, 2008. (Cited on pages 24, 25, and 28.)

K. A. Schindler, S. Bialonski, M.-T. Horstmann, C. E. Elger, and K. Lehnertz. Evolving functional network properties and synchronizability during human epileptic seizures. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18(3):033119, 2008. (Cited on page 52.)

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *International conference on computational learning theory*, pages 416–426. Springer, 2001. (Cited on page 40.)

P. Schwab and W. Karlen. CXPlain: Causal explanations for model interpretation under uncertainty. In *Advances in Neural Information Processing Systems*, pages 10220–10230, 2019. (Cited on page 92.)

S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro. Network topology inference from spectral templates. *IEEE Transactions on Signal and Information Processing over Networks*, 3(3):467–483, 2017. ISSN 2373-776X. doi: 10.1109/TSIPN.2017.2731051. (Cited on pages 14 and 19.)

S. Segarra, S. P. Chepuri, A. G. Marques, and G. Leus. Statistical graph signal processing: Stationarity and spectral estimation. In *Cooperative and Graph Signal Processing*, pages 325–347. Elsevier, 2018. (Cited on page 19.)

R. Shafipour and G. Mateos. Online topology inference from streaming stationary graph signals with partial connectivity information. *arXiv preprint arXiv:2007.03653*, 2020. (Cited on page 19.)

R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos. Network topology inference from non-stationary graph signals. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5870–5874, 2017. (Cited on page 14.)

R. Shafipour, A. Hashemi, G. Mateos, and H. Vikalo. Online topology inference from streaming stationary graph signals. *IEEE Data Science Workshop (DSW)*, pages 140–144, 2019. (Cited on pages 14 and 19.)

D. Shah and T. Zaman. Rumors in a network: Who's the culprit? *IEEE Transactions on information theory*, 57(8):5163–5181, 2011. (Cited on page 2.)

Y. Shen, B. Baingana, and G. B. Giannakis. Kernel-based structural equation models for topology identification of directed networks. *IEEE Transactions on Signal Processing*, 65(10):2503–2516, 2017. (Cited on page 63.)

Y. Shen, B. Baingana, and G. B. Giannakis. Topology inference of directed graphs using nonlinear structural vector autoregressive models. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6513–6517, 2017. doi: 10.1109/ICASSP.2017.7953411. (Cited on pages 15 and 37.)

Y. Shen, T. Chen, and G. B. Giannakis. Random Feature-Based Online Multi-Kernel Learning in Environments with Unknown Dynamics. *The Journal of Machine Learning Research*, 20(1):773–808, January 2019. ISSN 1532-4435. (Cited on page 91.)

Y. Shen, G. B. Giannakis, and B. Baingana. Nonlinear structural vector autoregressive models with application to directed brain networks. *IEEE Transactions on Signal Processing*, 67(20):5325–5339, 2019. (Cited on pages 42, 52, 63, 69, and 84.)

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. (Cited on page 31.)

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. (Cited on pages 8 and 18.)

D. I. Shuman, B. Ricaud, and P. Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260–291, 2016. (Cited on page 10.)

D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard. Distributed signal processing via Chebyshev polynomial approximation. *IEEE Transactions on Signal and Information Processing over Networks*, 4(4):736–751, 2018. (Cited on page 20.)

SINTEF. Big Data, for better or worse: 90% of world's data generated over last two years, May 2013. URL https://www.sciencedaily.com/releases/2013/05/130522085217.htm. (Cited on page 89.)

K. C. Skåtun, T. Kaufmann, S. Tønnesen, G. Biele, I. Melle, I. Agartz, D. Alnæs, O. A. Andreassen, and L. T. Westlye. Global brain connectivity alterations in patients with schizophrenia and bipolar spectrum disorders. *Journal of psychiatry & neuroscience: JPN*, 41(5):331, 2016. (Cited on page 56.)

X.-Y. Song, Z.-H. Lu, J.-H. Cai, and E. H.-S. Ip. ABayesian modeling approach for generalized semiparametric structural equation models. *Psychometrika*, 78(4):624–647, 2013. (Cited on pages 39 and 65.)

P. Spachos and K. Plataniotis. Beacons and the City: Smart Internet of Things. In P. M. Djurić and C. Richard, editors, *Cooperative and Graph Signal Processing*, pages 757–776. Academic Press, 2018. (Cited on page 10.)

M. J. M. Spelta and W. A. Martins. Online temperature estimation using graph signals. *XXXVI Simpósio Brasileiro de Telecomunicaçoes e Processamento de Sinais-SBrT2018*, pages 154–158, 2018. (Cited on page 1.)

M. Speriosu, N. Sudan, S. Upadhyay, and J. Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 53–63, 2011. (Cited on page 25.)

R. W. Sperry, M. S. Gazzaniga, and J. E. Bogen. Interhemispheric relationships: the neocortical commissures; syndromes of hemisphere disconnection. *Handbook of clinical neurology*, 4:273–290, 1969. (Cited on page 37.)

O. Sporns. *Networks of the Brain*. MIT press, 2010. (Cited on page 37.)

The International HapMap Consortium, K.A. Frazer, D.G. Ballinger, et al. A second generation human haplotype map of over 3.1 million SNPs. *Nature*, 449:851 EP –, 10 2007. URL https://doi.org/10.1038/nature06258. (Cited on pages 36 and 62.)

F. A. Tobar, S. Kung, and D. P. Mandic. Multikernel least mean square algorithm. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2):265–277, 2014. ISSN 2162-237X. doi: 10.1109/TNNLS.2013.2272594. (Cited on page 50.)

Z. J. Towfic and A. H. Sayed. Adaptive penalty-based distributed stochastic convex optimization. *IEEE Transactions on Signal Processing*, 62(15):3924–3938, 2014. ISSN 1053-587X. doi: 10.1109/TSP.2014.2331615. (Cited on page 24.)

N. Tremblay, G. Puy, P. Borgnat, R. Gribonval, and P. Vandergheynst. Accelerated spectral clustering using graph filtering of random signals. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4094–4098, Shanghai, China, 2016. IEEE. (Cited on page 31.)

A. Tremeau and P. Colantoni. Regions adjacency graph applied to color image segmentation. *IEEE Transactions on Image Processing*, 9(4):735–744, 2000. (Cited on page 1.)

E. van Diessen, S. J. H. Diederen, K. P. J. Braun, F. E. Jansen, and C. J. Stam. Functional and structural brain networks in epilepsy: What have we learned? *Epilepsia*, 54(11):1855–1865,

2013. doi: 10.1111/epi.12350. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/epi.12350. (Cited on page 52.)

A. Venkataraman, T. J. Whitford, C.-F. Westin, P. Golland, and M. Kubicki. Whole brain resting state functional connectivity abnormalities in schizophrenia. *Schizophrenia Research*, 139(1): 7 – 12, 2012. ISSN 0920-9964. doi: https://doi.org/10.1016/j.schres.2012.04.021. URL http://www.sciencedirect.com/science/article/pii/S0920996412002538. (Cited on page 56.)

J.-P. Vert and Y. Yamanishi. Supervised graph inference. In *Advances in neural information processing systems*, pages 1433–1440, 2005. (Cited on page 37.)

S. Vlaski, H. P. Maretić, R. Nassif, P. Frossard, and A. H. Sayed. Online graph learning from sequential data. In *Proc. IEEE Data Science Workshop*, pages 190–194, Lausanne, Switzerland, 2018. (Cited on page 14.)

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001. URL http://papers.nips.cc/paper/1866-using-the-nystrom-method-to-speed-up-kernel-machines.pdf. (Cited on page 41.)

Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 15(11):1101–1113, 1993. (Cited on page 31.)

K. Yamada, Y. Tanaka, and A. Ortega. Time-varying graph learning based on sparseness of temporal variation. In *ICASSP 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5411–5415. IEEE, 2019. (Cited on page 19.)

G.-B. Ye and X. Xie. Learning sparse gradients for variable selection and dimension reduction. *Machine learning*, 87(3):303–355, 2012. (Cited on page 67.)

C. Yu and A. H. Sayed. Learning by networked agents under partial information. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3874–3878, New Orleans, USA, 2017. doi: 10.1109/ICASSP.2017.7952882. (Cited on page 24.)

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006. (Cited on page 41.)

A. Zalesky, A. Fornito, I. H. Harding, L. Cocchi, M. Yücel, C. Pantelis, and E. T. Bullmore. Whole-brain anatomical networks: does the choice of nodes matter? *Neuroimage*, 50(3): 970–983, 2010. (Cited on page 37.)

B. Zaman, L. M. Lopez-Ramos, D. Romero, and B. Beferull-Lozano. Online topology estimation for vector autoregressive processes in data networks. In *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5. IEEE, 2017. (Cited on page 14.)

L. Zhang, G. Wang, and G. B. Giannakis. Going beyond linear dependencies to unveil connectivity of meshed grids. In *Proc. IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5, Curaçao, Dutch Antilles, 2017. doi: 10.1109/CAMSAP.2017.8313078. (Cited on pages 15 and 63.)

D.-X. Zhou. Derivative reproducing properties for kernel methods in learning theory. *Journal of computational and Applied Mathematics*, 220(1-2):456–463, 2008. (Cited on page 66.)

L. Zhou, A.-X. Feng, R. Zhi, and Z.-Q. Gong. Topological analysis of temperature networks using bipartite graph model. *Acta Physica Sinica*, 59(9):6689–6696, 2010. (Cited on page 1.)