Complex networks
topology, spectrum and linear processes

Jokic, I.

DOI
10.4233/uuid:71e0ed4c-e8ea-404c-9846-b95bce6d17ed

Publication date
2023

Document Version
Final published version

Citation (APA)
Jokic, I. (2023). *Complex networks: topology, spectrum and linear processes*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:71e0ed4c-e8ea-404c-9846-b95bce6d17ed

COMPLEX NETWORKS

TOPOLOGY, SPECTRUM AND LINEAR PROCESSES

Complex Networks

Topology, Spectrum and Linear Processes

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates,
to be defended publicly on,
Thursday 14 September 2023 at 10:00 o'clock

by

Ivan Jokić

Master of Science in Power Systems and Automatic Control
University of Montenegro, Podgorica, Montenegro
born in Bijelo Polje, Montenegro.

This dissertation has been approved by the promotor:
Prof. dr. ir. P.F.A. Van Mieghem

Composition of the doctoral committee:

Rector Magnificus	chairperson
Prof. dr. ir. P.F.A. Van Mieghem	Delft University of Technology, promotor
Prof. dr. ir. B. De Schutter	Delft University of Technology, copromotor
Independent members:	
Prof. dr. ir. R.E. Kooij	Delft University of Technology
Prof. dr. ir. G.J.T. Leus	Delft University of Technology
Prof. dr. F. P. Pijpers	University of Amsterdam, Statistics Netherlands
Prof. dr. H. Cherifi	University of Burgundy, France
Dr. ir. E. Smeitink	Delft University of Technology, KPN Royal
Prof. dr. ir. G. Gaydadjiev	Delft University of Technology, reserve member

Keywords: graph spectra, paths, effective resistance, inverse shortest path, graph sparsification, clustering, linear process, networked systems

Printed by: Ipskamp Printing, Enschede

Front & Back: Designed by Ivan Jokić

Published by: Ivan Jokić

Email: Ivan.Jokic.93@gmail.com.

ISBN 978-94-6473-200-9

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

To my family and my love Jelena

Contents

SUMMARY

The concept of a network, defined as a collection of interconnected nodes or entities, has become a foundation for a new field of inquiry, namely network science. Despite the apparent simplicity of the concept, the pairwise representation of interconnecting nodes has enabled a plethora of insights into the structure of networks and the effects of interactions on dynamic processes. This generality of the network concept has paved the way for novel approaches with the aim of understanding complex systems, from social networks to biological pathways. It has opened up new avenues for research into the fundamental mechanisms underlying these systems. As such, network science has become a highly active and dynamic field, driving the development of new theoretical frameworks, computational tools, and empirical methods that continuously push the boundaries of knowledge and understanding in numerous science and engineering domains.

The first part of this thesis centres on the structural properties of complex networks and their practical applications. We demonstrate that the orthogonal eigenvectors of the adjacency matrix of a simple, unweighted, and undirected graph are sufficient to recover that graph, albeit potentially not in a unique manner (Chapter 2). This observation led us to uncover co-eigenvector graphs, which are graphs that share the same eigenvectors while having distinct eigenvalues. Co-eigenvector graphs are the dual counterparts of cospectral graphs, which share identical eigenvalues but possess distinct eigenvectors. In an unweighted graph, the number of walks between node pairs of a particular length can be expressed in terms of the corresponding power of the adjacency matrix. However, deriving a similar solution for the number of paths is significantly more intricate (Chapter 3). We present three distinct analytical solutions in matrix form for computing the number of paths of any length between node pairs, utilising different types of walks and leveraging principles from the mathematical field of combinatorics. The computational complexity of these solutions varies depending on the sparsity of the graph. The effective resistance metric, which characterises the entire network as perceived from the vantage point of two given nodes, represents a powerful tool for addressing a wide range of challenges in network theory. In Chapter 4, we leverage the information contained in effective resistance to solve the inverse all shortest path problem, wherein a weighted graph satisfying given upper bounds on the shortest path weights between node pairs is sought, with sparsity being a critical consideration. Additionally, we propose a novel graph sparsification algorithm that selectively removes links from an unweighted graph in a stepwise manner, with the goal of either minimising or maximising the effective resistance of the resultant graph.

The second part of this thesis pertains to linear processes on complex networks, exploring their properties and applications. Our research reveals that a simple process of attraction and repulsion between adjacent nodes on a one-dimensional line, based on the similarity of their neighbourhoods, can effectively group together nodes from the same community (Chapter 5). Our linear clustering process generally produces more accurate partitions than the most prevalent modularity-based clustering methods in the literature, requiring a comparable amount of computational complexity. An empirical part of our research on processes in complex networks became possible thanks to our network construction based on a unique data set containing each municipality's area, population and its geographically adjacent neighbouring municipalities. Thanks to this network construction, research became possible on a dynamic network of connected municipal nodes at a national level over the period from 1830 to 2019 (Chapter 6). By connecting the population data, area data and municipal merger data of all Dutch municipalities, we discovered that the logarithm of the municipal area and population size yields an almost linear difference equation over time. Research into the municipal merger process over the period 1830-2019 has shown that 873 of the 1228 Dutch municipalities have merged into adjacent larger municipalities with a larger population. Our simulation of municipality mergers based on network effects caused by population growth by municipality resulted in a county-level predictive accuracy of 91.7 % over a 200-year period. Suppose every node within a network exhibits linear internal dynamics of a specific order, and the dynamic interactions between these nodes are also linear. In that case, the entire network conforms to a collection of linear differential equations (Chapter 7). Our study offers an analytical solution for the comprehensive network dynamics in state space form, achieved by merging the fundamental topology and internal linear dynamics of every individual node.

Samenvatting

Het concept van een netwerk, gedefinieerd als verzameling van verbonden nodes, is het fundament geworden voor het domein der netwerkwetenschap. Ondanks de klaarblijkelijke eenvoud van dit concept levert de paarsgewijze representatie van verbonden nodes waardevolle inzichten op over netwerkstructuren en de interactie effecten hiervan op dynamische processen. Deze generalisatie van het netwerk concept heeft de weg geopend naar nieuwe onderzoeksmethoden met het doel om complexe systemen te begrijpen. Bijvoorbeeld voor sociale, economische en biologische netwerken is onderzoek mogelijk geworden naar de fundamentele mechanismen onder deze systemen. De netwerkwetenschap is een dynamisch onderzoeksdomein geworden dat de ontwikkeling van nieuwe theoretische kaders, rekenhulpmiddelen en empirische methoden oplevert die continu de grenzen verleggen van kennis en begrip in vele toegepaste onderzoeks- en engineering domeinen.

Het eerste deel van deze thesis gaat over structuureigenschappen van complexe netwerken en hun praktische toepassingen. We demonstreren dat het mogelijk is om met de orthogonale eigenvectoren van de adjacency matrix van een simpele, ongewogen en ongerichte graaf, deze graaf te reconstrueren, al is het niet altijd op een unieke manier (hoofdstuk 2). Deze observatie heeft geleid tot onderzoek aan co-eigenvector grafen. Co-eigenvector grafen hebben gelijke eigenvectoren maar hebben verschillende eigenwaarden. Co-eigenvector grafen kunnen beschouwd worden als tegenhangers van cospectrale grafen, die gelijke eigenwaarden hebben maar verschillende eigen-vectoren bezitten. In een ongewogen graaf kan het aantal wandelingen van een bepaalde lengte tussen node paren uitgedrukt worden in termen van de corresponderende kracht van de adjacency matrix. Echter, het afleiden van een vergelijkbare oplossing voor het bepalen van het aantal paden is aanzienlijk complexer (hoofdstuk 3). We presenteren drie verschillende analytische oplossingen in matrix vorm voor het berekenen van het aantal paden van alle denkbare lengtes tussen node paren, gebruikmakend van verschillende typen wandelingen waarbij principes uit het wiskundige gebied van de combinatoriek worden toegepast. De computationele complexiteit van deze oplossingen varieert afhankelijk van de dichtheid van een graaf. De metriek van de effectieve weerstand die een geheel netwerk karakteriseert zoals waargenomen vanuit het gezichtspunt van twee gegeven nodes, vertegenwoordigt een krachtig hulpmiddel voor de aanpak van een breed scala aan uitdagingen in de netwerktheorie. In hoofdstuk 4 benutten we informatie, zoals besloten in de effectieve weerstand, om het inverse kortste pad probleem op te lossen, waarin een gewogen graaf wordt gezocht die voldoet aan gegeven bovengrenzen van de gewichten van de kortste paden tussen node paren met netwerk dichtheid als kritische overweging. In aanvulling stellen we een nieuw graaf dichtheidsalgoritme voor dat op een stapsgewijze manier selectief linken verwijdert uit een ongewogen graaf met het doel om de effectieve weerstand van de resulterende graaf te minimaliseren of te maximaliseren.

Het tweede deel van deze thesis gaat over lineaire processen in complexe netwerken en het verkennen van hun eigenschappen en toepassingen. Onze research laat zien dat een proces van aantrekken en afstoten tussen twee aangrenzende nodes op een een-dimensionele lijn, gebaseerd op gelijkenis van hun community, de nodes behorend tot dezelfde community effectief kan groeperen (hoofdstuk 5). Ons lineaire clusterproces produceert meer accurate partities dan de meeste prevalente op modulariteit gebaseerde clustermethoden (bekend uit de literatuur) bij vergelijkbare rekencomplexiteit. Een empirisch deel van onze research op het gebied van processen in complexe netwerken werd mogelijk dankzij onze netwerkconstructie op basis van een unieke data set die van elke afzonderlijke gemeente de oppervlakte, de bevolking en de geografisch aangrenzende buurgemeenten bevat. Dankzij deze netwerkconstructie werd onderzoek mogelijk aan een dynamisch netwerk van verbonden gemeentelijke nodes op nationaal niveau over de periode van 1830 tot en met 2019 (hoofdstuk 6). Door van alle Nederlandse gemeenten de bevolkings data, oppervlakte data en gemeentefusie data te verbinden, hebben we ontdekt dat de logaritme van de gemeentelijke oppervlakte en de bevolkingsomvang een vrijwel lineaire verschil vergelijking over de tijd oplevert. Uit onderzoek aan het gemeente fusieproces over de periode 1830-2019 is gebleken dat 873 van de 1228 Nederlandse gemeenten zijn opgegaan in aangrenzende grotere gemeenten met een grotere bevolksomvang. Onze simulatie van gemeentefusies op basis van netwerkeffecten die zijn veroorzaakt door bevolkingsgroei per gemeente, resulteerde in een voorspellende nauwkeurigheid van 91.7 % op provinciaal niveau over een periode van 200 jaar. Stel dat elke node in een netwerk specifieke lineaire interne dynamiek laat zien en dat de dynamische interactie tussen deze nodes ook lineair is; in dat geval conformeert het gehele netwerk zich aan een verzameling lineaire differentiaalvergelijkingen (hoofdstuk 7). Onze research biedt een analytische oplossing voor de netwerk dynamiek in state space vorm voor het gehele netwerk, bereikt door samenvoegen van de fundamentele topologie en interne lineaire dynamiek van elke individuele node.

1

INTRODUCTION

Of all the frictional resistances,
the one that most retards human movement is ignorance.

Nikola Tesla

NETWORKS [1, 2] abound and increasingly shape our world, ranging from infrastructural networks (transportation, telecommunication, power grids, water, etc.) over social networks to brain and biological networks. In network science, a network generally consists of the underlying topology, defined by a graph and the dynamic process on the network.

1.1. TOPOLOGY AND SPECTRUM OF A GRAPH

The inception of modern graph theory can be traced back to Euler's work on the Konigsberg seven-bridge problem [3]. Since then, the concept of a graph has garnered the attention of researchers from a diverse range of disciplines due to its versatility. In this thesis, we specifically consider simple graphs, which are graphs that do not contain self-loops or multiple links. The adjacency matrix is the most straightforward way to represent the topology of a graph. This matrix captures node-pair connections and is known as the *topology domain*. When raised to a particular power, the adjacency matrix of an undirected graph contains information about the number of walks between node pairs of the corresponding length [4]. Eigenvalue decomposition of the adjacency matrix provides an equivalent graph representation, referred to as the *spectral domain*. In the case of a linear process on a network, the eigenvalues of the governing graph-based matrix determine how the process evolves along orthogonal directions, defined by the corresponding eigenvectors. Additionally, graph spectra play an essential role in analysing different nonlinear processes taking place on networks, such as synchronisation [5], epidemic spreading [6], and more. In addition to the topology and spectral domains, a third equivalent representation exists, called the *geometric domain*. In this domain, each possibly weighted undirected graph in this domain is a simplex in Euclidean space [7].

The community structure is one of the most crucial topological features of networks. Identifying communities and their corresponding hierarchical structure in real-world networks has been a topic of active research for several decades [8] and is essential in many applications. However, there is no single, precise definition of a community [9, 10]. In network science, a community is defined as a set of nodes that share links primarily with one another, with only a minority of links shared with other nodes in the network. Modularity, proposed by Newman and Girvan [11], is a commonly used quality function for a given graph partition. It compares the number of links between nodes from the same community with the expected number of intra-community links in a network with randomly connected nodes. The clustering problem is very challenging and remains an active topic of research. Numerous approaches have been proposed based on modularity optimisation [12–14], spectral decomposition of a graph-related matrix [15–19], and different processes taking place on networks, such as Potts models [20], superparamagnetic clustering [21], and finding attractors of dynamics [22].

Effective resistance is another key topological feature of networks that has gained increasing attention from researchers over time [23]. Originating from electrical systems theory, the concept of effective resistance between a pair of nodes specifies how power dissipates over the entire network as electrical energy is transmitted between the nodes. The concept is highly relevant to general network theory because effective resistance, as a metric, provides a description of the entire network from the perspective of two nodes. Therefore, effective resistance has found application in numerous graph problems, such as graph sparsification [24], random walks [25], and clustering [26].

The study of the topological and spectral properties of networks has provided a wealth of insights into how network topology affects its operation. These findings have also led to the development of the inverse approach, which involves recovering a network from its topological features. Inverse graph problems can be addressed when certain information about a graph's topology or spectrum is provided. The inverse shortest path problem (ISPP) is one such problem, which assumes that the upper bounds of the shortest path weights between node pairs are known, and aims to reconstruct a weighted graph (with as few edges as possible) with the same shortest path weight distribution. ISPP is a generally NP-hard problem. However, if the shortest path weights are computed for a tree graph, an analytical solution can be obtained using an analogy with the electric network of resistors [27]. A common approach for solving inverse graph problems is the iterative addition or removal of links. Therefore, graph sparsification approaches are crucial for inverse graph problems. Graph sparsification involves removing links from a graph and redistributing link weights in a way that minimises the change in a specific network metric, such as eigenvalues of the adjacency matrix or effective graph resistance. Various stochastic approaches have been proposed that utilise either effective resistance [24, 28, 29] or the graph spectrum [30].

1.2. DYNAMIC PROCESSES ON A NETWORK

Newman [31] observed that the progress in analysing the structural properties of the network has been faster than the one related to the dynamics taking place on the network.

Barzel, Harush *et al.* [32–34] showed that, while many real networks tend to have similar (universal) structural properties, there exist classes of dynamical processes that exhibit fundamentally different flow patterns. During the last two decades, dynamical processes on complex networks such as phase transitions [35], percolation [36], synchronization [37], diffusion [38], epidemic spreading [39–42], collective behaviour [43] and traffic [44] have been intensively researched [45].

The network dynamics depend on the network topology and the type of dynamic interaction between the nodes. The interplay between the network topology and dynamics has been an active field of scientific research in the past two decades [45]. The dynamics of the real-world networks are non-linear, and their underlying topology is time-varying [46]. However, complex networks with linear dynamics have been intensively researched recently [47, 48]. Linear processes on networks allow for the analytic solution of the dynamics evolution over time, using the eigenvalue decomposition. In addition, networked systems with linear dynamics allow for hierarchical structuring, i.e. providing the analytic solution for the network's system dynamics on different aggregation levels, without losing any information about individual systems dynamics [49].

1.3. NOTATION

The following notation is used throughout this thesis. The $N \times 1$ all-one vector is denoted as u, the $N \times N$ identity matrix is denoted as I, while the $m \times m$ all-one matrix is denoted as J. The $N \times N$ diagonal matrix $\text{diag}(x)$ contains the $N \times 1$ vector x on its main diagonal. The $N \times 1$ basic vector e_i contains only one non-zero element $(e_i)_i = 1$. The Hadamard product of two matrices with the same dimensions is denoted as $\circ$ and defines the element-wise product of the two matrices.

A graph $G(\mathcal{N}, \mathcal{L})$ consists of a set $\mathcal{N}$ of $N = |\mathcal{N}|$ nodes and a set $\mathcal{L}$ of $L = |\mathcal{L}|$ links and is defined by the $N \times N$ adjacency matrix A, where $a_{ij} = 1$ if node i and node j are connected by a link, otherwise $a_{ij} = 0$. The $N \times 1$ degree vector d obeys $A \cdot u$, while the corresponding $N \times N$ degree diagonal matrix is denoted by $\Delta = \text{diag}(d)$.

1.4. DOCUMENT STRUCTURE

This thesis consists of three parts, further divided into several chapters.

I. Topology and Spectrum of Graphs The first part consists of three chapters and deals with networks' topological and spectral properties. In Chapter 2, we demonstrate that a simple non-empty graph can be recovered, not necessarily uniquely, given the orthogonal eigenvectors of its adjacency matrix. As a consequence of this theorem, we introduce co-eigenvector graphs, which share the same eigenvectors but possess different eigenvalues. Co-eigenvector graphs form a duality with co-spectral graphs that share the same eigenvalues but possess different eigenvectors. We also provide an analysis of the properties of co-eigenvector graphs. Chapter 3 provides three different analytical solutions for the number of paths of a certain length between node pairs in a matrix form based on different types of walks. In addition, we propose an iterative algorithm that enumerates all possible paths between node pairs. A range of insights into how network topology impacts processes on the network motivates the inverse problem formulation; given the

1

desired performance or topological properties, how can we infer a network that satisfies given constraints? In Chapter 4, we utilise information captured by effective resistance while solving inverse graph problems. First, we consider graph sparsification, an iterative procedure or removing links from a graph, such that the spectrum of the reduced graph is as close as possible to the spectrum of the original graph. We propose a deterministic graph sparsification algorithm based on effective graph resistance, which removes links from a graph by either minimising or maximising the effective graph resistance of the graph. Another application of removing links from the graph is while solving the inverse shortest path problem, where the aim is to reconstruct a graph that satisfies given upper bounds on the shortest path weights between node pairs. Chapter 4 proposes an iterative algorithm for the inverse shortest path problem, utilising the information captured by the effective resistance between node pairs. The algorithm begins with the complete weighted graph, iteratively removes links, and redistributes link weights as long as given bounds are met. When the upper bounds for the shortest path weights are generated from a sparse graph, the proposed algorithm achieves phenomenal results by recovering the same weighted graph most of the time.

II. Linear Processes on Networks The second part of this thesis consists of three chapters, which explore dynamic processes on networks at different aggregation scales. Chapter 5 presents a linear clustering process on a network that governs nodal positions on a one-dimensional line. This is achieved through attraction and repulsion forces between adjacent nodes, with intensity proportional to the number of common and different neighbours, respectively. We estimate the number of clusters and each node's cluster membership based on their position. Chapter 6 analyses collected macroscopic measurements of population and area per Dutch municipality in the period 1830 – 2019. We apply Network Science and reconstruct the Dutch Municipality Network in each year of the researched period, where two municipalities are connected if they share a common border. From the evolution of the population and area distribution over time, we infer the impact of the underlying governing processes (such as the municipality merging process, population increase and people migration process) on the Dutch Municipality Network. In addition, we propose a model of the Dutch Municipality Network, composed of linear processes on the network, that achieves phenomenal prediction accuracy on a province level. In Chapter 7, we present a general solution for aggregating the dynamics of networked linear systems without losing any information about the dynamics of individual systems. Therefore, when individual systems perform linear dynamics and their interactions are linear, we propose an analytical solution for the dynamics of the networked system at different aggregation scales. Our solution allows for hierarchical structuring of networked linear systems.

I

Topological and Spectral Properties of Graphs

2

CO-EIGENVECTOR GRAPHS

If we knew what it was we were doing,
it would not be called research, would it?

Albert Einstein

Except for the empty graph, we show that the orthogonal matrix X of the adjacency matrix A determines that adjacency matrix completely, but not always uniquely. The proof relies on interesting properties of the Hadamard product $\Xi = X \circ X$. As a consequence of the theory, we show that irregular co-eigenvector graphs exist only if the number of nodes $N \geq 6$. Co-eigenvector graphs possess the same orthogonal eigenvector matrix X, but different eigenvalues of the adjacency matrix. Co-eigenvector graphs are the dual of co-spectral graphs, that share all eigenvalues of the adjacency matrix, but possess a different orthogonal eigenvector matrix. We deduce general properties of co-eigenvector graph and start to enumerate all co-eigenvector graphs on $N = 6$ and $N = 7$ nodes. Finally, we list many open problems.

This chapter is based on [50].

2

2.1. INTRODUCTION

A graph $G(\mathcal{N},\mathcal{L})$ is composed of a set $\mathcal{N}$ of $N = |\mathcal{N}|$ nodes and a set $\mathcal{L}$ of $L = |\mathcal{L}|$ links. An undirected and unweighted graph with N nodes can be represented by a $N \times N$ symmetric adjacency matrix A. The element a_{ij} of the adjacency matrix A equals $a_{ij} = 1$ if there exists a link between node i and j, else $a_{ij} = 0$. We exclude self-loops, implying that A has zero diagonal elements, i.e. $a_{jj} = 0$ for $1 \le j \le N$. We call a graph simple if it is undirected without self-loops. Just as any symmetric matrix, the symmetric, zero-one adjacency matrix A possesses the eigenvalue decomposition

$$A = X\Lambda X^T \tag{2.1}$$

as reviewed in the introduction of [51] and in Section 2.2. The equality in (2.1) implies that all information at the left-hand side, that we call the *topology domain*, is also contained in the right-hand side, that we call the *spectral domain*. Most insight so far in graphs is gained in the topology domain that allows a straightforward drawing of a graph: nodes are interconnected by links and the picture of a graph is attractive and understandable to humans. The spectral domain, consisting of the set of orthogonal and normalized eigenvectors $x_1, x_2, \ldots, x_N$ stored as columns in the orthogonal eigenvector matrix X in (2.1) and the corresponding set of eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_N$ stored in the eigenvalue vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_N)$ in $\Lambda = \text{diag}(\lambda)$, is less intuitive for humans; the meaning of an eigenvector and eigenvalue of a graph is not obvious. However, as mentioned in the preface of [51], the relation $A = X\Lambda X^T$ represents a transformation of a similar nature as a Fourier transform, which suggests that some information is better or more adequately accessible in one domain and other information in the other domain. Besides the topology domain and the spectral domain, there exists a third equivalent representation, called the *geometric domain*, where each, possibly weighted, undirected graph is a simplex in the $N-1$ dimensional Euclidean space [7].

Most of the spectral results are obtained for eigenvalues, in particular, for the largest eigenvalue or spectral radius [52]. While the number of mathematical results on other eigenvalues is already considerably less than for the spectral radius, results on eigenvectors are scarce [53, 54].

Earlier, Haemers and Van Dam [55] have conjectured that, when the number of nodes $N \to \infty$, the eigenvalue vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_N)$ characterizes the graph almost surely, i.e. the probability that eigenvalue vector λ determines the graph tends to 1. The Haemers and Van Dam conjecture practically means that the eigenvalue vector λ is a fingerprint of a real-world, large graph, that is comparable to a photoluminescence spectrum of a material (see e.g. [56]). Here, we present a kind of dual of the Haemers and Van Dam conjecture and concentrate on the orthogonal eigenvector matrix X in (2.1) rather than on the eigenvalue vector λ. In particular, in Appendix B.2, we will prove

Theorem 1 *The orthogonal eigenvector matrix X of the adjacency matrix A of an undirected, simple graph completely specifies that graph, except for the empty graph.*

Theorem 1 should be understood as "Given the orthogonal eigenvector matrix X of the adjacency matrix A of an undirected, simple (i.e. without self-loops) graph, then that adjacency matrix A can be retrieved". Since the empty graph trivially possesses any orthogonal X matrix with eigenvalue vector $\lambda = 0$, we exclude this extreme case. "Completely" means that the precise adjacency matrix is recovered, in contrast to a partial or approximated one as in network inference methods that estimate the most likely underlying graph. Section 2.4 discusses consequences of Theorem 1: we will show that co-eigenvector graphs exist and that the orthogonal eigenvector matrix X does not always "uniquely" specifies a graph, because different graphs can possess the same orthogonal eigenvector matrix X.

Before turning to the proof of Theorem 1 in Appendix B.2 and its consequences in Section 2.4, we briefly review the orthogonal eigenvector matrix X of a symmetric matrix in Section 2.2, introduce the Hadamard product $\Xi = X \circ X$ and derive some properties of the matrix Ξ in Appendix B.1, which we apply to the adjacency matrix of an undirected graph in Section 2.3. Section 2.5 deduces general properties of co-eigenvector graphs, for both regular and irregular graphs. Section 2.6 enumerates nearly all co-eigenvector graphs on $N = 6$ and $N = 7$ nodes. For $N < 6$, our enumeration algorithm did not find irregular co-eigenvector graphs. Proceeding with a higher number N of nodes rapidly becomes computationally challenging due to the huge increase in the number of unlabeled graph on N nodes. Section 2.7 concludes and poses many open problems.

2.2. EIGENVECTORS AND EIGENVALUES: BRIEF REVIEW

Following the notation of [51], we denote by x_k the $N \times 1$ eigenvector of the symmetric matrix A belonging to the eigenvalue λ_k, normalized so that $x_k^T x_k = 1$. In this Section 2.2, A is any symmetric matrix and not necessarily equal to the adjacency matrix. The eigenvalues of an $N \times N$ symmetric matrix $A = A^T$ are real and can be ordered as $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$. Let X be the orthogonal matrix with eigenvectors of A in the columns,

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_N \end{bmatrix}$$

or explicitly in terms of the m-th component $\left(x_j\right)_m$ of eigenvector x_j,

$$X = \begin{bmatrix} (x_1)_1 & (x_2)_1 & (x_3)_1 & \cdots & (x_N)_1 \\ (x_1)_2 & (x_2)_2 & (x_3)_2 & \cdots & (x_N)_2 \\ (x_1)_3 & (x_2)_3 & (x_3)_3 & \cdots & (x_N)_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (x_1)_N & (x_2)_N & (x_3)_N & \cdots & (x_N)_N \end{bmatrix} \tag{2.2}$$

where the element $X_{ij} = \left(x_j\right)_i$.

The relation $X^T X = I = XX^T$ (see e.g. [51, p. 223]) expresses, in fact, *double orthogonality*. The first equality $X^T X = I$ translates, with the Kronecker delta $\delta_{km} = 0$ if $k \neq m$, otherwise $\delta_{km} = \delta_{mm} = 1$, to the well-known orthogonality relation

$$x_k^T x_m = \sum_{j=1}^{N} (x_k)_j (x_m)_j = \delta_{km} \tag{2.3}$$

stating that the eigenvector x_k belonging to eigenvalue λ_k is orthogonal to any other eigenvector belonging to a different eigenvalue. The second equality $XX^T = I$, which

2

arises from the commutativity of the inverse matrix $X^{-1} = X^T$ with the matrix X itself, can be written as $\sum_{j=1}^{N} \left(x_j\right)_m \left(x_j\right)_k = \delta_{mk}$ and suggests us to define the row vector in X as $y_m = \left((x_1)_m, (x_2)_m, \ldots, (x_N)_m\right)$. Then, the second orthogonality condition $XX^T = I$ implies orthogonality of the vectors

$$y_l^T y_j = \sum_{k=1}^{N} (x_k)_l (x_k)_j = \delta_{lj} \tag{2.4}$$

The eigenvalue equation $Ax_k = \lambda_k x_k$ is written in matrix form for all eigenvectors as $AX = X\Lambda$. After right-multiplying both sides in $AX = X\Lambda$ by X^T and invoking the orthogonality relation $XX^T = I$, we obtain the matrix equation $A = X\Lambda X^T$ in (2.1), where $\Lambda = \text{diag}(\lambda)$ is an $N \times N$ diagonal matrix with $\Lambda_{kk} = \lambda_k$.

The $N \times N$ matrix $\Xi = X \circ X$, where $\circ$ denotes the Hadamard product[1],

$$\Xi = \begin{bmatrix} (x_1)_1^2 & (x_2)_1^2 & (x_3)_1^2 & \cdots & (x_N)_1^2 \\ (x_1)_2^2 & (x_2)_2^2 & (x_3)_2^2 & \cdots & (x_N)_2^2 \\ (x_1)_3^2 & (x_2)_3^2 & (x_3)_3^2 & \cdots & (x_N)_3^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (x_1)_N^2 & (x_2)_N^2 & (x_3)_N^2 & \cdots & (x_N)_N^2 \end{bmatrix} \tag{2.5}$$

will play an important role in this chapter.

2.3. THE MATRIX $\Xi = X \circ X$ OF THE ADJACENCY MATRIX

When applying the general theory in Appendix B.1 to the adjacency matrix A, formula (B.4) for integer powers $f(z) = z^k$ leads to nice formulae. Indeed, for $k = 0$, we find from (B.1) the *second* orthogonality relation (2.4); for $k = 1$ (since $a_{jj} = 0$, from which $\text{trace}(A) = \sum_{j=1}^{N} \lambda_j = 0$)

$$0 = \sum_{k=1}^{N} \lambda_k (x_k)_j^2 \text{ and } 0 = \Xi\lambda \tag{2.6}$$

that appeared earlier in [51, p. 229], while for $k = 2$ (since the degree of node j is $d_j = \left(A^2\right)_{jj}$)

$$d_j = \sum_{k=1}^{N} \lambda_k^2 (x_k)_j^2 \text{ and } d = \Xi\lambda^2 \tag{2.7}$$

For any adjacency matrix A without self-loops (i.e. $a_{jj} = 0$ for each $1 \le j \le N$), the instance (2.6)

$$\Xi\lambda = 0 \tag{2.8}$$

is the special case of the eigenvalue equation (B.10), where the eigenvalue vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_N)$ of the adjacency matrix A is the eigenvector of Ξ corresponding to eigenvalue zero. Lemma 22 states that $\lambda^T u = 0$ or $\sum_{j=1}^{N} \lambda_j = 0$. In addition, (B.10) implies that $\det(\Xi) = 0$, which is equivalent to the fact that $\text{rank}(\Xi) \le N - 1$. Thus, the rank of the matrix Ξ for an adjacency matrix is at most $N - 1$.

[1]The Hadamard product [57] (entrywise product) of two matrices is $(A \circ B)_{ij} = A_{ij} B_{ij}$. If A and B are both diagonal matrices, then $A.B = A \circ B$.

The general relation (B.7) simplifies for the adjacency matrix A to

$$\begin{bmatrix} 1 & 0 & d_1 & \cdots & (A^k)_{11} & \cdots & (A^{N-1})_{11} \\ 1 & 0 & d_2 & \cdots & (A^k)_{22} & \cdots & (A^{N-1})_{22} \\ 1 & 0 & d_3 & \cdots & (A^k)_{33} & \cdots & (A^{N-1})_{33} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & 0 & d_N & \cdots & (A^k)_{NN} & \cdots & (A^{N-1})_{NN} \end{bmatrix} = \Xi . \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^k & \cdots & \lambda_1^{N-1} \\ 1 & \lambda_2 & \lambda_2^2 & \cdots & \lambda_2^k & \cdots & \lambda_2^{N-1} \\ 1 & \lambda_3 & \lambda_3^2 & \cdots & \lambda_3^k & \cdots & \lambda_3^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 1 & \lambda_N & \lambda_N^2 & \cdots & \lambda_N^k & \cdots & \lambda_N^{N-1} \end{bmatrix} \quad (2.9)$$

2

where $\left(A^k\right)_{jj}$ equals the number of closed walks of length k from node j and back to node j.

2.3.1. EXAMPLES OF PARTICULAR GRAPHS

(a) In a line topology or path on N nodes, only even closed walks are possible and $\left(A^k\right)_{jj} = 0$ for odd k. For finite N and even k, symmetry is broken and $\left(A^k\right)_{jj} \neq \left(A^k\right)_{ll}$ for any pair (l, j) of nodes, due to the end nodes. Since all eigenvalues of the adjacency matrix of a path graph are distinct [51, p. 124], we deduce from (2.9) and (B.9) that $\text{rank}(\Xi_{\text{path}}) = \left[\frac{N}{2}\right]$, where $[x]$ is the integer part of the real number x. The same result, $\text{rank}(\Xi_{\text{path}}) = \left[\frac{N}{2}\right]$, can also be obtained from the explicit analytic expression (e.g. [51, p. 124]) for the orthogonal eigenvector matrix X_{path}.

(b) For a regular graph with degree r, the degree vector is $d = r.u$ and the first and third column in the non-negative matrix Y in (B.8) are dependent. Hence, rank(Y) is at most $N-2$ for regular graphs, but rank(Ξ) can still be $N-1$ as shown in (c) below.

(c) The adjacency matrix of the complete graph $A_{K_N} = J - I$, where $J = u.u^T$ is the all-one matrix. For the complete graph K_N, the matrix Y – the left-hand side matrix in (2.9) – can be computed analytically, because $\left(A^k_{K_N}\right)_{jj} = (J-I)^k_{jj} = \frac{1}{N}\left((N-1)^k - (-1)^k\right) + (-1)^k$, which is the same for any node j, as

$$Y_{K_N} = \left[\begin{array}{ccccccc} u & 0 & (N-1)\,u & \cdots & \left(\frac{(N-1)^k-(-1)^k}{N} + (-1)^k\right) u & \cdots & \left((J-I)^{N-1}_{jj}\right) u \end{array}\right]$$

Since all columns are multiples of the all-one vector u, we find that $\text{rank}(Y_{K_N}) = 1$. The adjacency matrix $A_{K_N} = J - I$ of the complete graph K_N has two eigenvalues: $N-1$ belonging to eigenvector $x_1 = u$ and -1 with multiplicity $N-1$. Hence, the rank of the Vandermonde matrix V in (B.7) is rank$(V) = 2$ and (B.7) is not effective to determine rank(Ξ). Fortunately, the orthogonal eigenvector matrix of adjacency matrix $A_{K_N} = J - I$ can be computed analytically, in at least two ways.

The eigenvalue equation for $\lambda = -1$ is $(J-I)\,x = -x$, which is equivalent to $0 = Jx = u.u^T x$. Hence, any set of $N-1$ independent vectors $\{x_2, x_3, \ldots, x_N\}$ with a component sum equal to zero is possible. In other words, there are infinitely many orthogonal X-matrices for the complete graph K_N. Perhaps, the simplest *not normalized* eigenvector for the complete graph K_N is

$$\widetilde{x}_j = e_j - \frac{1}{j-1}\sum_{m=1}^{j-1} e_m \qquad \text{for } j > 1$$

where e_j is the basic vector with component $\left(e_j\right)_k = \delta_{jk}$. The eigenvector $\widetilde{x}_j$ satisfies the eigenvalue equation $(J-I)\,\widetilde{x}_j = -\widetilde{x}_j$ or $J\widetilde{x}_j = 0$, because $Je_j = u$. In addition, using

$e_m^T e_k = \delta_{mk}$, the scalar product $\widetilde{x}_j^T \widetilde{x}_k = \delta_{jk}$ is

$$\begin{aligned}
\widetilde{x}_j^T \widetilde{x}_k &= \left(e_j^T - \frac{1}{j-1}\sum_{m=1}^{j-1} e_m^T\right)\left(e_k - \frac{1}{k-1}\sum_{l=1}^{k-1} e_l\right) \\
&= e_j^T e_k - \frac{1}{k-1}\sum_{l=1}^{k-1} e_j^T e_l - \frac{1}{j-1}\sum_{m=1}^{j-1} e_m^T e_k + \frac{1}{j-1}\frac{1}{k-1}\sum_{m=1}^{j-1}\sum_{l=1}^{k-1} e_m^T e_l \\
&= \delta_{jk} - \frac{1}{k-1}\sum_{l=1}^{k-1}\delta_{jl} - \frac{1}{j-1}\sum_{m=1}^{j-1}\delta_{mk} + \frac{1}{j-1}\frac{1}{k-1}\sum_{m=1}^{j-1}\sum_{l=1}^{k-1}\delta_{ml} \\
&= \delta_{jk} - \frac{1}{k-1} 1_{\{j\in[1,k-1]\}} - \frac{1}{j-1} 1_{\{k\in[1,j-1]\}} + \frac{1}{j-1}\frac{1}{k-1}\sum_{m=1}^{j-1} 1_{\{m\in[1,k-1]\}}
\end{aligned}$$

If $j = k$, then

$$\widetilde{x}_k^T \widetilde{x}_k = 1 + \frac{1}{(k-1)^2}\sum_{m=1}^{k-1} 1_{\{m\in[1,k-1]\}} = 1 + \frac{1}{k-1} = \frac{k}{k-1}$$

Without loss of generality, we may assume that $j < k$ (else interchange j and k) and then, with $\sum_{m=1}^{j-1} 1_{\{m\in[1,k-1]\}} = j-1$, we find

$$\widetilde{x}_j^T \widetilde{x}_k = -\frac{1}{k-1} + \frac{1}{j-1}\frac{1}{k-1}\sum_{m=1}^{j-1} 1_{\{m\in[1,k-1]\}} = 0$$

Hence, the normalized eigenvector $x_j = \frac{\widetilde{x}_j}{\sqrt{\widetilde{x}_j^T \widetilde{x}_j}} = \sqrt{\frac{j-1}{j}}\, e_j - \frac{1}{\sqrt{j(j-1)}}\sum_{m=1}^{j-1} e_m$ and the corresponding orthogonal eigenvector matrix for the complete graph K_N is

$$X_{K_N} = \begin{bmatrix}
\frac{1}{\sqrt{N}} & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}} & -\frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{5}} & \cdots & -\frac{1}{\sqrt{N(N-1)}} \\
\frac{1}{\sqrt{N}} & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{6}} & -\frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{5}} & \cdots & -\frac{1}{\sqrt{N(N-1)}} \\
\frac{1}{\sqrt{N}} & 0 & \sqrt{\frac{2}{3}} & -\frac{1}{2\sqrt{3}} & -\frac{1}{2\sqrt{5}} & \cdots & -\frac{1}{\sqrt{N(N-1)}} \\
\frac{1}{\sqrt{N}} & 0 & 0 & \frac{\sqrt{3}}{2} & -\frac{1}{2\sqrt{5}} & \cdots & -\frac{1}{\sqrt{N(N-1)}} \\
\frac{1}{\sqrt{N}} & 0 & 0 & 0 & \sqrt{\frac{5}{6}} & \cdots & -\frac{1}{\sqrt{N(N-1)}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\frac{1}{\sqrt{N}} & 0 & 0 & 0 & 0 & \cdots & \frac{N-1}{\sqrt{N}}
\end{bmatrix} \tag{2.10}$$

and the rank of the corresponding matrix $\Xi_{K_N} = X_{K_N} \circ X_{K_N}$ is $\text{rank}(\Xi_{K_N}) = N-1$. Barik *et al.* [58] have shown that only regular graphs, such as the complete graph K_N, for $N = 4k$ and $k \in \mathbb{N}_0$, and the regular bipartite graph $K_{2k,2k}$, are diagonalizable by a Hadamard matrix. An $n \times n$ Hadamard matrix H_n has as elements either -1 and 1 and obeys $H_n H_n^T = nI_n$, where the order n can only be $n = 1, 2$ or $n = 4k$, subject to the fact that Hadamard's conjecture, namely that there exist a Hadamard matrix H_{4k} for each integer k, holds. Hadamard's conjecture is still an open, unsolved problem. The normalized matrix $X_n = \frac{1}{\sqrt{n}} H_n$ is an orthogonal matrix, from which it follows that $\det H_n = n^{\frac{n}{2}}$, which is maximal among all $n \times n$ matrices with elements in absolute value less than or equal to 1 and the latter class includes all orthogonal matrices. Any relabeling (permutation of rows and columns) of a Hadamard matrix is again a Hadamard matrix; multiplying any row or column by −1 preserves the Hadamard properties. Following Barik *et al.* [58], let $H_n = \left[u|\widetilde{H}\right]$ so that $H_n e_1 = u$. Consider the diagonal matrix $D = I - e_1 e_1^T$, then

$$H_n D H_n^T = H_n H_n^T - H_n e_1 (H_n e_1)^T = nI_n - u.u^T = nI - J$$

Hence, the Laplacian matrix of the complete graph K_n is $Q_{K_n} = nI - J = H_n D H_n^T$. Since K_n is a regular graph, the eigenvectors of the Laplacian Q and the adjacency matrix A are the same[2]. In conclusion, any Hadamard matrix with $H_n e_1 = u$ provides the orthogonal matrix for the complete graph K_n. Since $H_n \circ H_n = J = u.u^T$, we find that the corresponding $\text{rank}\left(\Xi_{K_n}\right) = 1$, which is the minimum possible rank for any Ξ matrix.

In summary, depending on the choice of the orthogonal eigenvector matrix for the complete graph K_N for $N = 4k$, we believe that the rank of the corresponding Ξ matrix may vary over all possible values: $1 \leq \text{rank}\left(\Xi_{K_N}\right) \leq N-1$. However, we do not have a proof that $\text{rank}\left(\Xi_{K_N}\right)$ can attain any integer in the interval $[1, N]$.

2.4. CONSEQUENCES OF THEOREM 1

Our main Theorem 1 is proved in Appendix B.2. Here, we discuss the consequences.

If $\text{rank}(\Xi) < N-1$, then the proof of Theorem 1 shows that the orthogonal eigenvector matrix X may specify more than one undirected graph. Such graphs are called "co-eigenvector graphs" and possess a same orthogonal eigenvector matrix X, but a different eigenvalue vector λ, as opposed to co-spectral graphs that have a same eigenvalue vector λ, but a different orthogonal eigenvector matrix X. Only if $\text{rank}(\Xi) = N-1$, the eigenvalue equation $\Xi\lambda = 0$ in (2.8) possesses one eigenvalue vector λ and we find immediately from Theorem 1

Corollary 1 *The orthogonal eigenvector matrix X of the adjacency matrix A of an undirected graph only specifies the graph uniquely if* $\text{rank}(\Xi) = N-1$.

The proof of Theorem 1 fundamentally relies on the zero-one matrix structure when $\text{rank}(\Xi) < N-1$ to recover the adjacency matrix A from the orthogonal eigenvector matrix X and thus excludes an extension towards weighted graphs. However, if $\text{rank}(\Xi) = N-1$, then also a weighted adjacency matrix, apart from a scaling factor β, can be recovered.

Fig. 2.1 shows the metacode of a graph recovery algorithm, based on the proof of Theorem 1 in Appendix B.2.

If $\text{rank}(\Xi) = N-n$ with $n > 1$ and if $n = O\left(N^\gamma\right)$ for large N and $0 < \gamma \leq 1$, meaning that the dimension $n \simeq \alpha N^\gamma$ of the kernel space increases with the number N of nodes, then the proof of Theorem 1 and the corresponding metacode in Fig. 2.1 looses computational efficiency, because $2^n \sim 2^{\alpha N^\gamma}$ (in the loop in line 6 in Fig. 2.1) increases non-polynomially fast with size N of the graph, pointing towards (but not proving) the NP-hard nature of the graph recovery problem in the worst case. In the worst case of

[2]Indeed, for a regular graph with degree r, the Laplacian is $Q = rI - A$. If $Q = ZMZ^T$ and $A = X\Lambda X^T$, we observe that $ZMZ^T = X(rI - \Lambda)X^T$, implying that $X = Z$.

2

GRAPH RECOVERY

input: orthogonal matrix X with N orthonormal eigenvectors of A
output: adjacency matrix A

1. $\Xi \leftarrow X \circ X$ Hadamard product
2. $n \leftarrow$ size of the kernel space of Ξ
3. v_i with $i \in \{1, 2, \cdots, n\} \leftarrow$ eigenvectors of Ξ obeying $\Xi \cdot v_i = 0$
4. $C_{(\{1,2,\ldots,N^2\},i)} \leftarrow \text{vec}(X \cdot \text{diag}(v_i) \cdot X^T)$ for $i \in \{1, 2, \cdots, n\}$
5. $M_{n\times n} \leftarrow n$ non-zero rows j of C, where $j \neq k(N+1)+1$, $k \in \{0, 1, \ldots, N-1\}$ such that $\text{rank}(M) = n$

6. **For** $(j \leftarrow 1$ to $2^n - 1)$ **do**
7. $\quad \hat{a}_{n\times 1} \leftarrow$ binary representation in n digits of j
8. $\quad \hat{\beta}_{n\times 1} \leftarrow M^{-1} \cdot \hat{a}$
9. $\quad \hat{\lambda}_{N\times 1} \leftarrow \sum_{i=1}^{n} \hat{\beta}_i \cdot v_i$
10. $\quad \hat{A}_{N\times N} \leftarrow X \cdot \text{diag}(\hat{\lambda}) \cdot X^T$
11. $\quad$ **If** $(\hat{A}$ contains only ones and zeros)
12. $\quad\quad$ return $\hat{\lambda}$, $\hat{A}$
13. $\quad$ **End If**
14. **End For**

Figure 2.1: Metacode of the algorithm for graph recovery, given the orthogonal eigenvector matrix X

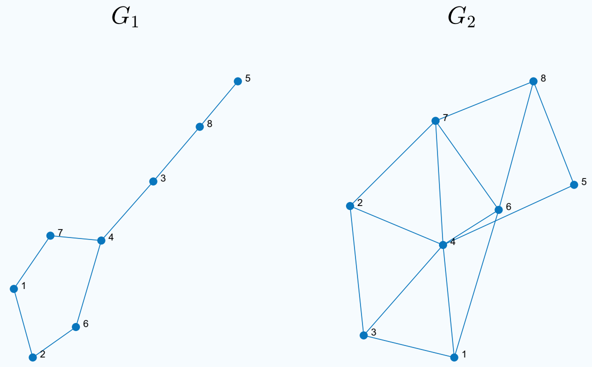
low rank(Ξ) or high n, one might argue that the proof of Theorem 1 is hardly better than the trivial method of finding the eigenvector λ by inversion of (2.1), i.e. finding the adjacency matrix A that diagonalizes $X^T AX = \Lambda$, by checking all $2^{\binom{N}{2}}$ possible $N \times N$ adjacency matrices. Since X is the orthogonal eigenvector matrix of "a particular" adjacency matrix, we certainly known that at least one of all possible $N \times N$ adjacency matrices converts $X^T AX$ to a diagonal matrix Λ. However, extensive simulations so far indicate that rank(Ξ) $< N - 1$ occurs for relatively small graphs and is extremely rare for large N. In other words, for large graphs, nearly always rank(Ξ) $= N - 1$ holds, so that Corollary 1 applies.

Fig. 2.2 and 2.3 exemplify the existence of co-eigenvector graphs.

When $X = \frac{1}{\sqrt{n}} H_n$ is given for $n = 8$, then rank(Ξ) $= 1$ as shown in Section 2.3.1 and the algorithm in Fig. 2.1 finds $2^{n-1} = 128$ labeled co-eigenvector graphs, that are all regular graphs with integer eigenvalues. Indeed, any regular graph has all eigenvectors, except for the principal eigenvector $x_1 = u$, orthogonal to the all-one vector u and thus shares a common basis of eigenvectors with the complete graph. Regular graphs are further examined in Section 2.5.1.

Fig. 2.4 presents some co-eigenvector graphs of the line topology.

$$X = \begin{bmatrix} -0.1229 & -0.6015 & -0.1348 & -0.2361 & 0.3717 & 0.4237 & 0.3342 & 0.3461 \\ -0.1229 & 0.6015 & -0.1348 & -0.2361 & -0.3717 & 0.4237 & 0.3342 & 0.3461 \\ 0.4625 & 0 & -0.2935 & -0.5802 & 0 & -0.1881 & -0.4653 & 0.3338 \\ -0.6034 & 0 & -0.2508 & 0.1914 & 0 & -0.4989 & -0.1008 & 0.5266 \\ 0.1581 & 0 & -0.4919 & 0.5960 & 0 & 0.4606 & -0.3969 & 0.0906 \\ 0.3665 & -0.3717 & 0.3052 & 0.2745 & -0.6015 & -0.0978 & 0.1584 & 0.4031 \\ 0.3665 & 0.3717 & 0.3052 & 0.2745 & 0.6015 & -0.0978 & 0.1584 & 0.4031 \\ -0.3132 & 0 & 0.6216 & -0.0970 & 0 & 0.3542 & -0.5850 & 0.1960 \end{bmatrix}$$



$$\lambda_1 = \begin{bmatrix} -1.9816 & -1.6180 & -1.2635 & -0.1627 & 0.6180 & 0.7691 & 1.4739 & 2.1648 \end{bmatrix}$$

$$\lambda_2 = \begin{bmatrix} -1.8360 & 0.6180 & 1.7733 & 0.4839 & -1.6180 & -1.8524 & -1.2199 & 3.6511 \end{bmatrix}$$

Figure 2.2: Example of two co-eigenvector graphs

2.5. PROPERTIES OF CO-EIGENVECTOR GRAPHS

Appendix B.2 has demonstrated that co-eigenvector graphs can exist, provided that $\text{rank}(\Xi) < N-1$. In this Section 2.5, we deduce some properties of two co-eigenvector graphs $G_1(\mathcal{N},\mathcal{L}_1)$ and $G_2(\mathcal{N},\mathcal{L}_2)$ on N nodes, that possess the same eigenvectors, but a different set of eigenvalues:

$$\begin{cases} A_1 = X\Lambda_1 X^T = \sum_{i=1}^{N} \lambda_i(A_1)\, x_i x_i^T \\ A_2 = X\Lambda_2 X^T = \sum_{i=1}^{N} \lambda_i(A_2)\, x_i x_i^T \end{cases} \tag{2.11}$$

where the $N \times N$ diagonal matrices Λ_1 and Λ_2 contain on the main diagonal the eigenvalues of the adjacency matrices A_1 and A_2, respectively.

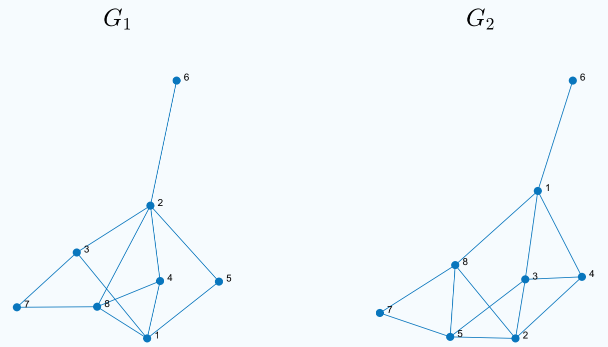
First, the sum of the adjacency matrices A_1 and A_2

$$A_1 + A_2 = \sum_{i=1}^{N} (\lambda_i(A_1) + \lambda_i(A_2))\, x_i x_i^T \tag{2.12}$$

again represents an adjacency matrix, provided that the graphs G_1 and G_2 do not share common links (i.e. $|\mathcal{L}_1 \cap \mathcal{L}_2| = 0$). In Theorem 3 below, we derive the number of common links between two co-eigenvector graphs explicitly. Second, the product of the adjacency

2

$$X = \begin{bmatrix} -0.4466 & -0.0993 & 0.4895 & 0.2580 & -0.3541 & 0.4121 & 0.0986 & -0.4238 \\ -0.5064 & -0.1988 & -0.4582 & -0.0438 & 0.0923 & -0.3272 & -0.4012 & -0.4643 \\ 0.4177 & -0.1223 & -0.1963 & 0.4850 & -0.3963 & -0.4053 & 0.3262 & -0.3303 \\ 0.2086 & 0.5953 & -0.2222 & 0.3117 & 0.3773 & 0.3824 & -0.1069 & -0.3945 \\ 0.3275 & 0.2108 & -0.0450 & -0.5620 & -0.5754 & 0.1277 & -0.3378 & -0.2624 \\ 0.1740 & 0.1406 & 0.6590 & 0.1150 & 0.2030 & -0.4918 & -0.4478 & -0.1372 \\ -0.2624 & 0.4710 & 0.1052 & -0.3990 & 0.0815 & -0.3546 & 0.5950 & -0.2297 \\ 0.3460 & -0.5434 & 0.1232 & -0.3329 & 0.4333 & 0.1695 & 0.2069 & -0.4469 \end{bmatrix}$$



$$\lambda_1 = \begin{bmatrix} -2.9101 & -1.4136 & -0.6953 & -0.3811 & 0.4548 & 0.6653 & 0.8960 & 3.3841 \end{bmatrix}$$

$$\lambda_2 = \begin{bmatrix} -2.5665 & -0.7062 & 0.7428 & 2.2432 & -1.7438 & -0.8379 & -0.2201 & 3.0886 \end{bmatrix}$$

Figure 2.3: Example of two other co-eigenvector graphs

matrices A_1 and A_2

$$A_1 \cdot A_2 = \sum_{i=1}^{N} \lambda_i(A_1)\lambda_i(A_2)x_i x_i^T, \tag{2.13}$$

contains the same set of eigenvectors as A_1 and A_2 due to orthogonality of the eigenvectors. Lemma [51, p. 253] indeed tells us that if any two matrices B and C have a common complete set of eigenvectors, then B and C commute. Relation (2.13) may be regarded as another demonstration of that Lemma. The diagonal element $(A_1 \cdot A_2)_{ii} = \sum_{k=1}^{N}(A_1)_{ik}(A_2)_{ik}$ equals the number of common neighbors of node i in G_1 and G_2, i.e. each node k for which $(A_1)_{ik} = (A_2)_{ik} = 1$.

The $N \times N$ Hadamard product $A_c = A_1 \circ A_2$ represents the adjacency matrix of the graph $G_c(\mathcal{N}_c, \mathcal{L}_c)$, composed of common links $\mathcal{L}_c = \mathcal{L}_1 \cap \mathcal{L}_2$ between G_1 and G_2,

$$A_c = \left(\sum_{i=1}^{N} \lambda_i(A_1)x_i x_i^T\right) \circ \left(\sum_{j=1}^{N} \lambda_j(A_2)x_j x_j^T\right). \tag{2.14}$$

Using the distributive property of a Hadamard product [57, p. 32], we transform (2.14) as

$$A_c = \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i(A_1)\lambda_j(A_2)\left(x_i x_i^T\right) \circ \left(x_j x_j^T\right).$$

$$X = \begin{bmatrix} -0.16123 & 0.30301 & 0.40825 & 0.46424 & 0.46424 & 0.40825 & -0.30301 & 0.16123 \\ 0.30301 & -0.46424 & -0.40825 & -0.16123 & 0.16123 & 0.40825 & -0.46424 & 0.30301 \\ -0.40825 & 0.40825 & 0 & -0.40825 & -0.40825 & 0 & -0.40825 & 0.40825 \\ 0.46424 & -0.16123 & 0.40825 & 0.30301 & -0.30301 & -0.40825 & -0.16123 & 0.46424 \\ -0.46424 & -0.16123 & -0.40825 & 0.30301 & 0.30301 & -0.40825 & 0.16123 & 0.46424 \\ 0.40825 & 0.40825 & 0 & -0.40825 & 0.40825 & 0 & 0.40825 & 0.40825 \\ -0.30301 & -0.46424 & 0.40825 & -0.16123 & -0.16123 & 0.40825 & 0.46424 & 0.30301 \\ 0.16123 & 0.30301 & -0.40825 & 0.46424 & -0.46424 & 0.40825 & 0.30301 & 0.16123 \end{bmatrix}$$

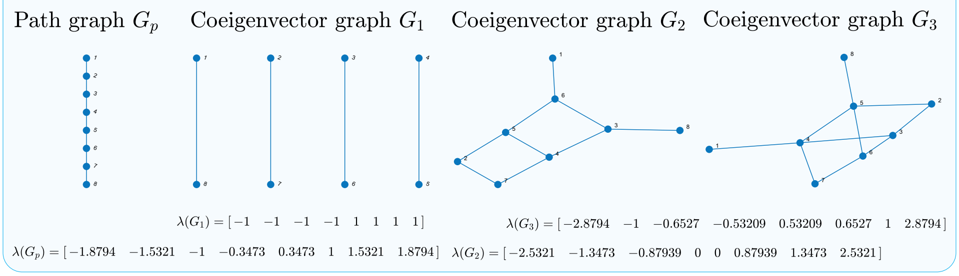


Figure 2.4: Example of co-eigenvector graphs of a line topology on $N = 8$ nodes.

The Hadamard product of outer products $x_i x_i^T$ and $x_j x_j^T$ is written [57] as

$$\left(x_i x_i^T\right) \circ \left(x_j x_j^T\right) = \text{diag}(x_i) x_j x_j^T \text{diag}(x_i) = \left(x_i \circ x_j\right) \left(x_i \circ x_j\right)^T$$

simplifying (2.14) further as

$$A_c = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i(A_1) \lambda_j(A_2) \left(x_i \circ x_j\right) \left(x_i \circ x_j\right)^T. \tag{2.15}$$

Definition 2 *Two co-eigenvector graphs G_1 and G_2 are called non-overlapping if they do not share common links.*

Another way to determine the number of common links between G_1 and G_2 is by summing the elements of the product $A_1 \cdot A_2$ on the main diagonal

$$2|\mathcal{L}_1 \cap \mathcal{L}_2| = \text{trace}\left(A_1 \cdot A_2\right). \tag{2.16}$$

Theorem 3 *Consider two co-eigenvector graphs $G_1(\mathcal{N}, \mathcal{L}_1)$ and $G_2(\mathcal{N}, \mathcal{L}_2)$ on N nodes, defined by the $N \times N$ adjacency matrices A_1 and A_2, respectively. Graphs G_1 and G_2 are non-overlapping if their eigenvalue vectors are orthogonal.*

Proof: Since the graph G_c, with the $N \times N$ adjacency matrix A_c defined in (2.15), is composed of common links between G_1 and G_2, twice the number of common links between the co-eigenvector graphs G_1 and G_2 equals the sum of elements of A_c

$$2|\mathcal{L}_1 \cap \mathcal{L}_2| = u^T \left(A_1 \circ A_2\right) u = u^T A_c u \tag{2.17}$$

where u denotes the all-one vector. By substituting (2.15) into (2.17) we obtain

$$2|\mathcal{L}_1 \cap \mathcal{L}_2| = \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i(A_1)\lambda_j(A_2) u^T \left(x_i \circ x_j\right)\left(x_i \circ x_j\right)^T u.$$

The inner product $\left(x_i \circ x_j\right)^T u = x_i^T x_j$ equals 1 if $i = j$, otherwise 0, because the eigenvectors of a symmetric adjacency matrix are orthogonal. Thus, the relation (2.17) simplifies to

$$2|\mathcal{L}_1 \cap \mathcal{L}_2| = (\lambda(A_1))^T \cdot \lambda(A_2). \tag{2.18}$$

Since "non-overlapping" in Definition 1 means that $|\mathcal{L}_1 \cap \mathcal{L}_2| = 0$, relation (2.18) completes the proof. □

Theorem 3 states that if two co-eigenvector graphs G_1 and G_2 do not share common links, their eigenvalue vectors $\lambda(A_1)$ and $\lambda(A_2)$ are orthogonal. The vectors $\lambda(A_1)$ and $\lambda(A_2)$ span the kernel space of the $N \times N$ matrix $\Xi = X \circ X$, as shown in the proof of Theorem 1 in Appendix B.2 provided that $\text{rank}(\Xi) = N - 2$. The sum of two non-overlapping co-eigenvector graphs A_1 and A_2 is another co-eigenvector graph $A_s = A_1 + A_2$, with the eigenvalue vector $\lambda(A_s) = \lambda(A_1) + \lambda(A_2)$, as derived in (2.12). Thus, the eigenvalue vector $\lambda(A_s)$ also lies in the kernel space of the matrix Ξ, and, hence, $\text{rank}(\Xi) \leq N - 2$.

The Hadamard product in (2.15) allows us to determine the number of non-common links in G_1 and G_2.

Corollary 2 *Consider a pair of co-eigenvector graph $G_1(\mathcal{N}, \mathcal{L}_1)$ and $G_2(\mathcal{N}, \mathcal{L}_2)$ on N nodes with corresponding adjacency matrices A_1 and A_2, respectively. The number of non-common links in G_1 and G_2 is given by*

$$|\mathcal{L}_1 \setminus \mathcal{L}_2| + |\mathcal{L}_2 \setminus \mathcal{L}_1| = \sum_{i=1}^{N} (\lambda_i(A_1) - \lambda_i(A_2))^2 \tag{2.19}$$

Proof: A graph $G_u\left(\mathcal{N}, (\mathcal{L}_1 \setminus \mathcal{L}_2) \cup (\mathcal{L}_2 \setminus \mathcal{L}_1)\right)$ contains only non-common links of G_1 and G_2 and has the corresponding $N \times N$ adjacency matrix $A_u = A_1 + A_2 - 2\left(A_1 \circ A_2\right)$. By using the identity $A \circ A = A$, that holds for any zero-one matrix, and importing (2.15), we obtain

$$\begin{aligned} A_u &= A_1 \circ A_1 + A_2 \circ A_2 - 2\left(A_1 \circ A_2\right) \qquad (2.20)\\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \left(\lambda_i(A_1)\lambda_j(A_1) + \lambda_i(A_2)\lambda_j(A_2) - 2\lambda_i(A_1)\lambda_j(A_2)\right)\left(x_i \circ x_j\right)\left(x_i \circ x_j\right)^T \end{aligned}$$

from which the number of not-common links in G_1 and G_2 is computed as the sum of elements of A_u

$$u^T \cdot A_u \cdot u = \sum_{i=1}^{N} \left(\lambda_i^2(A_1) + \lambda_i^2(A_2) - 2\lambda_i(A_1)\lambda_i(A_2)\right),$$

which completes the proof. □

An equivalent way to compute the number of not-common links in G_1 and G_2 is to distract twice the number of common links in G_1 and G_2 from the sum of elements of $A_1 + A_2$

$$|\mathscr{L}_1 \setminus \mathscr{L}_2| + |\mathscr{L}_2 \setminus \mathscr{L}_1| = u^T \cdot (A_1 + A_2) \cdot u - 2 \cdot \text{trace}(A_1 \cdot A_2), \tag{2.21}$$

which, after substituting (2.12) and (2.13) again leads to (2.19). The adjacency matrix A_u with only non-common links in G_1 and G_2 in (2.20), using the distributive property of the Hadamard product, can be transformed into

$$A_u = (A_1 - A_2) \circ (A_1 - A_2), \tag{2.22}$$

where relation (2.22) holds for adjacency matrices A_1 and A_2 of any two unweighted graphs G_1 and G_2.

2.5.1. REGULAR GRAPHS

In a regular graph G_r on N nodes, defined by the $N \times N$ adjacency matrix A_r, each node has the same degree r. The complement graph G_r^c of G_r is also a regular graph with degree $q = N - 1 - r$ and the $N \times N$ adjacency matrix [51, p. 13] is

$$A_r^c = J - I - A_r, \tag{2.23}$$

where the $N \times N$ all-one matrix is denoted by $J = u.u^T$. Since each node in G_r has degree r, it holds that $A_r \cdot u = d_r = r \cdot u$. Thus, the principal eigenvalue $\lambda_1(A_r) = r$ corresponds to the principal eigenvector $x_1 = \frac{1}{\sqrt{N}} u$. The remaining $N - 1$ eigenvectors of A_r are orthogonal to u, implying that $u^T x_j = 0$ or

$$\sum_{i=1}^{N} \left(x_j\right)_i = 0, \tag{2.24}$$

where $1 < j \le N$. The following theorem is also provided in [59, p.15].

Theorem 4 *A regular graph G_r on N nodes with degree r and its complement graph G_r^c compose a pair of co-eigenvector graphs.*

Proof: By multiplying the $N \times N$ adjacency matrix A_r^c of the complement graph G_r^c, defined in (2.23), with the eigenvector x_j of A_r, where $j > 1$, we obtain

$$A_r^c \cdot x_j = (J - I - A_r) \cdot x_j.$$

From (2.24) we conclude that $J \cdot x_j = u \cdot u^T \cdot x_j = 0$ and the above equation becomes

$$A_r^c \cdot x_j = \left(-1 - \lambda_j(A_r)\right) \cdot x_j. \tag{2.25}$$

Additionally, multiplying the adjacency matrix A_c with the principal eigenvector $\frac{1}{\sqrt{N}} u$ yields

$$A_r^c \cdot \frac{1}{\sqrt{N}} \cdot u = (J - I - A_r) \cdot \frac{1}{\sqrt{N}} \cdot u = (N - 1 - r_1) \cdot \frac{1}{\sqrt{N}} \cdot u$$

showing that the adjacency matrix A_r^c shares the same eigenvectors with A_r, which completes the proof. □

Relation (2.25) shows that the adjacency matrix A_r^c of the complement graph G_r^c of a regular graph possesses the spectral decomposition

$$A_r^c = \frac{N-1-r}{N} \cdot u \cdot u^T + \sum_{j=2}^{N} \left(-1-\lambda_j(A_r)\right) x_j x_j^T. \tag{2.26}$$

Theorem 3 states that the eigenvalue vectors $\lambda(A_r)$ and $\lambda(A_r^c)$ are orthogonal. Indeed, the inner product $(\lambda(A_r))^T \cdot \lambda(A_r^c)$ transforms, after using (2.26), into

$$\begin{aligned}(\lambda(A_r))^T \cdot \lambda(A_r^c) &= r \cdot (N-1-r) + \sum_{j=2}^{N} \lambda_j(A_r) \cdot \left(-1-\lambda_j(A_r)\right) \\ &= r \cdot N - \left(r + \sum_{j=2}^{N} \lambda_j(A_r)\right) - \left(r^2 + \sum_{j=2}^{N} (\lambda_j(A_r))^2\right).\end{aligned} \tag{2.27}$$

The adjacency matrix A_r represents a simple graph without self-loops and thus $\text{trace}(A_r) = \sum_{i=1}^{N} \lambda_i(A_r) = 0$. Further, the sum of squared eigenvalues is $\sum_{i=1}^{N} (\lambda_i(A_r))^2 = r \cdot N$, simplifying (2.27) to $(\lambda(A_r))^T \cdot \lambda(A_r^c) = 0$. The following Corollary is proved in [59, p.15], while we provide another proof.

Corollary 3 *The eigenvectors of a regular graph G_r on N nodes and degree r are also eigenvectors of the complete graph K_N on N nodes, implying that a regular graph G_r and the complete graph K_N compose a pair of co-eigenvector graphs.*

Proof: The sum of adjacency matrices A_r of a regular graph G_r and A_r^c of its complement graph G_r^c establishes the adjacency matrix $J - I = A_r + A_r^c$ of the complete graph K_N, as directly follows from (2.23). By substituting (2.12) and (2.26), the previous relation transforms into

$$J - I = \left(\frac{r}{N} + \frac{N-1-r}{N}\right) \cdot u \cdot u^T + \sum_{j=2}^{N} \left(\lambda_j(A_r) + \left(-1-\lambda_j(A_r)\right)\right) \cdot x_j \cdot x_j^T,$$

while after grouping terms, the adjacency matrix of the complete graph K_N becomes

$$J - I = \frac{N-1}{N} \cdot u \cdot u^T - \sum_{j=2}^{N} x_j \cdot x_j^T, \tag{2.28}$$

from which we observe that the complete graph K_N, together with a regular graph G_r (or with its complement graph G_r^c) compose a pair of co-eigenvector graphs, which completes the proof. □

Corollary 4 *Not each set of eigenvectors of the complete graph K_N can represent the eigenvectors of a regular graph G_r.*

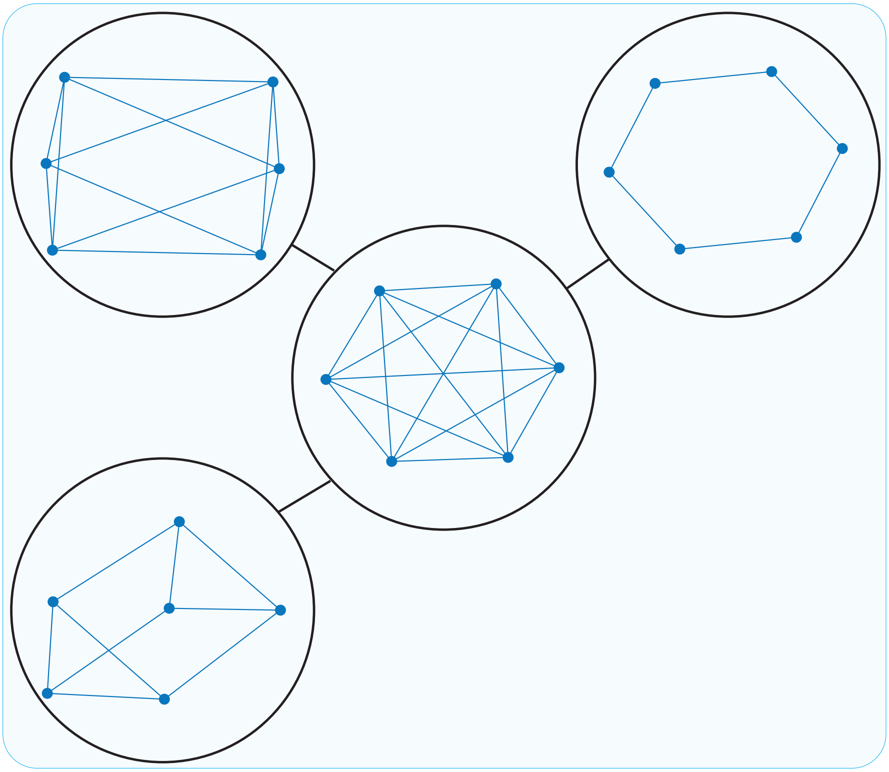


Figure 2.5: Example of pairs of regular co-eigenvector graphs on $N = 6$ nodes. Each regular graph is enclosed in a circle, where circles are connected if the two corresponding regular graphs compose a pair of co-eigenvector graphs.

Proof: In Section 2.3.1, we have shown two orthogonal eigenvector matrices of the complete graph with extremal different rank(Ξ) = 1 and rank(Ξ) = $N-1$. Corollary 1 tells us that the $N \times N$ eigenvector matrix X_{K_N} in (2.10) with rank(Ξ) = $N-1$ determines the complete graph K_N uniquely. In other words, the eigenvectors in (2.10) cannot be the eigenvectors of a non-complete regular graph G_r, although the eigenvectors of any regular graph G_r can also be the eigenvectors of the complete graph K_N. As illustrated by X_{K_N} in (2.10), the reverse does not always hold, which completes the proof. □

Corollary 3 shows that a regular graph G_r together with the complete graph K_N compose a pair of co-eigenvector graphs. However, Corollary 4 informs us that two regular graphs G_{r_1} and G_{r_2} do not form a pair of co-eigenvector graphs, in general. Figure 2.5 presents the pairs of co-eigenvector graphs of size $N = 6$ that are regular graphs.

2.5.2. IRREGULAR CO-EIGENVECTOR GRAPHS

The definition of co-eigenvector graphs imposes a strong constraint on the $N \times N$ adjacency matrix A of an undirected graph G. The $N \times 1$ all-one vector u is [51, Sec. 3.3] the only eigenvector, corresponding to the principal eigenvalue $\lambda_1 = r$ of a regular graph G_r with degree r. Thus, a regular graph G_r and an irregular graph G_1 cannot form a pair of co-eigenvector graphs. Therefore, it is relevant to determine how often co-eigenvector graphs emerge among irregular graphs.

We consider the $N \times N$ adjacency matrix A_1 of a graph G_1 and the $N \times N$ adjacency matrix A_2 of a relabeled graph G_2, such that

$$A_2 = P^T \cdot A_1 \cdot P, \tag{2.29}$$

where the $N \times N$ permutation matrix P [51, p. 21] is an orthogonal matrix, satisfying $P^T P = I$. In other words, the adjacency matrices A_1 and A_2 define two isomorphic graphs. While G_1 and G_2 are co-spectral graphs and share the same set of eigenvalues, because a permutation does not influence eigenvalues [51], they are not a pair of different co-eigenvector graphs.

Graph relabelling does not affect the eigenvalues of an adjacency matrix. On the other side, two isomorphic graphs in general do not constitute a pair of co-eigenvector graphs.

Corollary 5 *Consider a pair of co-eigenvector graphs G_1 and G_2, with the corresponding $N \times N$ adjacency matrices A_1 and A_2. When using the same $N \times N$ permutation matrix P, the relabeled graphs G_1 and G_2 still compose a pair of co-eigenvector graphs.*

Proof: The i-th eigenvector x_i corresponds to the i-th eigenvalue $\lambda_i(A_1)$, but also to the i-th eigenvalue $\lambda_i(A_2)$. After permutation with P, the relabeled eigenvector $P^T \cdot x_i$ satisfies the eigenvector equation for both relabeled graphs

$$\begin{aligned}
P^T \cdot A_1 \cdot P \cdot (P^T \cdot x_i) &= P^T \cdot A_1 \cdot x_i = \lambda_i(A_1) \cdot (P^T \cdot x_i) \\
P^T \cdot A_2 \cdot P \cdot (P^T \cdot x_i) &= P^T \cdot A_2 \cdot x_i = \lambda_i(A_2) \cdot (P^T \cdot x_i),
\end{aligned}$$

where $i \in \mathcal{N}$. Thus, relabeled graphs G_1 and G_2 share eigenvectors, which completes the proof. □

Corollary 5 is understood geometrically. The N eigenvectors of an adjacency matrix A define a polytope on N points in the N-dimensional space. If two adjacency matrices A_1 and A_2 form a pair of co-eigenvector graphs, the $N \times N$ eigenvector matrix X of both adjacency matrices contains the same polytope in the N-dimensional space. The permutation matrix P changes the coordinate system, but not the nature of the polytope on N points.

2.6. IDENTIFYING CO-EIGENVECTOR GRAPHS

We identify pairs of co-eigenvector graphs of different size N. Firstly, for a fixed N, we create all possible unlabeled graphs. The first co-eigenvector graphs, that are *not* regular graphs, occur for $N = 6$. We present an algorithm, with metacode in Figure 2.6, for identifying pairs of co-eigenvector graphs, among all possible connected, irregular graphs with N nodes based on permutation or relabeling (Section 2.5.2). The $N \times N$ adjacency matrix A of each possible unlabeled graph with N nodes is provided as input to the algorithm. Using the double for loop (line 2-3), we examine each pair of graphs. Graph relabeling in (2.29) affects eigenvectors. Therefore, we need to account for each possible permutation whether a pair of non-isomorphic graphs share the same eigenvectors. In line 9, we define each possible $N \times N$ permutation matrix P and observe that the matrix $\left(P \cdot X_j\right)^T \cdot A_i \cdot \left(P \cdot X_j\right)$ is a diagonal matrix only if $\left(P \cdot X_j\right) = X_i$. The proposed algorithm returns the $N_u \times N_u$ matrix C, whose entry $C_{ij} = 1$ if graphs G_i and G_j share the same eigenvectors, otherwise $C_{ij} = 0$.

CoEigenvectorGraphs($A_1, A_2, \ldots A_{N_u}$)

Input: $A_1, A_2, \ldots A_{N_u}$
Output: C
1. $C \leftarrow O_{N_u \times N_u}$
2. **for** $i \leftarrow 1$ to $N_u - 1$
3.   **for** $j \leftarrow i+1$ to N_u
4. $X_i \leftarrow N \times N$ eigenvector matrix of A_i
5. $X_j \leftarrow N \times N$ eigenvector matrix of A_j
6. $m \leftarrow 1$
7.   **while** $\left(C_{ij} = 0\right)$ and $(m < N!)$
8. $P_m \leftarrow N \times N$ m-th permutation matrix
9. $T_i \leftarrow (P_m \cdot X_i)^T \cdot A_j \cdot (P_m \cdot X_i)$
10. $T_j \leftarrow \left(P_m \cdot X_j\right)^T \cdot A_i \cdot \left(P_m \cdot X_j\right)$
11.   **if** $(I \circ T_i = T_i)$ or $\left(I \circ T_j = T_j\right)$
12. $C_{ij} \leftarrow 1, C_{ji} \leftarrow 1$
13.   **end if**
14. $m \leftarrow m+1$
15.   **end while**
16.   **end for**
17. **end for**
18. **return** C

Figure 2.6: Pseudocode for identifying co-eigenvector graphs among all possible unlabeled graphs with N nodes (in total N_u of them), provided as input.

Computing all N_u unlabeled graphs on N nodes is intractable for large N, because their number increases as $O\left(\frac{2^{\binom{N}{2}}}{N!}\right)$. Further, the proposed algorithm cannot guarantee that each pair of co-eigenvector graphs, for a given network size N, is identified. The limitation is due to the fact that some graphs may contain multiple sets of eigenvectors (i.e. multiple different orthogonal X-matrices), while the algorithm in Figure 2.6 computes, for each adjacency matrix A_i, only one $N \times N$ eigenvector matrix X_i (line 4-5).

Some examples of irregular co-eigenvector graphs with $N = 6$ nodes are drawn in Figure 2.7. The algorithm identified two triples of co-eigenvector graphs with $N = 6$ nodes. Figure 2.8 overviews of the identified irregular, connected and unlabeled, co-eigenvector graphs with $N = 7$ nodes.

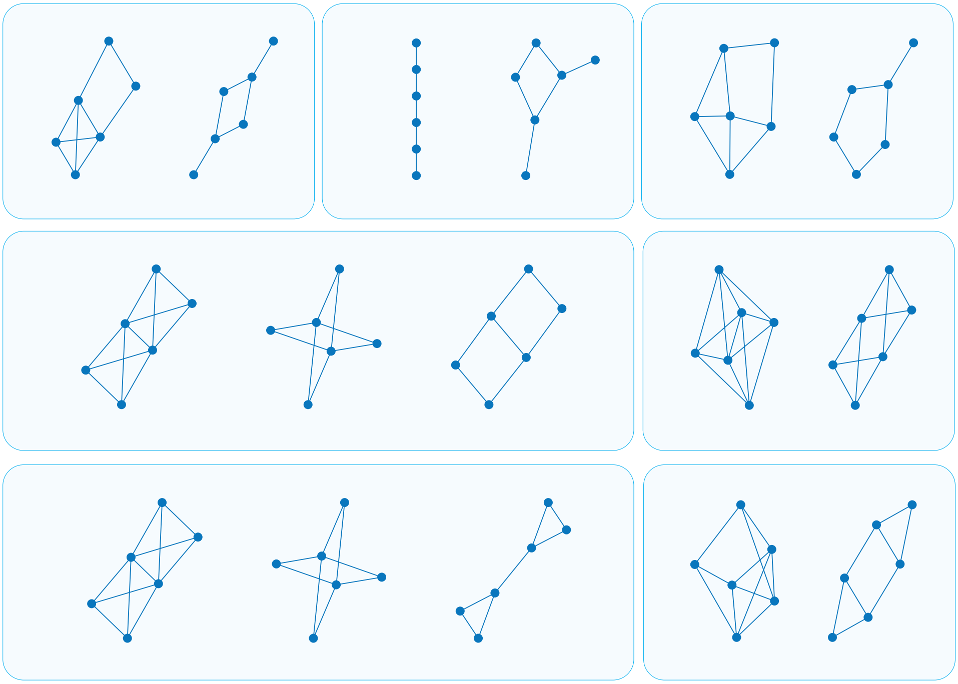


Figure 2.7: Pairs of non-regular, unlabeled, co-eigenvector graphs with $N = 6$ nodes.

2.7. CONCLUSION AND OPEN QUESTIONS

The proof of Theorem 1 relies on the zero-one structure of the adjacency matrix and reveals that only unweighted graphs can be recovered when $\text{rank}(\Xi) < N-1$. The idea to reconstruct the unweighted, undirected graph from the orthogonal eigenvector matrix X of the adjacency matrix A can be extended similarly to the orthogonal eigenvector matrix Z of the Laplacian $Q = \Delta - A$, as outlined in Appendix B.1.3. The remainder of the chapter has deduced properties of co-eigenvector graphs. In particular, irregular co-eigenvector graphs, that are less trivial to find than their regular companions, are found by a rather exhaustive algorithm, based on Theorem 1 and the rank of the matrix Ξ.

A deeper knowledge of the matrix Ξ is desirable. The meaning of the $\text{rank}(\Xi)$ turns out to be difficult. For example, if the graph is connected, then $\text{rank}(\Xi)$ can be smaller than $N-1$. The reverse also is observed: if $\text{rank}(\Xi) = N-1$, then the graph can be disconnected. The relation between $\text{rank}(\Xi)$ and the number of distinct eigenvalues of the adjacency matrix A is also unclear. The relation to the diameter of the graph needs to be investigated. It is also unclear whether the matrix Ξ is diagonalizable. Since Ξ is doubly-stochastic, the underlying associated Markov graph is van2014performance However, an irreducible matrix may still possess a Jordan block. Another question concerns the number of co-eigenvector graphs of size N and its relation to rank(Ξ). Simulations suggest that the less structure or symmetry a graph possesses, the higher the probability that rank(Ξ) $= N - 1$.

Earlier [60], the reconstructability coefficient θ was defined as the smallest value of m in $\widetilde{A} = \sum_{k=1}^{m} \lambda_k x_k x_k^T$ that allows us to exactly reconstruct the zero-one adjacency matrix A. Figure 2.9 seems to suggest for small Erdős-Rényi graphs that there is hardly any correlation between the reconstructability coefficient θ and rank(Ξ). Perhaps, other graph classes or/and larger graphs may reveal a relation?

Furthermore, one may ask whether the confinement to undirected graphs, that possess a symmetric adjacency matrix, can be relaxed to directed graphs, whose general eigenvector matrix X may be complex. If that extension is favorable, one may consider Hermitian matrices, which may open possible applications to quantum mechanics and quantum computing. Data measured over time on complex networks is often related to a dynamic process that runs on the underlying graph. If that dynamic process is linear or proportional to the graph (as e.g. the flow of currents in a resistor or impedance network [23]), then the eigendecomposition of the graph is reflected by that data and Theorem 1 may provide insight into the underlying topology on which data is collected.

At last, from an information theoretical point of view discussed in [61], Theorem 1 is not surprising, because the presentation of the orthogonal X matrix needs more digits (i.e. more information) than the zero-one adjacency matrix.

Acknowledgements We are grateful to Karel Devriendt, Xiangrong Wang and Willem Haemers for useful comments and to Geert Leus for informing us about the article of Segarra *et al.* [62], whose Proposition 1 is related to Theorem 1.

2

2

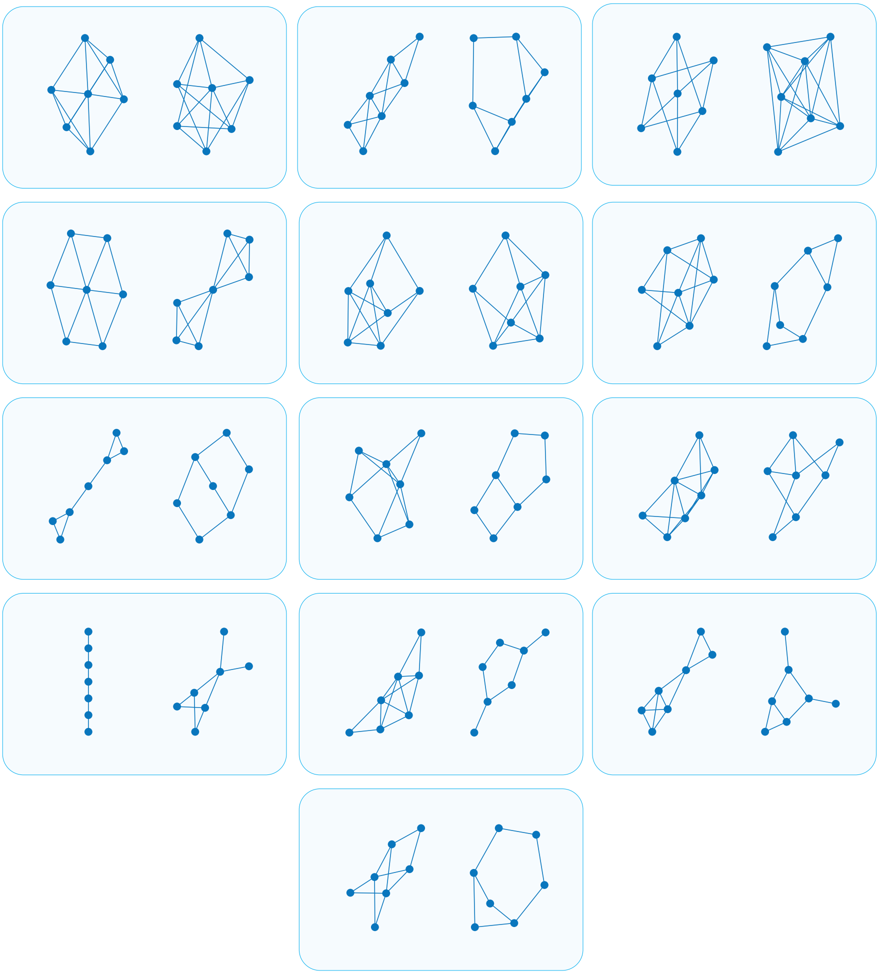


Figure 2.8: Pairs of non-regular, unlabeled, co-eigenvector graphs with $N = 7$ nodes.

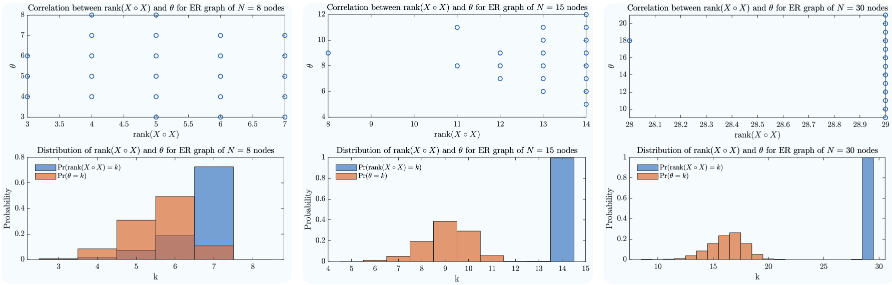


Figure 2.9: Correlation between reconstructability coefficient θ and the rank(Ξ) for ER graphs with $N = 8$ (left-hand side figures), $N = 15$ (figures in the middle) and $N = 30$ nodes (right-hand side figures). The link density p is varied between $p = \frac{3\log N}{4N}$ and $p = \frac{3\log N}{2N}$, while 10^5 connected graphs are generated for each network size N.

3

Number of paths in a graph

Make everything as simple as possible,
but not simpler.

Albert Einstein

The k-th power of the adjacency matrix of a simple undirected graph represents the number of walks with length k between pairs of nodes. As a walk where no node repeats, a path is a walk where each node is only visited once. The set of paths constitutes a relatively small subset of all possible walks. We introduce three types of walks, representing subsets of all possible walks. Considered types of walks allow for deriving an analytic solution for the number of paths of a certain length between node pairs in a matrix form. Depending on the path length, different approaches possess the lowest computational complexity. We also propose a recursive algorithm for determining all paths in a graph, which can be generalised to directed (un)weighted networks.

This chapter is based on [63].

3.1. INTRODUCTION

Networks emerge naturally in many real-world systems [1, 2]. From a Network Science perspective, a network consists of an underlying topology, called the graph, and a dynamic process taking place on the graph. Examples of real-world networks include infrastructural networks (such as road traffic networks, the Internet and the power grid networks), biological networks (such as the protein interaction network) and social networks.

A walk in a graph represents a sequence of nodes, where adjacent nodes in the sequence share a common link, as defined below in Definition 5. A path, defined in Definition 7, is a walk where no node repeats in the sequence. Paths reflect the cost of connections between node pairs in a network, where the cost is quantified by weights of the links in the paths. If all link weights are constant or unity, then the weight of a path equals the hopcount [64, Chapter 16], the number of hops or links in the path. A significant part of the literature on paths in graphs considers the problem of determining the number of paths, both in random graphs [65] or in special types of graphs [66]. However, to the best of our knowledge, an explicit solution for the number of paths of a certain length in a matrix form is not yet known. Our idea is to derive an analytic solution for the number of length k paths between node pairs by removing those walks traversing a node multiple times from all possible walks with hopcount k, contained in the k-th power of the adjacency matrix.

The problem of determining whether there is a path in a graph with at least k links is NP-complete [67]. Williams proposed in [68] an algorithm for finding paths with hopcount k, while Schmid *et al.* proposed an logarithm in [69] for computing Tutte Paths. A graph is Hamiltonian if there is a path traversing each node in a graph. Bjorklund proposed in [70] a Monte Carlo algorithm that solves the Hamiltonian problem. Bax introduced an algorithm for the Hamiltonian path problem in [71], based on the inclusion-exclusion formula. We generalise the approach of Bax in [71] and derive the solution for the number of paths with any hopcount k.

Section 3.2 introduces the notion of walks and paths. We firstly introduce walks traversing a node multiple times in Section 3.3 and derive an analytic solution for the number of paths between node pairs with hopcount $k \leq 4$. Section 3.4 analyses walks traversing a node exactly once, while those walks not traversing a node we examine in Section 3.5. In Section 3.6, we provide a recursive algorithm that identifies all possible paths in a graph. Finally, we conclude in Section 3.7.

3.2. WALKS IN A GRAPH

Definition 5 *A walk of length k from node i to node j is a sequence of k links of the form $(n_0 \to n_1)(n_1 \to n_2)\dots(n_{k-1} \to n_k)$, where $n_0 = i$ and $n_k = j$.*

The length of a walk is the number of links in the walk and is often referred to as the walk hopcount [51]. The first node in the sequence n_0 is the source node, while a walk ends with the destination node n_k. The Definition 5 naturally leads to the question of how many ways are there to reach node i from node j, in k hops.

Theorem 6 *The number of walks of length k from node i to node j is equal to the element $\left(A^k\right)_{ij}$.*

Proof Provided in [51, p. 26] □

The k-th power of the adjacency matrix A^k contains the number of walks of length k between each pairs of nodes in the graph. Any matrix derived solely from the adjacency matrix A, either via matrix product or the Hadamard product, carries information about walks. We denote the set of all possible walks of length k as the k-dimensional walk space $\mathscr{W}[k]$, while the corresponding $N \times N$ walk matrix is A^k. Since a matrix commutes with itself, it holds that

$$A^k = A^{k-p} \cdot A^p = A^p \cdot A^{k-p}, \tag{3.1}$$

where p is here an integer between $1 \leq p \leq k$, although (3.1) holds for any complex p for which the matrix A^p exists. The relation (3.1) teaches us that a walk can be split into sub-walks, while the walk matrix can be obtained as the multiplication of walk matrices of the corresponding sub-walks.

3

3.2.1. PATHS IN A GRAPH

Definition 7 *A path is a walk in which all nodes are different. A path of length k is defined by a sequence of $k+1$ node pairs: $(n_0 \to n_1)(n_1 \to n_2)\ldots(n_{k-1} \to n_k)$, where $n_l \neq n_m$ for all $0 \leq l \neq m \leq k$.*

Paths account for a relatively small subset of all possible walks $\mathscr{W}[k]$ of length k. We denote the set of all possible paths of length k as $\mathscr{P}[k]$, where $\mathscr{P}[k] \subseteq \mathscr{W}[k]$, while equality holds only for $k = 1$, as shown later in (3.11). The $N \times N$ path matrix P_k contains the number of paths of length k between any pair of nodes, with $(P_k)_{ij}$ denoting the number of paths between node i and node j, of hopcount k.

In the following sections, we introduce three different types of walk sets: walks with a node reappearing in the node sequence, walks where a node is not traversed and those walks traversing a node exactly once. Based on each mentioned type of walks, we derive an analytic solution for the $N \times N$ path matrix P_k with hopcount k.

3.3. NODE REAPPEARANCE IN A WALK

A walk, introduced in Definition 5, can traverse the same node multiple times, defining the first walk type we consider in this section.

Definition 8 *The set of all possible walks of length k, where the same node appears at least twice in the node sequence, on positions i and j (i.e. $n_i = n_j$), where $j > i+1$, is denoted as $\mathscr{W}_{(i,j)}[k]$. The $N \times N$ walk matrix with the number of such walks between any pair of nodes is denoted by $M(\mathscr{W}_{(i,j)}[k])$.*

From Definition 8, we observe the following identity $\mathscr{W}_{(i,j)}[k] = \mathscr{W}_{(j,i)}[k]$, because $n_i = n_j$. The introduced constraint $j > i+1$ excludes two trivial cases. When $i = j$, the corresponding set of walks is actually the set of all possible walks of length k, i.e. $\mathscr{W}_{(i,i)}[k] = \mathscr{W}[k]$. On the other side, when $j = i+1$, the corresponding set of walks $\mathscr{W}_{(i,i+1)}[k] = \emptyset$, because a node cannot be adjacent to itself in a walk sequence, because there are no self-loops in a simple network. Figure 3.1 illustrates a few examples of walks where a node is traversed multiple times.

It is of particular interest to consider walks, where the source node is also the destination node, i.e. the set $\mathscr{W}_{(0,k)}[k]$.

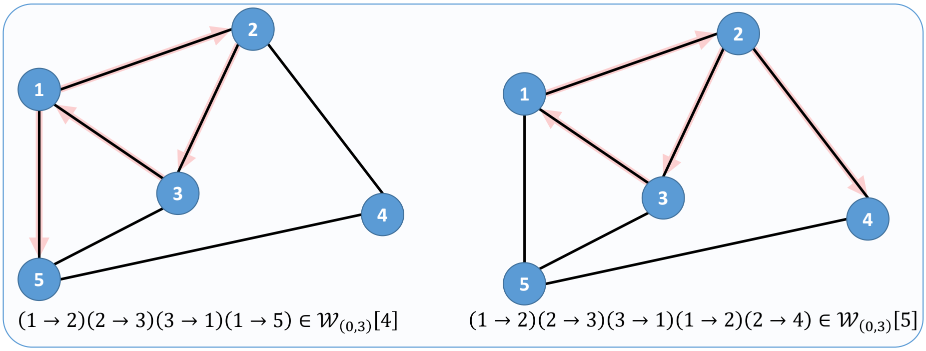


Figure 3.1: Examples of walks with node reappearance. Traversed links are colored in red, while arrows follow labeling in the node sequence.

Definition 9 *A closed walk of length k is a walk that starts in node i and returns, after k hops, to that same node i (i.e. where $n_0 = n_k$). The set of all possible closed walks of length k is $\mathcal{W}_{(0,k)}[k]$. The corresponding $N \times N$ walk matrix is $M\left(\mathcal{W}_{(0,k)}[k]\right) = I \circ A^k$, where I is the $N \times N$ identity matrix, while $\circ$ denotes the Hadamard product. The total number of closed walks of length k in a graph is trace(A^k) and equals trace$(A^k) = \sum_{i=1}^{N} \lambda_i^k$, where λ_j is the j-th largest eigenvalue of the adjacency matrix [51].*

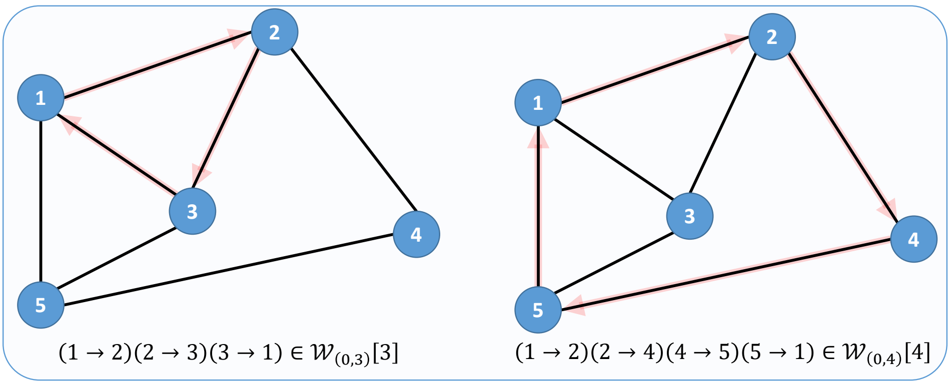


Figure 3.2: Examples of closed walks. Traversed links are colored in red, while arrows follow labeling in the node sequence.

A few examples of closed walks are depicted in Figure 3.2. A walk set $\mathcal{W}_{(i,j)}[k]$ can be split into three sub-walks: the set of all possible walks $\mathcal{W}[i]$ of length i, the set of closed walks $\mathcal{W}_{(0,j-i)}[j-i]$ of length $j-i$ and the set of all possible walks $\mathcal{W}[k-j]$ of length $k-j$. The corresponding $N \times N$ walk matrix $M\left(\mathcal{W}_{(i,j)}[k]\right)$, after applying Theorem 6, Definition

9 and relation (3.1) becomes

$$M\left(\mathscr{W}_{(i,j)}[k]\right) = A^i \cdot \left(I \circ A^{j-i}\right) \cdot A^{k-j}. \tag{3.2}$$

The $N \times N$ walk matrix $M\left(\mathscr{W}_{(i,j)}[k]\right)$ in (3.2) is asymmetric, since the order of nodes in a walk sequence is labelled. In other words, if we label nodes of the walk sequence in $\mathscr{W}_{(i,j)}[k]$ from the destination node to the source node, we obtain the walk matrix $M\left(\mathscr{W}_{(k-j,k-i)}[k]\right) = A^{k-j} \cdot \left(I \circ A^{j-i}\right) \cdot A^i$, which together with (3.2) leads to the following identity

$$M\left(\mathscr{W}_{(k-j,k-i)}[k]\right) = M^T\left(\mathscr{W}_{(i,j)}[k]\right). \tag{3.3}$$

3

Identity (3.3) holds only for undirected networks, informing us that a walk can be performed from the source towards the destination node, but also in the reverse order, because all links are undirected.

3.3.1. ANALYTIC SOLUTION FOR THE $N \times N$ PATH MATRIX P_k

Theorem 10 *The set of all possible walks $\mathscr{W}[k]$ of length k consists of the following subsets*

$$\mathscr{W}[k] = \left(\bigcup_{i=0}^{k-2} \bigcup_{j=i+2}^{k} \mathscr{W}_{(i,j)}[k]\right) \cup \mathscr{P}[k]. \tag{3.4}$$

Proof A walk of length k has either a repeating node in its sequence or represents a path of length k. Thus, by computing the set union of walk sets, defining walks with all possible repetitions of a node, and the set of paths, we obtain the set $\mathscr{W}[k]$ of all possible walks of length k in (3.4), which completes the proof. □

A walk of length k is either a path or a node is traversed multiple times. From Definition 8 and Definition 7, we observe

$$\mathscr{W}_{(i,j)}[k] \cap \mathscr{P}[k] = \emptyset, \tag{3.5}$$

where $0 \le i \le k-2$ and $j \ge i+2$. Based on Theorem 6, equations (3.4) and (3.5), we obtain a general solution for the $N \times N$ path matrix P_k

$$P_k = A^k - M\left(\bigcup_{i=0}^{k-2} \bigcup_{j=i+2}^{k} \mathscr{W}_{(i,j)}[k]\right). \tag{3.6}$$

The double set union in (3.6) defines the union of $\frac{k\cdot(k-1)}{2}$ walk sets. Thus, the number of walk sets of the form $\mathscr{W}_{(i,j)}[k]$ increases as a square function of the hopcount k. In general, the sets of walks $\mathscr{W}_{(i_1,j_1)}[k]$ and $\mathscr{W}_{(i_2,j_2)}[k]$ overlap, which complicates the computation of equation (3.6). Therefore, we apply the inclusion-exclusion formula.

3.3.2. INCLUSION-EXCLUSION FORMULA

The inclusion-exclusion formula [64, p. 10] defines the cardinality of the union of sets and thus transforms the second term on the right-hand side of the equation in (3.6) as follows

3

$$
\begin{aligned}
M\left(\bigcup_{i=0}^{k-2}\bigcup_{j=i+2}^{k}\mathscr{W}_{(i,j)}[k]\right) &= \sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k]\right)\\
&-\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k]\cap\mathscr{W}_{(i_2,j_2)}[k]\right)\\
&+\cdots+\\
&+(-1)^{k-1}\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\cdots\sum_{i_k=i_{k-1}}^{k-2}\sum_{j_k=q_k}^{k} M\left(\bigcap_{z=1}^{k}\mathscr{W}_{(i_z,j_z)}[k]\right),
\end{aligned}
\tag{3.7}
$$

where $q_m = j_{m-1}+2$ if $i_m = i_{m-1}$, otherwise $q_m = i_m + 2$ with $1 < m \leq k$. Since there are $\frac{k\cdot(k-1)}{2}$ different walk sets with a repeating node, the total number of terms in the inclusion-exclusion formula in (3.7) is $2^{\left(\frac{k\cdot(k-1)}{2}\right)}-1$. However, not all walk set intersections in (3.7) define possible walks, because a node cannot be adjacent to itself in the walk sequence. Under the assumption that each matrix in (3.6) has an analytic solution, the complexity of computing the $N\times N$ path matrix P_k with the number of path of length k between node pairs is $O\left(kN^3 2^{\frac{k^2-k}{2}}\right)$.

3.3.3. RECURSIVE SOLUTION FOR THE $N\times N$ PATH MATRIX P_k

In this subsection we reason why an analytic recursive solution for the $N\times N$ path matrix P_k of the hopcount k seems infeasible. From (3.6), we derive the following identity

$$
P_k + M\left(\bigcup_{i=0}^{k-2}\bigcup_{j=i+2}^{k}\mathscr{W}_{(i,j)}[k]\right) = P_{k-1}\cdot A + M\left(\bigcup_{i=0}^{k-3}\bigcup_{j=i+2}^{k-1}\mathscr{W}_{(i,j)}[k-1]\right)
$$

from where we conclude that the $N\times N$ path matrix P_k of length k obeys the following recursion

$$
P_k = P_{k-1}\cdot A - F_k, \tag{3.8}
$$

where the $N\times N$ matrix F_k is defined as

$$
F_k = M\left(\bigcup_{i=0}^{k-2}\bigcup_{j=i+2}^{k}\mathscr{W}_{(i,j)}[k]\right) - M\left(\bigcup_{i=0}^{k-3}\bigcup_{j=i+2}^{k-1}\mathscr{W}_{(i,j)}[k-1]\right)\cdot A. \tag{3.9}
$$

In Appendix C.1, we derive the first two sum terms of the $N\times N$ matrix F_k, that illustrate the difficulty to derive a complete closed form solution. To provide an argument for why the recursive solution does not seem possible, we denote a path $p_k = (n_0 \to n_1)(n_1 \to n_2)\ldots(n_{k-1}\to n_k)$ as a node sequence, where $n_l \neq n_m$ for all $0 \leq l \neq m \leq k$. The number of paths between node i and node j of length $k+1$ can be computed as follows

$$
(P_{k+1})_{ij} = \sum_{p_k\in\mathscr{P}[k]} \mathbf{1}_{\{n_0=i\}}\mathbf{1}_{\{n_k\in\mathscr{N}_j\}}\mathbf{1}_{\{n_2\neq j\}}\mathbf{1}_{\{n_3\neq j\}}\ldots\mathbf{1}_{\{n_{k-2}\neq j\}}, \tag{3.10}
$$

where the set of node j neighbours is denoted as $\mathscr{N}_j$, while $\mathbf{1}_x$ is the indicator function that equals 1 if statement x is true, otherwise $\mathbf{1}_x = 0$. On the other side, we observe from

(3.8) that the first term of the recursive solution

$$(P_{k+1})_{ij} = \sum_{m=1}^{N} (P_k)_{im} \cdot a_{mj} - (F_k)_{ij}$$

examines only the first two conditions from (3.10), because $\sum_{p_k \in \mathscr{P}[k]} \mathbf{1}_{\{n_0=i\}} \mathbf{1}_{\{n_k \in \mathcal{N}_j\}} = (P_k \cdot A)_{ij}$. The element $(P_k \cdot A)_{ij}$ contains the number of walks between node i and node j of length $k+1$, composed of all paths of length k, from node i to an adjacent node $m \in \mathcal{N}_j$. However, not each such a walk represents a path. To obtain length $k+1$ paths, we need to distract from $(P_k \cdot A)_{ij}$ those paths of length k between node i and node $m \in \mathcal{N}_j$, traversing also node j. The $N \times N$ path matrix P_k counts the number of length k paths between pairs of nodes, without comprising information about traversed nodes per each path. Therefore, the recursive solution in (3.8) requires manipulating an exponentially large number of walk matrices, to account for walks in $P_k \cdot A$ that are not paths.

In the following subsections, we derive an explicit form of the $N \times N$ path matrix P_k up to hopcount $k \leq 4$.

3.3.4. Path matrix P_1 of length $k = 1$

The $N \times N$ adjacency matrix A, by its definition, defines all paths of length $k = 1$ and thus the $N \times N$ path matrix P_1

$$P_1 = A, \tag{3.11}$$

because there is a path of length $k = 1$ between node i and j only if they share a link (i.e. if $a_{ij} = 1$). Only in case $k = 1$, the set of all walks

$$\mathscr{W}[1] = \mathscr{P}[1]$$

consists of only paths, because there are no self-loops in simple networks.

3.3.5. Path matrix P_2 of length $k = 2$

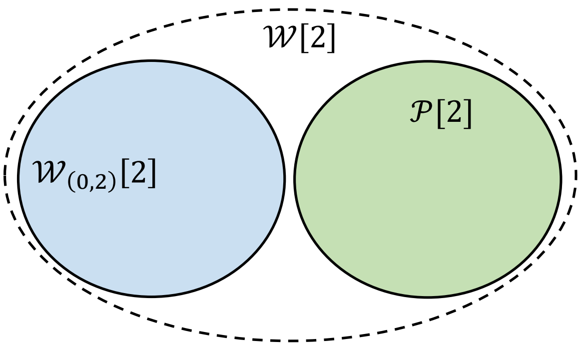


Figure 3.3: The set $\mathscr{W}[2]$ of all possible walks of length $k = 2$ and subsets.

When the hopcount $k > 1$, walks with repeating nodes emerge, revealing the graph connectivity patterns. The only possible repetition of nodes in a walk with the hopcount $k = 2$ is when $n_0 = n_2$, forming the set $\mathscr{W}_{(0,2)}[2]$ of closed walks. Since walks of length $k = 2$ consist of either closed walks or paths, the set $\mathscr{W}[2]$ of all possible walks of length $k = 2$ is as follows

$$\mathscr{W}[2] = \mathscr{W}_{(0,2)}[2] \cup \mathscr{P}[2].$$

The Venn diagram for the walk space of the hopcount $k = 2$ is provided on Figure 3.3. From (3.2) we obtain the $N \times N$ walk matrix $M\left(\mathscr{W}_{(0,2)}[2]\right)$ as follows

$$M\left(\mathscr{W}_{(0,2)}[2]\right) = I \circ A^2.$$

By importing the above equation into (3.6), we obtain the $N \times N$ path matrix P_2

$$P_2 = A^2 - I \circ A^2. \tag{3.12}$$

3.3.6. PATH MATRIX P_3 OF LENGTH $k = 3$

The set $\mathscr{W}[3]$ of all walks with the hopcount $k = 3$ represents the set union of the walk sets with any possible node repetition and the path set $\mathscr{P}[3]$. By applying (3.4) for the hopcount $k = 3$, we obtain

$$\mathscr{W}[3] = \mathscr{W}_{(0,2)}[3] \cup \mathscr{W}_{(0,3)}[3] \cup \mathscr{W}_{(1,3)}[3] \cup \mathscr{P}[3].$$

By importing (3.6), the above equation transforms

$$\begin{aligned} A^3 = {} & M\left(\mathscr{W}_{(0,2)}[3]\right) + M\left(\mathscr{W}_{(0,3)}[3]\right) + M\left(\mathscr{W}_{(1,3)}[3]\right) + P_3 \\ & - M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(0,3)}[3]\right) - M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(1,3)}[3]\right) - M\left(\mathscr{W}_{(0,3)}[3] \cap \mathscr{W}_{(1,3)}[3]\right) \\ & + M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(0,3)}[3] \cap \mathscr{W}_{(1,3)}[3]\right). \end{aligned} \tag{3.13}$$

The set intersection $(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(0,3)}[3])$ defines walks where $n_0 = n_2 = n_3$. Since a node is not adjacent to itself in a simple network (i.e. $a_{ii} = 0$), such walks do not exist and thus $(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(0,3)}[3]) = \emptyset$. The same reasoning holds for the sets $(\mathscr{W}_{(0,3)}[3] \cap \mathscr{W}_{(1,3)}[3]) = \emptyset$ and $(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(0,3)}[3] \cap \mathscr{W}_{(1,3)}[3]) = \emptyset$, which simplifies the relation (3.13)

$$A^3 = M\left(\mathscr{W}_{(0,2)}[3]\right) + M\left(\mathscr{W}_{(0,3)}[3]\right) + M\left(\mathscr{W}_{(1,3)}[3]\right) + P_3 - M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(1,3)}[3]\right). \tag{3.14}$$

The set $\mathscr{W}[3]$ of all possible walks with the hopcount $k = 3$, the walk subsets $\mathscr{W}_{(0,2)}[3], \mathscr{W}_{(0,3)}[3], \mathscr{W}_{(1,3)}[3]$ with a repeating node and the path set $\mathscr{P}[3]$ are presented in Figure 3.4.

A walk of length $k = 3$ where $n_0 = n_2$ and $n_1 = n_3$ starts from a node i, visits a neighbouring node j, traverses again the node i and ends in the adjacent node j. Thus, for a pair of adjacent nodes i and j, there is only one such a path. We denote the $N \times N$ corresponding walk matrix $M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(1,3)}[3]\right)$

$$M\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(1,3)}[3]\right) = A \circ A \circ A = A.$$

Finally, after importing the above equation and (3.2) into (3.14), we obtain

$$P_3 = A^3 - \left(I \circ A^2\right) \cdot A - I \circ A^3 - A \cdot \left(I \circ A^2\right) + A. \tag{3.15}$$

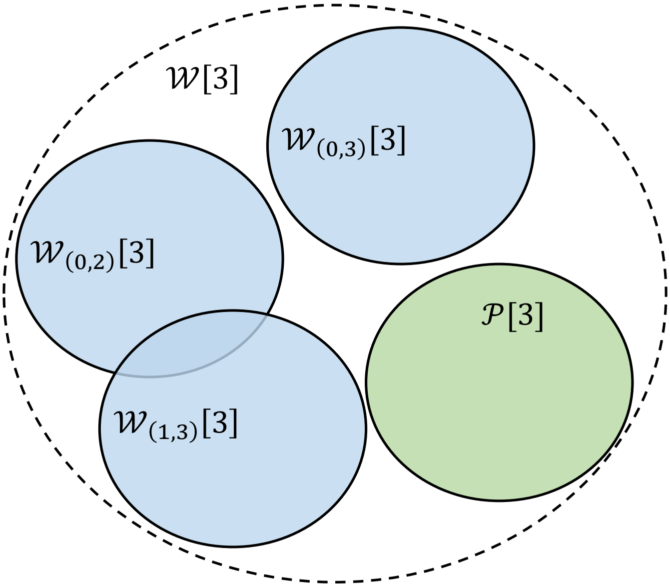


Figure 3.4: The set of all possible walks of length $k = 3$, the subset of paths and subsets of walks with a repeating node.

3.3.7. PATH MATRIX P_4 OF LENGTH $k = 4$

The set $\mathscr{W}[4]$ of all possible walks of length $k = 4$ represents the set union of the following sets

$$\mathscr{W}[4] = \mathscr{W}_{(0,2)}[4] \cup \mathscr{W}_{(0,3)}[4] \cup \mathscr{W}_{(0,4)}[4] \cup \mathscr{W}_{(1,3)}[4] \cup \mathscr{W}_{(1,4)}[4] \cup \mathscr{W}_{(2,4)}[4] \cup P[4],$$

The set $\mathscr{W}[4]$ of all walks with the hopcount $k = 4$, the path set $\mathscr{P}[4]$ and the walk sets with a repeating node are presented in Figure 3.5. Not all defined walk subsets with repeating nodes overlap, as presented in Figure 3.5.

$$\begin{aligned}
A^4 = P_3 &+ M\left(\mathscr{W}_{(0,2)}[4]\right) + M\left(\mathscr{W}_{(0,3)}[4]\right) + M\left(\mathscr{W}_{(0,4)}[4]\right) + M\left(\mathscr{W}_{(1,3)}[4]\right) + M\left(\mathscr{W}_{(1,4)}[4]\right) + M\left(\mathscr{W}_{(2,4)}[4]\right) \\
&- M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4]\right) - M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,4)}[4]\right) - M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) \\
&- M\left(\mathscr{W}_{(0,3)}[4] \cap \mathscr{W}_{(1,4)}[4]\right) - M\left(\mathscr{W}_{(0,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) - M\left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4]\right) \\
&- M\left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) - M\left(\mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) \qquad (3.16) \\
&+ M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) + M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(0,4)}[4]\right) \\
&+ M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) + M\left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) \\
&- M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right).
\end{aligned}$$

In the following part, we derive in sequel the walk matrices of the set intersections with three and four walk sets from the above relation.

Walk set $\left(\mathscr{W}_{(0,2)}[3] \cap \mathscr{W}_{(1,3)}[3] \cap \mathscr{W}_{(2,4)}[3]\right)$ defines walks with the hopcount $k = 4$ where $n_0 = n_2 = n_4$ and $n_1 = n_3$ originate from node n_0, visits node n_1, returns to node n_0 and

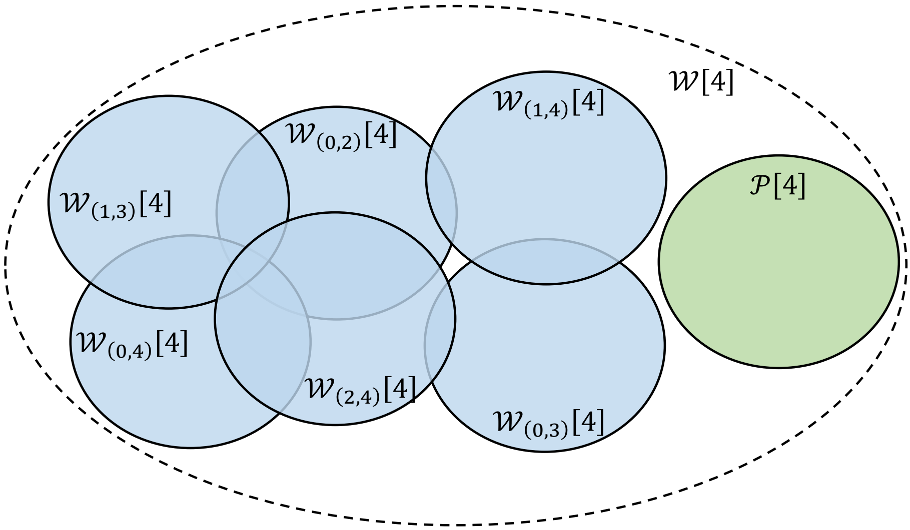


Figure 3.5: The set $\mathscr{W}[4]$ of all possible walks of length $k = 4$, the subset of paths and subsets of walks with repeating nodes.

repeats the same walk pattern, finishing at node n_0.

$$M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) = I \circ (A \cdot (A \circ A \circ A)) = I \circ A^2. \tag{3.17}$$

In addition, walk set $\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(0,4)}[4]\right)$ and $\left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right)$ define walks of length $k = 4$ where $n_0 = n_2 = n_4$ and $n_1 = n_3$. Thus, we observe

$$\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(0,4)}[4]\right) = \left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) = \left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right).$$

On the contrary, the walk set $\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(2,4)}[4]\right)$ defines walks with hopcount $k = 4$ where $n_0 = n_2 = n_4$. Such walks start from a node n_0, visits an adjacent node, returns to node n_0, traverses an adjacent node once more and returns again to node n_0. The corresponding $N \times N$ walk matrix $M\left(\mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right)$ is as follows

$$M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) = (I \circ (A \cdot A)) \cdot (I \circ (A \cdot A)) = \left(I \circ A^2\right)^2. \tag{3.18}$$

The set $M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right)$ defines walks of length $k = 4$ where $n_0 = n_2 = n_4$ and $n_1 = n_3$ and thus

$$M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(0,4)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) = M\left(\mathscr{W}_{(0,2)}[4] \cap \mathscr{W}_{(1,3)}[4] \cap \mathscr{W}_{(2,4)}[4]\right) = I \circ A^2.$$

Finally, we derive the $N \times N$ path matrix P_4 of length $k = 4$

$$\begin{aligned}P_4 = &A^4 - \left(I \circ A^2\right) \cdot A^2 - \left(I \circ A^3\right) \cdot A - I \circ A^4 - A \cdot \left(I \circ A^2\right) \cdot A - A \cdot \left(I \circ A^3\right) - A^2 \cdot \left(I \circ A^2\right) \\ &+3 \cdot \left(I \circ A^2\right)^2 + 3 \cdot A \circ A^2 + I \circ \left(A \cdot \left(I \circ A^2\right) \cdot A\right) + 2 \cdot A^2 \\ &-3 \cdot \left(I \circ A^2\right) - \left(I \circ A^2\right)^2 \\ &+ \left(I \circ A^2\right).\end{aligned} \tag{3.19}$$

Determining walk matrices of all walk subsets in (3.7) with increasing hopcount k becomes intractable. While we provide above an explicit solution for the $N \times N$ path matrix P_k, with $k \leq 4$, already for $k = 5$, providing the explicit enumeration of the number of paths between any pair of nodes becomes far more involving.

3

3.4. WALKS TRAVERSING A NODE EXACTLY ONCE

Applying the inclusion exclusion formula in (3.7) produces an exponential number of matrix terms, which do not seem to be solvable for a general hopcount k. Instead, we consider here walks in which a node appears exactly once.

Definition 11 *The set of all possible walks with length k, where node $i \in \mathcal{N}$ is traversed exactly once, is denoted as $\mathcal{W}_{(i)}[k]$. The corresponding $N \times N$ walk matrix $M\left(\mathcal{W}_{(i)}[k]\right)$ with the number of such walks between node pair equals*

$$\begin{aligned}M\left(\mathcal{W}_{(i)}[k]\right) = &\left(\left(e_i \cdot u^T\right) \circ A\right) \cdot \left(\left((u - e_i) \cdot (u - e_i)^T\right) \circ A\right)^{k-1} \\ &+ \sum_{m=1}^{k-1} \left(\left((u - e_i) \cdot (u - e_i)^T\right) \circ A\right)^{m-1} \cdot \left(A \circ \left(u \cdot e_i^T\right)\right) \cdot \left(\left((u - e_i) \cdot (u - e_i)^T\right) \circ A\right)^{k-m} \\ &+ \left(\left((u - e_i) \cdot (u - e_i)^T\right) \circ A\right)^{k-1} \cdot \left(\left(e_i \cdot u^T\right) \circ A\right).\end{aligned}$$

The $N \times N$ walk matrix $M\left(\mathcal{W}_{(i)}[k]\right)$ consists of $k + 1$ terms, because node i can appear on $k + 1$ positions in a walk of length k. We illustrate examples of walks traversing a node exactly once in Figure 3.6.

3.4.1. ANALYTIC SOLUTION FOR THE $N \times N$ PATH MATRIX P_k

Defining the walk sets $\mathcal{W}_{(i)}[k]$ with node $i \in \mathcal{N}$ appearing only once allows us to determine the set of paths $\mathcal{P}[k]$ with hopcount k, not by excluding walks with node reappearance from all possible walks $\mathcal{W}[k]$, but instead as the intersection of those walk sets of the form $\mathcal{W}_{(i)}[k]$, $i \in \mathcal{N}$

$$\mathcal{P}[k] = \mathcal{W}_{(i_0 \in \mathcal{N})}[k] \cap \mathcal{W}_{(i_1 \in \mathcal{N} \setminus i_0)}[k] \cap \cdots \cap \mathcal{W}_{(i_k \in (\mathcal{N} \setminus (i_0 \cup i_1 \cup \cdots \cup i_{k-1})))}[k]. \tag{3.20}$$

Theorem 12 *The $N \times N$ path matrix P_k, whose entries comprise the number of paths with hopcount k between any pair of nodes is defined as follows*

$$P_k = \sum_{i_0 \in \mathcal{N}} \sum_{i_1 \in \mathcal{N} \setminus i_0} \cdots \sum_{i_k \in \mathcal{N} \setminus (i_0 \cup i_1 \cup \cdots \cup i_{k-1})} \prod_{z=1}^{k} \left(\left(e_{i_{z-1}} \cdot e_{i_z}^T\right) \circ A\right), \tag{3.21}$$

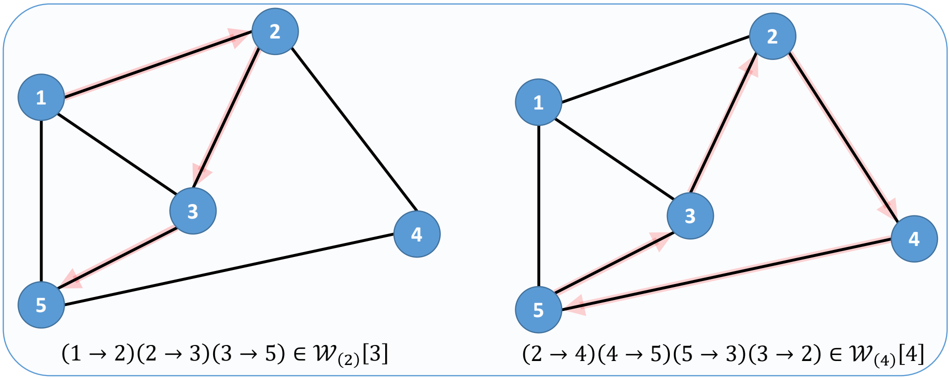


Figure 3.6: Examples of walks in which a node is traversed exactly once. Traversed links are colored in red, while arrows follow labeling in the node sequence.

3

or alternatively

$$P_k = \sum_{i_0=1}^{N} \sum_{i_1=1}^{N} \cdots \sum_{i_k=1}^{N} \prod_{z=1}^{k} \left(\left(e_{i_{z-1}} \cdot e_{i_z}^T \right) \circ A \right).$$

Proof Relation (3.21) examines all possible labeled sequences of $k+1$ nodes. For each sequence $(i_0, i_1, \ldots, i_k)$, we remove all elements from the $N \times N$ adjacency matrix A, except for the element $a_{(i_z-1, i_z)}$ between adjacent nodes in the sequence $\left(e_{i_{z-1}} \cdot e_{i_z}^T \right) \circ A$, where $1 \le z \le k$. If a node sequence composes a path of lenth k, the (i_0, i_k)th element of the product $\prod_{z=1}^{k} \left(\left(e_{i_{z-1}} \cdot e_{i_z}^T \right) \circ A \right)$ equals 1, otherwise 0. □.

The solution for the $N \times N$ path matrix P_k in (3.21) represents a deterministic counterpart to the relation in [65, eq. 6], defining the probability of a path existence between two nodes. For each possible labelled sequence of $k+1$ different nodes, in total $(k+1)!$ of them, relation (3.21) forces all entries of the $N \times N$ adjacency matrix A to zero, expect for the entries between adjacent nodes in the sequence and provides an element one on the position $(i\,j)$, if the remaining elements compose a path of length k between node i and node j. By summing over each possible labelled node sequence, we obtain the $N \times N$ path matrix P_k, with complexity $O\left(k! k N^3\right)$.

3.5. WALKS NOT TRAVERSING A NODE

For a general hopcount k, there are in total $k!$ matrix terms in (3.21), as each node sequence is labelled. Therefore, an explicit enumeration for an arbitrary hopcount k is infeasible. However, when computing the $N \times N$ path matrix P_k in a matrix form, labelling nodes in the sequence is not necessary, because the matrix product naturally preserves the information about the source and destination node of each walk, as illustrated in (3.1). In this section we introduce walks where a node is not traversed. Originally, this type of walks was defined by Bax in [71].

Definition 13 *The set of all possible walks with length $k = N-1$, where node $m \in \mathcal{N}$ is not traversed, is denoted as $\mathcal{W}_m$. The $N \times N$ corresponding walk matrix with the number of such walks between any pair of nodes equals*

$$M(\mathcal{W}_m) = \left(\left((u-e_m)\cdot(u-e_m)^T\right)\circ A\right)^{N-1},$$

where e_i denotes the $N \times 1$ basic vector with only one non-zero element $(e_i)_i = 1$.

Figure 3.7 provides two examples of walks not traversing a node.

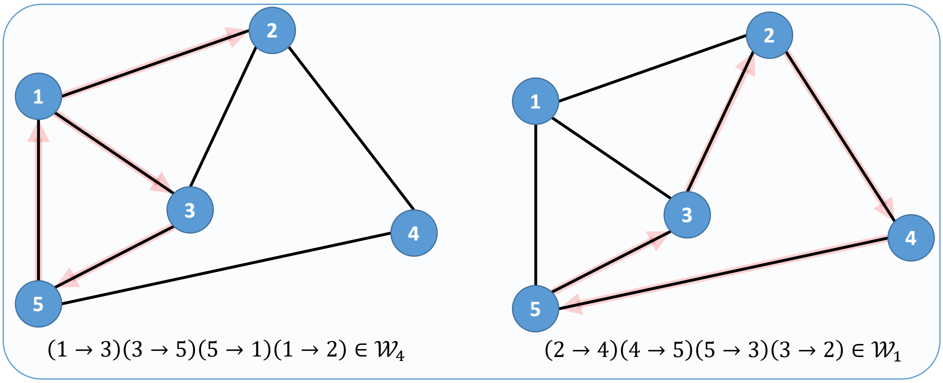


Figure 3.7: Examples of walks with length $k = N-1$ in which a node is not traversed. Traversed links are colored in red, while arrows follow labeling in the node sequence.

3.5.1. HAMILTONIAN PATH MATRIX P_{N-1}

A path of length $N-1$, also known as a Hamiltonian path, is defined by a sequence of N nodes

$$(n_0, n_1, \ldots, n_{N-1})$$

such that $n_k \neq n_l$, for $0 \leq k \leq N-1$ and for $l \in \mathcal{N} \setminus k$, where also $a_{(n_k, n_{k-1})} = 1$, for $0 \leq k \leq N-2$. Because a path by definition consists of different nodes in the sequence, a Hamiltonian path traverses each node in the graph exactly once. Such observation allowed Bax in [71] to define a set of all walks with hopcount $k = N-1$, where node $m \in \mathcal{N}$ is not traversed.

Walk sets of the form $\mathcal{W}_m$ allow us to define the set of all possible walks of length $k = N-1$ as follows

$$\mathcal{W}[N-1] = \bigcup_{i=1}^{N} \mathcal{W}_i \cup \mathcal{P}[N-1] \tag{3.22}$$

Relation (3.22) informs us that either a node is not traversed in a walk of length $N-1$ or that walk represents a path, leading to the following general solution for the $N \times N$ Hamiltonian path matrix P_{N-1}

$$P_{N-1} = A^{N-1} - M\left(\bigcup_{i=1}^{N} \mathcal{W}_i\right). \tag{3.23}$$

By applying the inclusion-exclusion formula on the set union from the equation above, Bax obtained

$$\begin{aligned}
M\left(\bigcup_{i=1}^{N}\mathcal{W}_i\right) =& \sum_{i_1=1}^{N} M\left(\mathcal{W}_{i_1}\right)\\
&-\sum_{i_1=1}^{N-1}\sum_{i_2=i_1+1}^{N} M\left(\mathcal{W}_{i_1}\cap\mathcal{W}_{i_2}\right)\\
&+\sum_{i_1=1}^{N-2}\sum_{i_2=i_1+1}^{N-1}\sum_{i_3=i_2+1}^{N} M\left(\mathcal{W}_{i_1}\cap\mathcal{W}_{i_2}\cap\mathcal{W}_{i_3}\right)\\
&-\ldots\\
&+(-1)^{N-3}\sum_{i_1=1}^{2}\sum_{i_2=i_1+1}^{3}\cdots\sum_{i_N=i_{N-1}+1}^{N} M\left(\bigcap_{z=1}^{N}\mathcal{W}_{i_z}\right).
\end{aligned} \tag{3.24}$$

A set of intersections from (3.24) defines all possible walks with hopcount $N-1$, where multiple nodes are not traversed. The corresponding $N\times N$ walk matrix of such a walk set is

$$M\left(\mathcal{W}_{i_1}\cap\mathcal{W}_{i_2}\cap\cdots\cap\mathcal{W}_{i_m}\right) = \left(\left(\operatorname{diag}\left(u-e_{i_1}-e_{i_2}-\ldots e_{i_m}\right)\right)\cdot A\cdot\left(\operatorname{diag}\left(u-e_{i_1}-e_{i_2}-\ldots e_{i_m}\right)\right)\right)^{N-1}. \tag{3.25}$$

By combining (3.23) and (3.25) Bax derived in [71] the $N\times N$ Hamiltonian path matrix P_{N-1} as follows

$$\begin{aligned}
P_{N-1} =& A^{N-1} - \sum_{i_1=1}^{N}\left(\operatorname{diag}\left(u-e_{i_1}\right)\cdot A\cdot\operatorname{diag}\left(u-e_{i_1}\right)\right)^{N-1}\\
&+\sum_{i_1=1}^{N-1}\sum_{i_2=i_1+1}^{N}\left(\operatorname{diag}\left(u-e_{i_1}-e_{i_2}\right)\cdot A\cdot\operatorname{diag}\left(u-e_{i_1}-e_{i_2}\right)\right)^{N-1}\\
&-\ldots\\
&+(-1)^{N-3}\cdot\sum_{i_1=1}^{2}\sum_{i_2=i_1+1}^{3}\cdots\sum_{i_{N-1}=i_{N-2}+1}^{N}\left(\operatorname{diag}\left(u-\sum_{z=1}^{N-1}e_{i_z}\right)\cdot A\cdot\operatorname{diag}\left(u-\sum_{z=1}^{N-1}e_{i_z}\right)\right)^{N-1}.
\end{aligned} \tag{3.26}$$

3.5.2. ANALYTIC SOLUTION FOR THE $N\times N$ PATH MATRIX P_k

We here extend the approach of Bax in [71] and derive an analytic solution for the $N\times N$ path matrix P_k of any hopcount $1\le k\le N-1$. The idea behind computing the number of paths with hopcount k between node pairs is to examine all possible unlabeled sequences of $k+1$ nodes, in total $\binom{N}{k+1}$ of them. For each node sequence, we remove links from the graph, not adjacent to any node in the sequence. A path of length k in such a reduced graph is equivalent to a Hamiltonian path in the original graph, and thus, the idea of Bax from [71] can be applied.

Theorem 14 *The $N\times N$ path matrix P_k, whose entries comprise the number of paths with* *hopcount k between any pair of nodes can be computed as follows*

$$
\begin{aligned}
P_k = &\sum_{i_0=0}^{N-k-1}\sum_{i_1=i_0+1}^{N-k}\cdots\sum_{i_k=i_{k-1}+1}^{N}\left[\left(\left(\left(\sum_{z=0}^{k} e_{i_z}\right)\cdot\left(\sum_{z=0}^{k} e_{i_z}^T\right)\right)\circ A\right)^k\right.\\
&-\sum_{j_0=0}^{k}\left(\left(\left(\sum_{z=0}^{k} e_{i_z}-e_{i_{j_0}}\right)\cdot\left(\sum_{z=0}^{k} e_{i_z}-e_{i_{j_0}}\right)^T\right)\circ A\right)^k\\
&+\sum_{j_0=0}^{k-1}\sum_{j_1=j_0+1}^{k}\left(\left(\left(\sum_{z=0}^{k} e_{i_z}-e_{i_{j_0}}-e_{i_{j_1}}\right)\cdot\left(\sum_{z=0}^{k} e_{i_z}-e_{i_{j_0}}-e_{i_{j_1}}\right)^T\right)\circ A\right)^k\\
&-\ldots\\
&\left.+(-1)^{k-2}\sum_{j_0=0}^{1}\sum_{j_1=j_0+1}^{2}\cdots\sum_{j_{k-1}=j_{k-2}+1}^{k}\left(\left(\left(\sum_{z=0}^{k} e_{i_z}-\sum_{q=0}^{k-1} e_{i_{j_q}}\right)\cdot\left(\sum_{z=0}^{k} e_{i_z}-\sum_{q=0}^{k-1} e_{i_{j_q}}\right)^T\right)\circ A\right)^k\right].
\end{aligned}
\tag{3.27}
$$

Proof We examine all possible unlabeled sequences of $k+1$ nodes. For each such a sequence $(i_0, i_1, \ldots, i_k)$ we transform the $N\times N$ adjacency matrix A by removing all links not adjacent to any node in the sequence $\left(\left(\sum_{z=0}^{k} e_{i_z}\right)\cdot\left(\sum_{z=0}^{k} e_{i_z}^T\right)\right)\circ A$. The modified adjacency matrix allows for applying the inclusion exclusion formula Bax derived in [71], because a path of length k in the modified adjacency matrix is equivalent to a Hamiltonian path in the $N\times N$ original adjaceny matrix A. □

There are $\binom{N}{k+1} = \frac{N!}{(k+1)!(N-k-1)!}$ ways to choose $k+1$ nodes out of N nodes. For each set of $k+1$ nodes, relation (3.27) defines in total 2^k matrix terms[1] and thus computing the $N\times N$ path matrix P_k implies complexity $O\left(\binom{N}{k+1} kN^3 2^k\right)$.

3.5.3. COMPLEXITY OF COMPUTING THE $N\times N$ PATH MATRIX P_k

We present three analytic solutions for the $N\times N$ path matrix P_k, comprising in its entries the number of length k paths between node pairs. Figure 3.8 provides complexity of computing the $N\times N$ path matrix P_k, as a function of the hopcount k, for a graph of $N = 20$ nodes. Complexity of the solution in (3.6) (blue color), based on walks traversing a node multiple times, represents the most complex approach for almost the entire range of hopcount k values. Despite its complexity, for small k, the solution in (3.6) is the most insightful, from a linear algebra point of view.

Computing the $N\times N$ path matrix P_k using (3.21) (red color in Figure 3.8) requires the least computational effort, for smaller values of the hopcount k, because it examines all possible labeled sequences of $k+1$ nodes. In contrast, for smaller values of hopcount k, the third approach in (3.27) (presented in green color in Figure 3.8)) is far more computationally demanding, because it applies the inclusion exclusion formula on each possible unlabeled sequence of $k+1$ nodes. As the path length k increases, there are less unlabeled sequences of nodes, allowing the third solution in (3.27) to perform the best, in terms of complexity.

[1]Each matrix term in (3.27) represent the k-th power of the adjacency matrix with reduced number of links. Therefore, complexity of computing a matrix term is $O(kN^3)$.

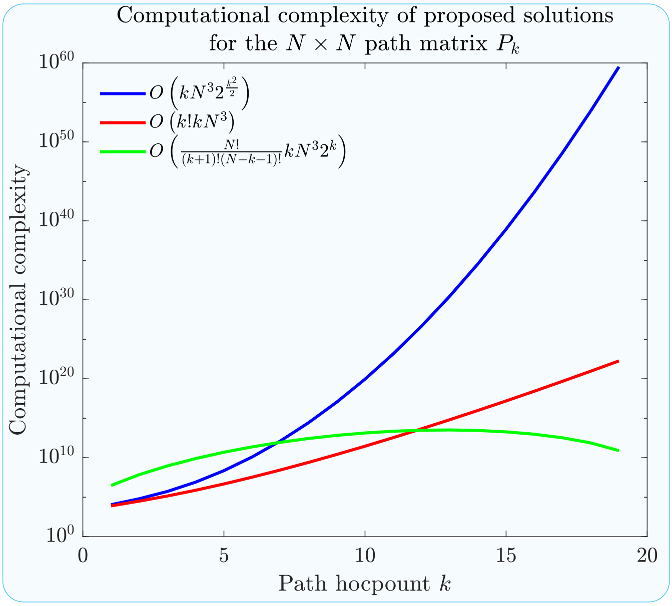


Figure 3.8: Computational complexity of computing the $N \times N$ path matrix P_k with different hopcount $1 \le k \le N-1$ using (3.6) (blue color), (3.21) (red color) and (3.27)(green color), where $N = 20$.

3.6. RECURSIVE ALGORITHM FOR COMPUTING THE NUMBER OF PATHS

We provide a simple recursive algorithm 3.10 that computes the $N \times N$ path matrices P_k, where $1 \le k \le N-1$. The proposed algorithm treats each path independently (via recursions) and prevents repeating nodes in the walk sequence.

The proposed recursive algorithm identifies each possible path in a graph and increment the corresponding element of the $N \times N$ path matrix P_k, with $1 \le k < N$. For each node $i \in \mathcal{N}$ in G, we set the hopcount to 0 and call the recursive procedure, as provided in line 4 of the pseudocode 3.9. The recursive algorithm 3.10 returns the $N-1 \times N$ node i based path matrix T, where the element T_{jm} denotes the number of length j paths between node i and node m. Therefore, in line 6 we store the j-th row of the $N-1 \times N$ node based path matrix T as the i-th row of the $N \times N$ path matrix P_j.

The recursive procedure described in Algorithm 3.10 takes the $(N-1) \times N$ node-based path matrix T, the $N \times N$ adjacency matrix A, the destination node n_k and the current hop count k as inputs. Firstly, we increment the hop count k in line 1. Next, in line 2, we identify the neighbors of the destination node $j \in \mathcal{N}_{n_k}$. In the subsequent step, we account for the paths reaching any neighbors in $\mathcal{N}_{n_k}$. A crucial step is to remove all links adjacent to the destination node n_k, as defined in line 4, to prevent paths from reaching node n_k again. Therefore, within the recursive function, we remove all links that would lead to the reappearance of a node, allowing us to consider each adja-

DeterminePaths (A, N)

Input: A, N
Output: $P_1, P_2, \ldots, P_{N-1}$
$T \leftarrow O_{(N-1)\times N}$
$P_k \leftarrow O_{N\times N}$, where $1 \leq k < N$
for $i \leftarrow 1$ to N
    $T \leftarrow$ ComputePaths ($O_{(N-1)\times N}$, A, i, 0)
    for $j \leftarrow 1$ to $N-1$
        Store j-th row of T as the i-th row of P_j
    end for
end for
return $P_1, P_2, \ldots, P_{N-1}$

Figure 3.9: Pseudocode for calling the recursive Algorithm for determining all paths in a graph, with the graph size N and the the $N \times N$ adjacency matrix A as input.

ComputePaths (T, A, n_k, k)

Input: T, A, n_k, k
Output: T
$k \leftarrow k+1$
$\mathcal{N}_{n_k} \leftarrow \{j \,|\, a_{n_k,j} = 1, j \in \mathcal{N}\}$
$T_{k,j} \leftarrow T_{k,j} + 1$, where $j \in \mathcal{N}_{n_k}$
$a_{n_k,j} \leftarrow 0$ and $a_{j,n_k} \leftarrow 0$, where $j \in \mathcal{N}_{n_k}$
for $m \leftarrow 1$ to $|\mathcal{N}_{n_k}|$
    if $|\mathcal{N}_{j_m}| > 0$
        $T \leftarrow$ ComputePaths (T, A, j_m, k)
    end if
end for
return T

Figure 3.10: Metacode of the recursive algorithm for determining all paths in a graph, originating from a singe node, with the $N \times N$ adjacency matrix A, the $(N-1) \times N$ node-based path matrix T, destination node n_k and hopcount k as input. The recursive function returns the $(N-1) \times N$ node-based path matrix T.

cent node of the destination node as a valid extension of the path. After removing the links, for each neighbor j (line 5) with non-zero degree (line 6), we invoke the recursive algorithm in line 7, with the incremented hop count and the updated destination node j. The recursion terminates when a destination node has no neighbors, as defined in line 6. In appendix C.2, we adjust the proposed recursive algorithm to identify number of length k paths only, between all node pairs.

By executing Algorithm 3.10 once, we can gather information about every possible path in a graph. Since the recursive algorithm 3.10 we propose accounts for all possible paths, its complexity scales linearly with the total number of paths, given by $\frac{1}{2} \cdot \sum_{i=1}^{N-1} u^T \cdot P_i \cdot u$. Thus, Figure 3.11 illustrates the total number of paths in an Erdős-Rényi (ER) random graph with $N = 6$ (left), $N = 8$ (middle), and $N = 10$ (right) nodes, respectively, for different link densities p. It is evident that the complexity of the proposed algorithm, $O(N(1+p)^{2N})$, grows exponentially with the network size N.

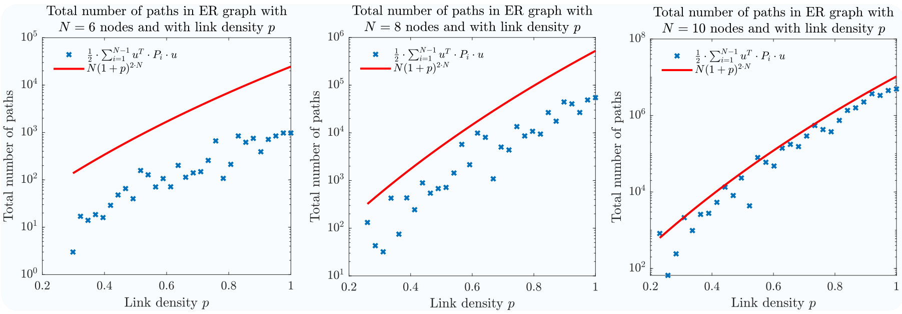


Figure 3.11: Total number of paths in Erdős Rényi graph with $N = 6$ (left figure), $N = 8$ nodes (middle figure) and $N = 10$ nodes (right figure), for different values of the link density p.

3.7. CONCLUSION

We introduce three types of walks: walks with a node reappearing in the sequence, walks traversing a node exactly once, and those not traversing a node. Based on considered walk types, we derive analytic solutions for the number of paths of a certain length between node pairs in a matrix form. Depending on the path length, different solutions require the least computational effort. We propose a recursive algorithm for determining all possible paths between node pairs, whose complexity scales linearly with the total number of paths in a graph. The proposed recursive algorithm applies to a directed (un)weighted network.

4

EFFECTIVE RESISTANCE IN GRAPH THEORY

It is not knowledge, but the act of learning,
not possession but the act of getting there,
which grants the greatest enjoyment.

Carl Friedrich Gauss

The effective resistance between two nodes is determined by the electrical systems theory, which explains how electrical energy is dissipated throughout the network while being transmitted between the nodes. This concept is significant for the general network theory since the effective resistance metric describes the entire network from the viewpoint of two nodes. In this chapter, we make use of the information captured by the effective resistance. We present an iterative algorithm that solves the inverse all-shortest-path problem by beginning with a complete graph and progressively removing links until the given upper bounds on the shortest path weights are exceeded. Furthermore, we propose an iterative algorithm for deterministic graph sparsification, which either minimises or maximises the effective graph resistance, or minimises the Laplacian eigenvalue deviation.

This chapter is partially based on [72].

4.1. Introduction

Effective resistance is a key concept in graph theory, which has significant applications in various areas, including electrical, social, and transportation networks. The effective resistance between two nodes in an electrical network is the resistance between those nodes when all other links are removed and replaced with their corresponding resistances [73]. This measure takes into account all possible paths between the two nodes and is closely related to the concept of electrical resistance in physics. Effective resistance has been extensively studied in the literature, and one of its critical properties is being a distance matrix and thus obeying the triangle inequality [4]. This property enables the estimation of effective resistances between any pair of nodes using only local information, making it useful in the analysis of complex networks. Effective resistance was first introduced by Kirchhoff in 1847 to calculate the electrical resistance of a network of resistors [74]. Since then, effective resistance has found diverse applications in several fields.

One of the earliest applications of effective resistance was in the study of random walks on graphs. The commute time between two nodes in an unweighted graph is proportional to the effective resistance between these two nodes [75]. This result has been applied in the study of diffusion processes in networks, including social and biological networks [76]. Conversely, the escape probability - the probability of a random walk starting from node i and reaching node j before returning to node i - is inversely proportional to the effective resistance between the two nodes [77]. Furthermore, the effective resistance between two nodes quantifies a ratio of spanning trees in a graph that traverses that link [78]. Effective resistance has also been used in the analysis of epidemic spreading [79]. In recent years, effective resistance has gained significant attention in network science. It has been utilized to identify important nodes or edges in complex networks [23], such as those with high effective resistance, which play a critical role in the network's overall connectivity. Effective resistance has also been applied in community detection [26], where it can identify communities of nodes with similar effective resistance properties.

This chapter introduces two applications of effective resistance in graph theory. In Section 4.2, we propose an iterative algorithm that solves the inverse all shortest path problem, while in Section 4.3, we propose a deterministic graph sparsification algorithm that removes links from an unweighted graph iteratively, while either minimising or maximising the effective graph resistance of the resulting graph. Finally, we conclude in Section 4.4.

4.1.1. The Laplacian matrix Q

The eigenvalue decomposition of the $N \times N$ Laplacian $Q = \Delta - A$,

$$Q = Z \cdot \text{diag}(\mu) \cdot Z^T, \tag{4.1}$$

defines the set of N orthogonal $N \times 1$ eigenvectors z_i contained in columns of the $N \times N$ eigenvector matrix Z and N eigenvalues $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_N$. Due to double orthogonality of the eigenvector matrix Z (i.e. $Z \cdot Z^T = I$ and $Z^T \cdot Z = I$), where I is the $N \times N$ identity matrix, relation (4.1) can be transformed into a weighted sum of N outer vector products

$$Q = \sum_{i=1}^{N} \mu_i \cdot z_i \cdot z_i^T. \tag{4.2}$$

As of any real, symmetric matrix [80], the eigenvalues of Laplacian Q are real and non-negative because Q is a positive semidefinite matrix [80, p.67]. From $Q \cdot u = 0$, we observe that $\mu_N = 0$ and $z_N = u$ and thus $\det Q = 0$. Consequently, the Laplacian Q is not invertible. However, the pseudoinverse[1]

$$Q^\dagger = \sum_{i=1}^{N-1} \frac{1}{\mu_i} \cdot z_i \cdot z_i^T \tag{4.3}$$

obeys $Q^\dagger \cdot Q = Q \cdot Q^\dagger = I - \frac{1}{N} \cdot J$. In this work we consider a weighted graph G, where a link l between node i and node j is defined by its weight

$$w_{ij} = w_l = \frac{1}{r_l},$$

with $r_l > 0$ denoting link l resistance.

4

4.1.2. EFFECTIVE RESISTANCE

The effective resistance ω_{ij} between node i and node j is defined as

$$\omega_{ij} = \left(e_i - e_j\right)^T \cdot Q^\dagger \cdot \left(e_i - e_j\right), \tag{4.4}$$

where the $N \times 1$ basic vector e_i has only one non-zero element $(e_i)_i = 1$. The effective resistance ω_{ij} quantifies the dissipated power when the current of 1 Ampere is applied between the nodes i and j. Relation (4.4) can be transformed into a matrix form, defining the $N \times N$ effective resistance matrix

$$\Omega = \zeta \cdot u^T + u \cdot \zeta^T - 2 \cdot Q^\dagger, \tag{4.5}$$

where the $N \times 1$ vector $\zeta = \left(Q^\dagger_{11}, Q^\dagger_{22}, \ldots, Q^\dagger_{NN}\right)$ contains the diagonal elements of the pseudoinverse of Laplacian $Q^\dagger$. The effective resistance ω_{ij} between directly connected nodes i and j (i.e. $a_{ij} = 1$), represents the effective resistance of a parallel connection

$$\frac{1}{\omega_{ij}} = \frac{1}{r_{ij}} + \frac{1}{(\omega_{G^*})_{ij}} \tag{4.6}$$

between the resistance of a direct link r_{ij} and the effective resistance $(\omega_{G^*})_{ij}$ between nodes i and j in the graph $G^* = G \setminus l_{ij}$, where the link l_{ij} is removed.

Lemma 15 *A link $l_{ij} \in \mathcal{L}$ of a graph $G(\mathcal{N}, \mathcal{L})$ connects two disconnected sub-graphs G_1 and G_2, i.e. $\mathcal{L}(G_1) \cup \mathcal{L}(G_2) \cup l_{ij} = \mathcal{L}(G)$ and $\mathcal{L}(G_1) \cap \mathcal{L}(G_2) = \emptyset$ if and only if it holds*

$$\omega_{ij} = r_{ij}.$$

[1] We restrict the analysis to connected graphs, as the number of zero eigenvalues of Laplacian Q equals the number of connected components in a graph. More precisely, relation (4.3) does not hold in the case of a disconnected graph.

Proof: In case link l_{ij} of a graph G connects two disconnected sub-graphs G_1 and G_2, the effective resistance of a graph $G^* = G \setminus l_{ij}$ equals $r^*_{ij} = \infty$. Therefore, relation (4.6) transforms into $\omega_{ij} = r_{ij}$, which completes the proof. □

The effective resistance ω_{ij} between adjacent nodes i and j is upper bounded by the resistance r_{ij} of the direct link between them

$$\omega_{ij} = \frac{r_{ij} \cdot (\omega_{G^*})_{ij}}{r_{ij} + (\omega_{G^*})_{ij}} \le \min\left(r_{ij}, (\omega_{G^*})_{ij}\right).$$

Otherwise, when $a_{ij} = 0$, the effective resistance ω_{ij} is upper bound by the sum of resistances of links forming the shortest path between the nodes. In both cases, if more paths exist connecting two nodes, then there are more possible paths for the current to flow simultaneously and thus, the effective resistance lowers. The sum of all elements of the $N \times N$ effective resistance matrix Ω defines the effective graph resistance

4

$$R_G = \frac{1}{2} \cdot u^T \cdot \Omega \cdot u = N \cdot \sum_{i=1}^{N-1} \frac{1}{\mu_i}. \tag{4.7}$$

4.2. INVERSE ALL SHORTEST PATHS PROBLEM

Problem 16 (Inverse All Shortest Path Problem (IASPP)) *Given an $N \times N$ symmetric demand matrix D with zero diagonal elements but positive off-diagonal elements. Determine an $N \times N$ weighted adjacency matrix $\widetilde{A}$, such that*[2] *the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$*

Since an element in the shortest path weight matrix S can be any positive number by scaling the weighted adjacency matrix, the IASPP generally has infinitely many solutions. One possibility is to add optimisation criteria, such that the IASPP asks to determine an $N \times N$ weighted adjacency matrix under the constraints that the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$ and minimizes a norm $||D - S||$. This instance of IASPP is called [27] the optimized inverse shortest path problem (OIASPP)[27].

Problem 17 (Optimized Inverse Shortest Path Problem (OIASPP)) *Given an $N \times N$ symmetric demand matrix D with zero diagonal elements but positive off-diagonal elements. Determine an $N \times N$ weighted adjacency matrix $\widetilde{A}$, such that the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$ and minimizes a norm $||D - S||$.*

Any topology resulting in a connected graph (i.e. from a tree graph to the complete graph [27]) can represent the solution of the IASPP problem 16, with appropriate link weights. Instead, in the following part of the chapter, we consider a variation of the IASPP problem, where the total link budget $b = \sum_{l \in \mathcal{L}} w_l$ is fixed.

Problem 18 (Inverse Shortest Path Problem with Link Budget (IASPP$_B$)) *Given an $N \times N$ symmetric demand matrix D with zero diagonal elements but positive off-diagonal elements and a positive link budget b. Determine an $N \times N$ weighted adjacency matrix $\widetilde{A}$ with the least number of links L, such that $u^T \cdot W \cdot u = 2b$ and the corresponding shortest path weight matrix S obeys $S \preccurlyeq D$.*

[2] The notation $\preccurlyeq$ is used for componentwise inequality, i.e. $S \preccurlyeq D$ means that $s_{ij} \le d_{ij}$ for each $i = 1, 2, \ldots, N$ and each $j = 1, 2, \ldots, N$.

With the total link budget b introduced, not each graph topology solves the IASPP problem. When the number of links L_H in the obtained graph H is reduced, the shortest path weights increase on average because the link weights always sum to b.

4.2.1. OMEGA-BASED LINK REMOVAL (OLR)

A shortest path weight between two nodes is the sum of link weights (i.e. corresponding elements of the $N \times N$ weighted adjacency matrix W) belonging to that path. On the contrary, from the electric graph theory point of view, by summing the link weights we sum the inverse resistance of each link forming the path[3] Therefore, to utilise the analogy between shortest paths and effective resistance, we additionally define the $N \times N$ matrix $\hat{W}$ containing the inverse link weights

$$\hat{w}_{ij} = \begin{cases} \frac{1}{w_{ij}} & \text{if } w_{ij} > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{4.8}$$

where $i, j \in \mathcal{N}$. The corresponding $N \times N$ effective resistance matrix computed using $\hat{W}$ instead of W is denoted as $\hat{\Omega}$.

In Figure 4.1, we propose an iterative algorithm that solves the IASPP_B problem by utilising the information contained in the effective resistance between pairs of nodes. The OLR algorithm is initialised with the complete graph in line 2 (with the adjacency matrix $A = J - I$), while the link weights equal (line 3) corresponding shortest path weights in the $N \times N$ demand matrix D, scaled to sum up to b,

$$W = \frac{b}{u^T \cdot (A \circ D) \cdot u} \cdot (A \circ D),$$

for two reasons. Firstly, if the proposed OLR algorithm recovers the exact topology as in the original graph, the link weights would also be the same. Secondly, when additional links exist in the obtained graph H, their weights exist at the cost of reduced weights of links from the original graph G, thus still satisfying the bound $S \preccurlyeq D$. To determine which link should be removed in each iteration, in line 7, we compute the $N \times N$ matrix

$$R = \left(\hat{\Omega} - \hat{W}\right) \circ (D - S) \circ A,$$

whose elements are dimensionless and denote the inverse effective resistance $\left(\hat{\Omega} - \hat{W}\right)_{ij}$ between a pair of neighbouring nodes (i.e. $a_{ij} = 1$), in case the direct link between them is removed (as in (4.6)), multiplied by the gap $(d_{ij} - s_{ij})$ between the shortest path weight between them and the given upper bound in D. We remove the existing link with the highest value in R (line 8) because the adjacent nodes are easily reachable via the rest of the graph when the link is removed, and the margin between the current shortest path weight and the upper bound is relatively high. After updating the adjacency matrix A (line 9), we redistribute the link weights (line 10) as $W = \frac{b}{u^T \cdot (A \circ D) \cdot u} \cdot (A \circ D)$ and update (line 11) the $N \times N$ shortest path weight matrix S.

Link removal is performed until at least one shortest path weight in the obtained graph H exceeds the given upper bound in the $N \times N$ demand matrix D. At that point,

[3] A link weight in the electric graph theory defines the inverse resistance of that link.

OLR(D, b)

Input: D, b
Output: W, S
1. $N \leftarrow$ number of rows (or columns) in D
2. $A_{N\times N} \leftarrow J_{N\times N} - I_{N\times N}$ adjacency matrix of a complete graph
3. $W \leftarrow \frac{b}{u^T\cdot(A\circ D)\cdot u}\cdot(A\circ D)$ weighted adjacency matrix
4. $S_{N\times N} \leftarrow$ shortest path weight matrix of W
5. **do**
6. $\quad\Omega_{N\times N} \leftarrow$ effective resistance matrix of $\hat{W}$
7. $\quad R \leftarrow \left(\hat{\Omega} - \hat{W}\right)\circ(D-S)\circ A$
8. $\quad(i,j) \leftarrow$ indices of the maximum element in R
9. $\quad A \leftarrow A - e_i\cdot e_j^T - e_j\cdot e_i^T$
10. $\quad W \leftarrow \frac{b}{u^T\cdot(A\circ D)\cdot u}\cdot(A\circ D)$
11. $\quad S_{N\times N} \leftarrow$ shortest path weight matrix of W
12. **while** $(S \preccurlyeq D)\wedge(R_{ij}>0)$
13. $A \leftarrow A + e_i\cdot e_j^T + e_j\cdot e_i^T$
14. $W \leftarrow \frac{b}{u^T\cdot(A\circ D)\cdot u}\cdot(A\circ D)$
15. $S_{N\times N} \leftarrow$ shortest path weight matrix of W
16. **return** W, S

Figure 4.1: Pseudocode of the proposed OLR algorithm. For a given $N\times N$ symmetric demand matrix D, the algorithm returns the $N\times N$ weighted adjacency matrix W and the $N\times N$ corresponding shortest path weight matrix S of the recovered graph G, obeying $S \preccurlyeq D$ and $u^T\cdot W\cdot u = 2b$.

the last removed link is returned (line 13), while the $N\times N$ weighted adjacency matrix W and the $N\times N$ corresponding shortest path weight matrix S of the obtained graph H are provided as output (lines 14-16).

In the following subsection, we compare the performance of our OLR algorithm to that of the DOR algorithm proposed in [72]. The DOR algorithm assumes a complete graph with the link weights as provided in the demand matrix D and computes the minimum spanning tree. Iteratively, DOR adds links between those nodes whose shortest path weight is below the upper bound, provided in D, by assigning the given shortest path weight in D as the link weight. Eventually, DOR outputs a weighted graph whose shortest path weights exactly equal given bounds in the demand matrix D.

4.2.2. SIMULATION RESULTS

In this section, we generate Erdős–Rényi (ER) random graphs $G_p(N)$, with a different number of nodes N and a different link density p. We uniformly assign a random weight to each link in G, thus defining the $N\times N$ link weight matrix W and determining the total link budget $b = \frac{1}{2}\cdot u^T\cdot W\cdot u$. For each generated ER graph, we provide the $N\times N$ shortest path weight matrix D and the link budget b to the algorithm DOR and OLR and obtain the resulting graph H, whose $N \times N$ shortest path weight matrix is denoted as S.

To test the performance of the proposed inverse shortest path algorithms, we introduce four complementary criteria: (i) the number $L_H - L_G$ of additional links in the resulting graph H, (ii) the number $\frac{1}{2} \cdot u^T \cdot (A \circ A_H) \cdot u$ of common links in the original graph G and the resulting graph H and (iii) the norm $\frac{1}{2} \sum_i \sum_j \frac{d_{ij} - s_{ij}}{d_{ij}}$ of the demand matrix D and the shortest path weight matrix S and (iv) the differences $\frac{b_H - b}{b}$ of total budgets between G and H. For each number of nodes N and different link density p, we execute 100 times simulations and calculate the average of each criterion.

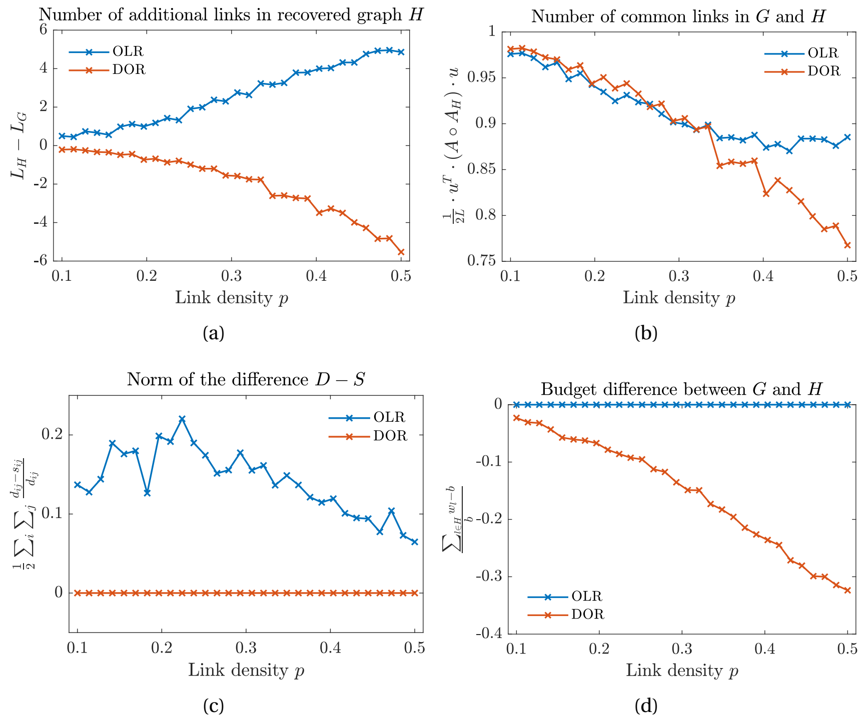


Figure 4.2: Performance of the DOR and OLR algorithm on ER graphs with $N = 10$ nodes and different link density p.

Figure 4.2 illustrates the results for ER graphs of $N = 10$ nodes for DOR (red line) and OLR (blue line). Figure 4.2(a) depicts the difference in the number of links $L_H - L_G$ between the recovered graph H and the original graph G. For a small link density p, the recovered graph H contains almost the same number of links L_H as that of the original graph L_G. The difference in the number of links $L_H - L_G$ increases linearly with the link density p when H is obtained by OLR, while $L_H - L_G$ decreases for DOR. Figure 4.2(b) informs us that the percentage of links in G also recovered in H obtained by OLR, is very high overall while almost linearly decaying with the link density p. When we recover graph H using DOR, the percentage of common links between the original graphs G

and the resulting graph H decreases with link density p increasing because the resulting graph H generally has fewer links than the original graph G with a higher link density. Combining insights from Figures 4.2(a) and 4.2(b), we observe that in the case of a sparse graph G, two algorithms can nearly recover the same graph most of the time.

Figure 4.2(c) illustrates the norm of the matrix $D-S$, revealing an interesting property of the IASPP problem. Namely, the norm $\frac{1}{2}\sum_i\sum_j \frac{d_{ij}-s_{ij}}{d_{ij}}$ is minimised for both sparse and dense graphs when using OLR. However, in the case of a dense graph, the recovered graph H contains a considerably smaller portion of original links and overall more links than in G. Still, the matrix $D-S$ norm is minimised, informing us that there are multiple different topologies, with the shortest path weights between node pairs, as in the original graph G. DOR always achieves the norm $\frac{1}{2}\sum_i\sum_j \frac{d_{ij}-s_{ij}}{d_{ij}} = 0$.

4

We present the difference of total link budgets $\frac{1}{2}u^T W_H u - b$ in Figure 4.2(d). For DOR, the difference between the total budget of graph H and b decreases with the increment of link density p, while the graph H obtained by WDOR always has a lower total budget because WDOR has fewer links than graph H. In contrast, OLR has a fixed link budget equaling b.

4.3. DETERMINISTIC GRAPH SPARSIFICATION

Spielman [24] proposed a stochastic method for graph sparsification, which employs effective resistance. In this section, we suggest a deterministic technique for graph sparsification, which utilises effective resistance.

Relation (4.6) allows us to formulate an iterative procedure of removing L_r links from a graph, such that the increase[4] in the effective graph resistance R_G is minimised (or maximised). We propose the OGS (Omega-based graph sparsification) - an iterative algorithm for removing links from a graph by minimising the effective graph resistance R_G, outlined in Figure 4.3.

OGS(A)

Input: A **the** $N \times N$ **adjacency matrix of a graph** G
Output: A **the** $N \times N$ **adjacency matrix of a sparsified graph** H
1. $\Omega \leftarrow$ the $N \times N$ effective resistance matrix of G
2. $R \leftarrow A \circ \left(\hat{\Omega} - A\right)$
3. $(i, j) \leftarrow$ indices of the maximum element in R
4. **return** $A \leftarrow A - e_i \cdot e_j^T - e_j \cdot e_i^T$

Figure 4.3: Pseudocode of the algorithm for removing a link from a graph G, by minimising the effective graph resistance R_G.

For a given graph G, defined with the $N \times N$ adjacency matrix A, we compute the

[4] Link removal from a graph causes a strictly larger effective graph resistance, i.e. $R_G\left(G\left(\mathcal{N}, \mathcal{L} \setminus (i,j)\right)\right) > R_G(G)$, where $(i,j) \in \mathcal{L}$.

$N \times N$ effective resistance matrix Ω in line 1, as in (4.5). In the following line, we compute the $N \times N$ matrix R, representing the inverse of the effective resistance (i.e. the effective conductance) between adjacent nodes in G, in case that direct link is removed, as in (4.6). We identify the link with the maximum value in R (line 3) and remove that link from the graph G (line 4). We denote the proposed algorithm for removing links while minimising R_G as OGS (Omega-based Graph Sparsification). The reasoning behind OGS algorithm in Figure 4.3 is twofold:

- Lemma 15 teaches us that when $a_{ij} = \omega_{ij} = 1$, the link $(i \sim j)$ cannot be removed, as the resulting graph would become disconnected. Therefore, by performing line 2 of the pseudocode in Figure 4.3 and considering only positive elements in R, we guarantee that the graph H remains connected.

- We choose link $(i \sim j)$ with the maximum value in R to remove from the graph. Directly connected node pair $(i \sim j)$ with high value R_{ij} is relatively easily reachable even when the link $(i \sim j)$ is removed because a large value R_{ij} indicates multiple alternative paths between nodes i and j, via the rest of the network.

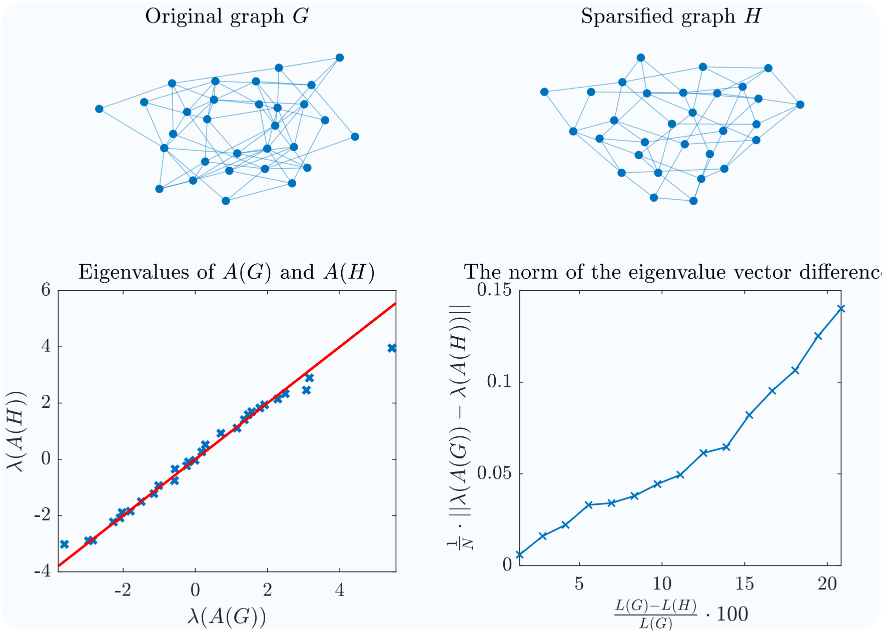


Figure 4.4: A graph G with $N = 30$ nodes and $L = 72$ links (upper left-hand side). A sparsified graph H with $L(H) = 57$ link, using OGS algorithm 4.3 (upper right-hand side). Correlation between eigenvalues of the adjacency matrix of the graphs G and H (lower left-hand side), where the red line illustrates the perfect correlation. The norm of the eigenvalue vector difference $\lambda(A(G)) - \lambda(A(H))$ between the original graph G and the sparsified graph H, for a different number of removed links $L(G) - L(H)$ (lower right-hand side).

In Figure 4.4, we illustrate the results of a graph sparsification using the proposed iterative algorithm variant a. From a graph G (upper left-hand side) consisting of $N = 30$ and $L = 72$ links, we remove 15 links and obtain the sparsified graph H, whose topology is depicted in the upper right-hand side of Figure 4.4. The eigenvalues of the adjacency matrix of the sparsified graph H slightly deviate from those of the adjacency matrix $A(G)$ of the original graph G, as depicted in the lower left part. Moreover, the norm of the eigenvalue vector difference $\lambda(A(G)) - \lambda(A(H))$ scales linearly with the number of removed links, as shown in the lower right-hand part of Figure 4.4.

4.3.1. EFFECTIVE GRAPH RESISTANCE MINIMISATION (MAXIMISATION) UNDER LINK REMOVAL

4

Another application of the proposed algorithm in Figure 4.3 is to obtain graph topology for a fixed number of nodes N and links L, such that the effective graph resistance R_G is minimised (or maximised). We, therefore, introduce two other variants for choosing a link based on effective resistance and denote these as OGSpath and OGSstar[5], respectively. OGSpath algorithm is outlined in Figure 4.5. We firstly compute the $N \times N$ matrix $R = \left(\text{diag}(\zeta) \cdot A + A \cdot \text{diag}(\zeta)\right) \circ \left(\hat{\Omega} - A\right)$ in line 2. The inverse of the effective resistance between adjacent nodes i and j, in case the direct link is removed (see (4.6)), is scaled by $(\zeta_i + \zeta_j)$, because ζ_i quantifies how well node i is connected to the rest of the network, as explained in [23]. OGSpath chooses a link connecting two hardly reachable nodes via the rest of the network. Therefore, instead of choosing the maximum element, as in line 3 of OGS in Figure 4.3, we identify a link with the minimum positive value in R (line 3).

On the contrary, OGSstar algorithm removes an existing link with the lowest positive value in the $N \times N$ matrix $R = \left(\text{diag}(\zeta)^{-1} \cdot A + A \cdot \text{diag}(\zeta)^{-1}\right) \circ \left(\hat{\Omega} - A\right)$. While algorithms OGS and OGSstar minimise the effective graph resistance R_G at each step, it is maximised in OGSpath.

OGSPATH(A)

Input: A **the** $N \times N$ **adjacency matrix of a graph** G
Output: A **the** $N \times N$ **adjacency matrix of a graph, after removing a link**
1. $\Omega \leftarrow$ the $N \times N$ effective resistance matrix of G
2. $R \leftarrow \left(\text{diag}(\zeta) \cdot A + A \cdot \text{diag}(\zeta)\right) \circ \left(\hat{\Omega} - A\right)$
3. $(i, j) \leftarrow$ indices of the minimum positive element in R
4. **return** $A \leftarrow A - e_i \cdot e_j^T - e_j \cdot e_i^T$

Figure 4.5: Pseudocode of the OGSpath algorithm for removing a link from a graph G, by maximising the effective graph resistance R_G.

Figure 4.7 presents an example wherein we commence with a complete graph (i.e. the $N \times N$ adjacency matrix $A = J - I$) with $N = 30$ nodes, apply OGS, OGSpath and

[5]The sparsification algorithm OGSpath (OGSstar), when applied to a complete graph, eventually leads to a path (star) topology, hence the name.

OGSSTAR(A)

Input: A **the** $N \times N$ **adjacency matrix of a graph** G
Output: A **the** $N \times N$ **adjacency matrix of a graph, after removing a link**
1. $\Omega \leftarrow$ the $N \times N$ effective resistance matrix of G
2. $R \leftarrow \left(\text{diag}(\zeta)^{-1} \cdot A + A \cdot \text{diag}(\zeta)^{-1}\right) \circ \left(\hat{\Omega} - A\right)$
3. $(i, j) \leftarrow$ indices of the minimum positive element in R
4. **return** $A \leftarrow A - e_i \cdot e_j^T - e_j \cdot e_i^T$

Figure 4.6: Pseudocode of the OGSstar algorithm for removing a link from a graph G, by minimising the effective graph resistance R_G.

OGSstar, and eliminate one link at a time until we achieve a tree graph. The OGS algorithm preserves the graph's regular configuration while sparsifying it, while OGSpath and OGSstar rapidly adopt path-like and star-like topologies.

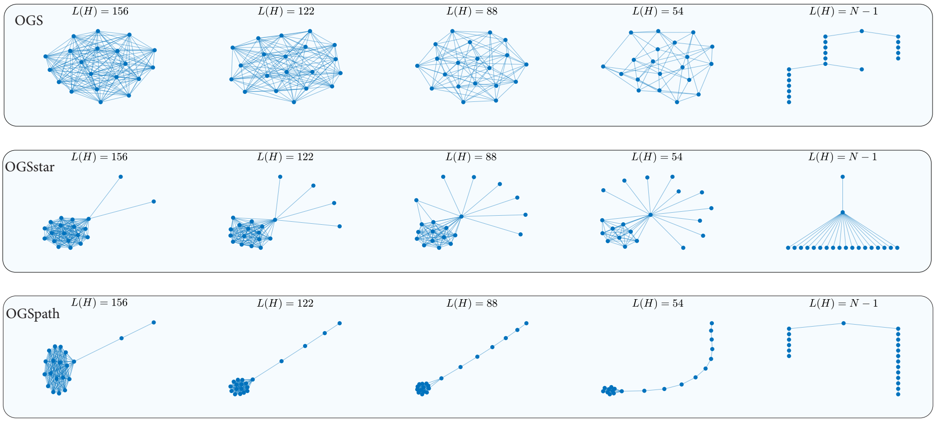


Figure 4.7: Graph H topology with $N = 30$ nodes and a different number of links $L(H)$ obtained by removing links from the complete graph G of $N = 30$ nodes using the OGS (upper part), OGSstar (middle part) and OGSpath (lower part) algorithm, outlined in Figures 4.3, 4.6 and 4.5, respectively.

For a different number of links $L(H)$ in the sparsified graph H, we compute the effective graph resistance R_G for each proposed algorithm and present it in Figure 4.8. OGSpath algorithm eliminates links adjacent to a node until the degree of that node reduces to one. Then OGSpath continues with links of the remaining neighbour while preserving the connected topology. Eventually, the graph topology reduces to a path graph with the largest possible effective graph resistance R_G of any connected undirected graphs. As demonstrated in Figure 4.8, the effective graph resistance R_G of the OGSpath algorithm exhibits a stair-like pattern that corresponds to different nodes. Moving from left to right, the length of stairs decreases while the spike between stairs increases. The reduction in the length of stairs[6] results from the removal of adjacent links that reduces the degree of each node after each stair. In contrast, the increase in the spike between stairs[7]. is attributed to the graph's sparser topology, which significantly affects the shortest paths and, thus, the effective resistance between node pairs. Similarly, the OGSstar algorithm removes links adjacent to a node until only one neighbour remains. At that point, the algorithm selects another node, and the only neighbour of the first node becomes the sole neighbour of each other node in the graph, resulting in a star topology when $L(H) = N - 1$.

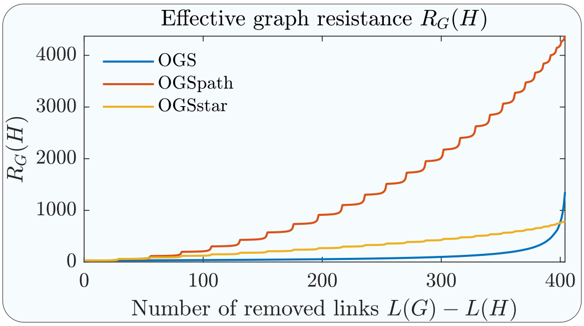


Figure 4.8: The effective graph resistance $R_G(H)$ for a different number of removed links $L(H) - L(G)$ from the complete graph G of $N = 30$ nodes, using OGS (blue), OGSpath (red) and OGSstar (yellow) algorithms, proposed in Figure 4.3, Figure 4.5 and Figure 4.6, respectfully.

The OGS algorithm achieves the lowest effective graph resistance R_G for nearly all values of the number of links $L(H)$, except for the sparse regime. This excellent performance of the OGS algorithm in terms of effective graph resistance R_G confirms the insights gained from (4.6). Removing links from a graph reduces the number of paths between node pairs, and the effective resistance between a node pair is upper-bounded by the sum of resistances of links constituting the shortest path between the nodes. Therefore, by removing a link from the shortest path, the effective resistance increases substantially. For example, in a cycle graph, where the effective resistance between two adjacent nodes satisfies $\omega_{ij} < 1$, removing a link $(i \sim j)$ results in an increase of ω_{ij} by a factor of $N - 1$.

4.4. CONCLUSION

This chapter utilises the information captured by effective resistance between node pairs in a network by proposing an iterative algorithm to solve the inverse all shortest path problem with a fixed link budget. Our OLR algorithm performs best when the provided upper bounds on the shortest path weights between node pairs are computed for

[6] In the first stair, $N - 2$ links adjacent to a node are removed. In the second stair, additional $N - 3$ links are removed. Thus, the degree of each node reduces after each stair.

[7] As the graph topology is more sparse, removing a link affects the shortest paths and thus the effective resistance between node pairs more.

a sparse graph. Additionally, the effective resistance is used in our algorithms for deterministic graph sparsification. We demonstrate that by using effective resistance, we can identify the link connecting nodes that are relatively easily reachable via the rest of the network, even without the direct link. Depending on the optimisation criteria, which may involve either maximising (minimising) the effective graph resistance R_G or minimising the deviation in Laplacian Q eigenvalues of the resulting graph, we either remove the link or, after removing it, also scale the remaining links, respectively.

II

Linear Processes on Networks

5

Linear Clustering Process on Networks

The important thing is
to never stop questioning.

Albert Einstein

We propose a linear clustering process on a network consisting of two opposite forces: attraction and repulsion between adjacent nodes. Each node is mapped to a position on a one-dimensional line. The attraction and repulsion forces move the nodal position on the line, depending on how similar or different the neighbourhoods of two adjacent nodes are. Based on each node position, the number of clusters in a network and each node's cluster membership is estimated. The performance of the proposed linear clustering process is benchmarked on synthetic networks against widely accepted clustering algorithms such as modularity, Leiden method, Louvain method and the non-back tracking matrix. The proposed linear clustering process outperforms the most popular modularity-based methods on synthetic and real-world networks, such as the Louvain method, while possessing a comparable computational complexity.

This chapter is based on [81].

5.1. INTRODUCTION

Networks [1, 2] abound and increasingly shape our world, ranging from infrastructural networks (transportation, telecommunication, power-grids, water, etc.) over social networks to brain and biological networks. In general, a network consists of a graph or underlying topology and a dynamic process that takes place on the network. Some examples of processes on a network are percolation [82] and epidemic spreading [6, 39], that possess a phase transition [35, 83]. While most real-world processes on networks are non-linear, linearisation allows for hierarchical structuring of processes on the network [49].

The identification of communities and the corresponding hierarchical structure in real-world networks has been an active research topic for decades [8], although a single, precise definition of a community does not seem to exist [9, 10]. In network science, a community is defined as a set of nodes that share links dominantly between themselves, while a minority of links is shared with other nodes in the network. Newman proposed in [84] a spectral clustering algorithm that reveals hierarchical structure of a network, by optimising modularity, a commonly used quality function of a graph partition. Xu *et al.* proposed an efficient clustering algorithm in [85], capable of detecting clusters while differentiating between hub and outlier nodes. A heuristic, modularity-based two-step clustering algorithm, proposed by Blondel *et al.* in [14], has proved to be computationally efficient and performed among the best in the comparative study conducted in [86]. Recently, Peixoto proposed in [87] a nested generative model, able to identify nested partitions at different resolutions, which thus overcomes an existing drawback of a majority of clustering algorithms, identifying small, but well-distinguished communities in a large network. Dannon *et al.* concluded in their comparative study [88] that those clustering algorithms performing the best tend to be less computationally efficient. A class of clustering algorithms exists, that perform clustering based on a dynamic process on the network, such as a random walk [89], consensus process [90] or synchronisation [91]. We refer to [8, 92] for a detailed review on existing clustering algorithms.

Our new idea is the proposal of a linear clustering process (LCP) on a graph, where nodes move in a one-dimensional space and tend to concentrate in groups that lead to network communities and therefore solve the classical[1] community detection problem. Linear means "proportional to the graph", which is needed because the aim is to cluster the graph, and the process should only help and not distract from our main aim of clustering. A non-linear process depends intricately on the underlying graph that we want to cluster and may result in worse clustering! Our LCP leads to a new and non-trivial graph matrix W in (5.10) in Theorem 19, whose spectral decomposition is at least as good as the best clustering result, based on the non-back tracking matrix [19]. Moreover, the new graph matrix W has a more "natural" relation to clustering than the non-back tracking matrix, that was not designed for clustering initially. Finally, our resulting LCP clustering algorithm seems surprisingly effective and can compete computationally with any other clustering algorithm, while achieving generally a better result!

In Section 5.2, we introduce notations for graph partitioning and briefly review basic theory on clustering such as modularity, normalised mutual information (NMI) measure

[1] A solution of the classical (or standard) community problem consists of assigning a cluster membership to each node in a network.

and different synthetic benchmarks. We introduce the linear clustering process (LCP) on a network in Section 5.3, while the resulting community detection algorithm is described in Section 5.4 and Section 5.5. We compare the performance of our LCP algorithm with that of the non-back tracking matrix, Newman's, Leiden and the Louvain algorithm and provide results in Section 5.6, after which we conclude.

5.2. NETWORK OR GRAPH CLUSTERING

The set of neighbours of node i is denoted by $\mathcal{N}_i = \{k \mid a_{ik} = 1, k \in \mathcal{N}\}$ and the degree of node i equals the cardinality of that set, $d_i = |\mathcal{N}_i|$. The set of common neighbours of node i and node j is $\mathcal{N}_i \cap \mathcal{N}_j$, while the set of neighbours of node i that do not belong to node j is $\mathcal{N}_i \setminus \mathcal{N}_j$. The degree of a node i also equals the sum of the number of common and different neighbours between nodes i and j

$$d_i = \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| + \left|\mathcal{N}_i \cap \mathcal{N}_j\right| \tag{5.1}$$

The number of common neighbours between nodes i and j equals the ij-th element of the squared adjacency matrix

$$\left|\mathcal{N}_i \cap \mathcal{N}_j\right| = \left(A^2\right)_{ij} \tag{5.2}$$

because $(A^k)_{ij}$ represents the number of walks with k hops between node i and node j (see [51, p. 32]). From (5.1), (5.2) and $d_i = (Au)_i = (A^2)_{ii}$, we have

$$\left|\mathcal{N}_i \setminus \mathcal{N}_j\right| = \left(A^2\right)_{ii} - \left(A^2\right)_{ij}$$

and

$$\left|\mathcal{N}_i \setminus \mathcal{N}_j\right| + \left|\mathcal{N}_j \setminus \mathcal{N}_i\right| = \left(A^2\right)_{ii} + \left(A^2\right)_{jj} - 2\left(A^2\right)_{ij}$$

The latter expression is analogous to the effective resistance ω_{ij} between node i and node j,

$$\omega_{ij} = Q^\dagger_{ii} + Q^\dagger_{jj} - 2Q^\dagger_{ij}$$

in terms of the pseudoinverse $Q^\dagger_{ii}$ of the Laplacian matrix $Q = \Delta - A$ (see e.g. [23]), as derived in (4.5).

Before introducing our linear clustering process (LCP) in Section 5.3, we briefly present basic graph partitioning concepts, while the overview of the more popular clustering methods is deferred to Appendix D.1.

5.2.1. NETWORK MODULARITY

Newman and Girvan [11] proposed the modularity as a concept for a network partitioning,

$$m = \frac{1}{2L} \cdot \sum_{i=1}^{N} \sum_{j=1}^{N} \left(a_{ij} - \frac{d_i \cdot d_j}{2L}\right) \cdot \mathbf{1}_{\{i \text{ and } j \in \text{ same cluster}\}}, \tag{5.3}$$

where $\mathbf{1}_x$ is the indicator function that equals 1 if statement x is true, otherwise $\mathbf{1}_x = 0$. The modularity m compares the number of links between nodes from the same community with the expected number of intra-community links in a network with randomly connected nodes. When the modularity m close to 0, the estimated partition is as good as a random partition would be. On the contrary, a modularity m close to 1 indicates that the network can be clearly partitioned into clusters. Optimising the modularity is proven to be NP-complete [93] and approximated in [4]. Defining the $N \times N$ modularity matrix C,

$$C_{ij} = \begin{cases} 1 & \text{if nodes } i \text{ and } j \text{ belong to the same cluster} \\ 0 & \text{otherwise,} \end{cases} \tag{5.4}$$

allows us to rewrite the modularity (5.3) as a quadratic form,

$$m = \frac{1}{2L} \cdot u^T \cdot \left(A \circ C - \frac{1}{2L} \cdot (d \cdot d^T) \circ C \right) \cdot u, \tag{5.5}$$

where $\circ$ denotes the Hadamard product [94]. The number of clusters in a network is denoted by c, while the $c \times 1$ vector $n = \begin{bmatrix} n_1 & n_2 & \dots & n_c \end{bmatrix}$ defines the size of each cluster, where the number of nodes in cluster i is denoted as n_i.

5

5.2.2. NORMALISED MUTUAL INFORMATION

Danon *et al.* [88] proposed the normalised mutual information (NMI) metric, based on a confusion matrix F, whose rows correspond to the original communities, while its columns are related to estimated clusters. Therefore the element F_{ij} of the confusion matrix denotes the number of nodes in the real community i, that also belong to the estimated community j. The normalised mutual information metric between the known P_0 and the estimated partition P_e, denoted as $I_n(P_0, P_e)$, is defined in [88] as follows

$$I_n(P_0, P_e) = \frac{-2 \sum_{i=1}^{c_0} \sum_{j=1}^{c_e} F_{ij} \log\left(\frac{F_{ij} N}{F_{i.} F_{.j}}\right)}{\sum_{i=1}^{c_0} F_{i.} \log\left(\frac{F_{i.}}{N}\right) + \sum_{j=1}^{c_e} F_{.j} \log\left(\frac{F_{.j}}{N}\right)}, \tag{5.6}$$

where the known and the estimated number of clusters are denoted as c_0 and c_e, respectively, the i-th row sum of F is denoted as $F_{i.}$, while its j-th column-sum is denoted as $F_{.j}$. In case two graph partitions are identical, the corresponding NMI measure equals 1, while tending to 0 when two partitions are independent. The NMI measure has been extensively used ever since, while analysing the performance of different clustering algorithms [8].

5.2.3. BENCHMARKS

The performance of the clustering methods in this chapter are benchmarked on random graphs, generated by the Stochastic Block Model (SBM), proposed by Holland [95]. The SBM model generates a random graph with community structure, where a link between two nodes exists with different probability, depending on whether the nodes belong to the same cluster or not. We provide additional information on the stochastic block model in Appendix D.2.1.

Girvan and Newman [96] focused on a special case of the SBM model (GN benchmark), where the graph consists of $N = 128$ nodes, distributed in $c = 4$ communities of equal size while fixing the average degree $E[D] = 16$. The GN benchmark has been extensively used in literature, despite introducing strong assumptions, such as communities of equal size, each node having the same degree and fixed graph size. Therefore, Lancichinetti *et al.* [97] proposed the LFR benchmark, where both the node degree vector d and community size vector n follows a power law distribution, a property found in many real-world networks. Additional details on LFR benchmark are deferred to Appendix D.2.2.

5.3. LINEAR CLUSTERING PROCESS (LCP) ON A GRAPH

5.3.1. CONCEPT OF THE CLUSTERING PROCESS

Each node i in the graph G is assigned a position $x_i[k]$ on a line (i.e. in one-dimensional space) at discrete time k. We define the $N\times 1$ position vector $x[k]$ at discrete time k, where the i-th vector component consists of the position $x_i[k]$ of node i at time k. We initialize the $N\times 1$ position vector $x[0]$ by placing nodes equidistantly on the line and assign integer values from 1 to N to the nodes, thus, $x[0] = \begin{bmatrix} 1 & 2 & \dots & N \end{bmatrix}^T$. At last, we restrict the position $x_i[k]$ to positive real values.

We propose a dynamic process that determines the position of nodes over time. The position difference between nodes of the same cluster is relatively small. On the contrary, nodes from different clusters are relatively far away, i.e. their position difference is relatively high. Based on the position vector $x[k]$, we will distinguish clusters, also called communities, in the graph G.

The proposed clustering process consists of two opposite and simultaneous forces that change the position of nodes at discrete time k:

Attraction. Adjacent nodes sharing many neighbours are mutually attracted with a force proportional to the number of common neighbours. In particular, the attractive force between node i and its neighboring node j is proportional to $\alpha\cdot\left(\left|\mathcal{N}_j\cap\mathcal{N}_i\right|+1\right)$, where α is the attraction strength and $\left(\left|\mathcal{N}_j\cap\mathcal{N}_i\right|+1\right)$ equals the number of common neighbors plus the direct link, i.e. $a_{ij}=1$.

Repulsion. Adjacent nodes sharing a few neighbours are repulsed with a force proportional to the number of different neighbours. The repulsive force between node i and its neighboring node j is proportional to $\delta\cdot\left(\left|\mathcal{N}_j\setminus\mathcal{N}_i\right|-1\right)$, where δ is the repulsive strength and $\left(\left|\mathcal{N}_j\setminus\mathcal{N}_i\right|-1\right)$ equals the set of neighbours of node j that do not belong to node i minus the direct link (that is included in $\left|\mathcal{N}_j\setminus\mathcal{N}_i\right|$). Since the force should be symmetric and the same if i and j are interchanged, we end up with a resultant repulsive force proportional to $\frac{1}{2}\cdot\delta\cdot\left(\left|\mathcal{N}_j\setminus\mathcal{N}_i\right|+\left|\mathcal{N}_i\setminus\mathcal{N}_j\right|-2\right)$.

5.3.2. LCP IN DISCRETE TIME

Since computers operate with integers and truncated real numbers, we concentrate on discrete-time modeling. The continuous-time description is derived in Appendix D.3. We denote the continuous-time variables by $y(t)$ and the continuous time by t, while the discrete-time counterpart is denoted by $y[k]$, where the integer k denotes the discrete time or k-th timeslot. The transition from the continuous-time derivative to the discrete-time difference is

$$\frac{dx_i(t)}{dt}=\lim_{\Delta t\to 0}\frac{x_i(t+\Delta t)-x_i(t)}{\Delta t}\to\left.\frac{x_i(t+\Delta t)-x_i(t)}{\Delta t}\right|_{\Delta t=1}\overset{\text{def}}{=}x_i[k+1]-x_i[k]$$

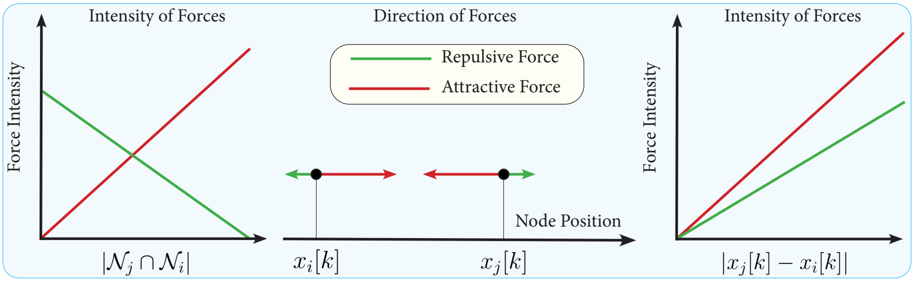


Figure 5.1: Dependence of the attractive and repulsive force on the number of common neighbours of adjacent nodes i and j (left-figure). Directions of the attraction and repulsion forces between the adjacent nodes (middle-figure). Dependence of the attractive and repulsive force on the absolute position distance between adjacent nodes i and j (right-figure).

5

Corresponding to the continuous-time law in Appendix D.3 and choosing the time step $\Delta t = 1$, the governing equation of position $x_i[k]$ of node i at discrete time k is

$$x_i[k+1] = x_i[k] + \sum_{j\in\mathcal{N}_i} \Bigg(\frac{\alpha \cdot \left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)}{d_j d_i} - \frac{\frac{1}{2} \cdot \delta \cdot \left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{d_j d_i} \Bigg) \cdot \left(x_j[k] - x_i[k]\right) \tag{5.7}$$

where α and δ are, in the discrete-time setting, the strength (in dimensionless units) for attraction and repulsion, respectively. The maximum position difference at the initial state is $x_N[0] - x_1[0] = N - 1$.

Node j attracts an adjacent node i with force proportional to their position difference $(x_j[k] - x_i[k])$. The intensity of the attractive force decreases as nodes i and j are closer on a line. The attraction is also proportional to the number common neighbours $\left|\mathcal{N}_j \cap \mathcal{N}_i\right|$ of node i and node j plus the direct link, as nodes tend to share most links with other nodes from the same cluster. On the contrary, node j repulses node i with a rate proportional to their position difference $(x_j[k] - x_i[k])$ and the average of the number of node j neighbours $\left|\mathcal{N}_j \setminus \mathcal{N}_i\right|$ that are not connected to the node i and, similarly, the number of node i neighbors, $\left|\mathcal{N}_j \setminus \mathcal{N}_i\right|$ that are not connected to the node j. The repulsive and attractive force are, as mentioned above, symmetric in strength, but opposite, if i is interchanged by j.

The directions of both attractive and repulsive forces between two adjacent nodes i and j as well the dependence of both forces on the number of common neighbours $\left|\mathcal{N}_j \cap \mathcal{N}_i\right|$ and the absolute position distance $\left|x_j[k] - x_i[k]\right|$ are illustrated in Figure 5.1.

In the continuous-time setting, as provided in Appendix D.9, we eliminate one parameter by scaling the time $t^* = \delta t$. Because the time step $\Delta t = 1$ is fixed and cannot be scaled, the discrete-time model consists of two parameters $\alpha \geq 0$ and $\delta \geq 0$.

So far, we have presented an additive law, derived in the common Newtonian approach. The corresponding multiplicative law in discrete time is

$$x_i[k+1] = x_i[k] \cdot \left(1 + \sum_{j \in \mathcal{N}_i} \left(\frac{\alpha \cdot \left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)}{d_i \cdot d_j} - \frac{\frac{1}{2} \cdot \delta \cdot \left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{d_i \cdot d_j}\right) \cdot \left(x_j[k] - x_i[k]\right)\right) \tag{5.8}$$

Although the physical intuition is similar, the multiplicative process in (5.8) behaves differently in discrete time than the additive law in (5.7). Since also the analysis is more complicated, we omit a further study of the multiplicative law.

We present the analogon of (5.7) in matrix form:

Theorem 19 *The discrete time process (5.7) satisfies the linear matrix difference equation*

$$x[k+1] = \left(I + W - diag(W \cdot u)\right) \cdot x[k], \tag{5.9}$$

where the $N \times 1$ vector u is composed of ones, the $N \times N$ identity matrix is denoted by I, while the $N \times N$ topology-based matrix W is defined as

$$W = (\alpha + \delta)\,\Delta^{-1} \cdot \left(A \circ A^2 + A\right) \cdot \Delta^{-1} - \frac{1}{2} \cdot \delta\left(\Delta^{-1} \cdot A + A \cdot \Delta^{-1}\right) \tag{5.10}$$

where $\circ$ denotes the Hadamard product. In particular,

$$w_{ij} = a_{ij} \frac{\alpha\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right) - \delta\left(\frac{\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right|}{2} - 1\right)}{d_i d_j} \tag{5.11}$$

The explicit solution of the difference equation (5.9) is

$$x[k] = \left(I + W - diag(W \cdot u)\right)^k x[0] \tag{5.12}$$

where the k-th component of the initial position vector is $(x[0])_k = k$.

Proof: Appendix D.4.1.

Theorem 19 determines the position of the nodal vector $x[k]$ at time k and shows convergence towards a state, where the sum of attractive and repulsive forces (i.e. the resulting force) acting on a node are in balance. Nodes with similar neighbourhoods are grouped on the line, i.e. in the one-dimensional space, while nodes with a relatively small number of common neighbours are relatively far away. A possible variant of the proposed linear clustering process may map the nodal position into a higher dimensional space, like a circular disk or square in two dimensions, and even with a non-Euclidean distance metric.

5.3.3. TIME-DEPENDENCE OF THE LINEAR CLUSTERING PROCESS

The $N \times N$ matrix $I + W - \text{diag}(W \cdot u)$ in the governing equation (5.9) has interesting properties. As shown in this section, the related matrix $W - \text{diag}(W \cdot u)$ belongs to the class of M-matrices, whose eigenvalues have a non-negative real part. The (weighted) Laplacian is another element of the M-matrix class.

Property 1 *The matrix $I+W-diag(W\cdot u)$ is a non-negative matrix.*

Proof: The governing equation (5.9)

$$x[k+1]=\left(I+W-\text{diag}\left(W\cdot u\right)\right)\cdot x[k]$$

holds for any non-negative vector $x[k]$. Let $x[0]=e_m$, the basic vector with components $(e_m)_i=\delta_{mi}$ and δ_{mi} is the Kronecker delta, then we find that the m-th column

$$x[1]=\left(I+W-\text{diag}\left(W\cdot u\right)\right)_{col(m)}$$

must be a non-negative vector. Since we can choose m arbitrary, we have established that $I+W-\text{diag}\left(W\cdot u\right)$ is a non-negative matrix. □

Property 2 *The principal eigenvector of the matrix $I+W-diag(W\cdot u)$ is the all-one vector u belonging to eigenvalue 1. All other eigenvalues of matrix $I+W-diag(W\cdot u)$ are real and, in absolute value, smaller than 1.*

5

Proof: Appendix D.4.2.

The linear discrete-time system in (5.9) converges to a steady-state, provided that $\lim_{k\to\infty}||x[k+1]||=\lim_{k\to\infty}||x[k]||=||x_s||$, which is only possible if the matrix $\left(I+W-\text{diag}\left(W\cdot u\right)\right)$ has all eigenvalues in absolute value smaller than 1 and the largest eigenvalue is precisely equal to 1. Property 2 confirms convergence and indicates that the steady-state vector $x_s=u$ in which the position of each node is the same. However, the steady state solution $x_s=u$ is a trivial solution, as observed from the governing equation in (5.7), because the sum vanishes and the definition of the steady state tells that $x[k+1]=x[k]$, which is obeyed by any discrete-time independent vector. In other words, the matrix equation (5.9) can be written as

$$x[k+1]-x[k]=\left(W-\text{diag}\left(W\cdot u\right)\right)\cdot\left(x[k]-u\right)$$

which illustrates that, if $x[k]$ obeys the solution, then $r[k]=x[k]+s\cdot u$ for any complex number s is a solution, implying that a shift in the coordinate system of the positions does not alter the physics.

Let us denote the eigenvector y_k belonging to the k-th eigenvalue β_k of the matrix $W-\text{diag}\left(W\cdot u\right)$, where $\beta_1\geq\beta_2\geq\cdots\geq\beta_N$, then the eigenvalue decomposition of the real, symmetric matrix is

$$W-\text{diag}\left(W\cdot u\right)=Y\text{diag}(\beta)Y^T$$

where the eigenvalue vector $\beta=(\beta_1,\beta_2,\cdots,\beta_N)$ and Y is the $N\times N$ orthogonal matrix with the eigenvectors $y_1,y_2,\cdots,y_N$ in the columns obeying $Y^TY=YY^T=I$. Since $\beta_1=0$ and $y_1=\frac{u}{\sqrt{N}}$, it holds for $k>1$ that $u^Ty_k=0$, which implies that the sum of the components of eigenvector y_k for $k>1$ is zero (just as for any weighted Laplacian [23]). The position vector in (5.12) is rewritten as

$$x[k]=Y\text{diag}(1+\beta)^kY^Tx[0]=\sum_{j=1}^{N}(1+\beta_j)^k y_j\left(y_j^Tx[0]\right)$$

Hence, we arrive at

$$x[k] - \frac{u^T x[0]}{\sqrt{N}} u = \sum_{j=2}^{N} (1+\beta_j)^k (y_j^T x[0])\, y_j \tag{5.13}$$

As explained above, the left-hand side is a translated position vector and physically not decisive for the clustering process. Since $-1 < \beta_j < 0$ for $j > 1$, relation (5.13) indicates that, for $k \to \infty$, the right-hand side tends to zero and the steady-state solution is clearly uninteresting for the clustering process. We rewrite (5.13) as

$$x[k] - \frac{u^T x[0]}{\sqrt{N}} u = (1+\beta_2)^k \left((y_2^T x[0])\, y_2 + \sum_{j=3}^{N} \left(\frac{1+\beta_j}{1+\beta_2} \right)^k (y_j^T x[0])\, y_j \right).$$

Since $|1+\beta_2| > |1+\beta_3|$, we observe that

$$\frac{x[k] - \frac{u^T x[0]}{\sqrt{N}} u}{(1+\beta_2)^k\, (y_2^T x[0])} = y_2 + O\left(\frac{1+\beta_3}{1+\beta_2} \right)^k, \tag{5.14}$$

which tells us that the left-hand side, which is a normalized or scaled, shifted position vector, tends to the second eigenvector y_2 with an error that exponentially decreases in k. Hence, for large enough k, but not too large k, the scaled shifted position vector provides us the information on which we will cluster the graph.

The steady state in Property 2 can be regarded as a reference position of the nodes and does not affect the LCP process nor the $N \times 1$ eigenvector y_2, belonging to the second largest eigenvalue $(1+\beta_2)$ of the $N \times N$ "operator" matrix $I + W - \text{diag}(W \cdot u)$, which is analogous to Fiedler clustering based on the $N \times N$ Laplacian Q. While the Laplacian matrix Q essentially describes diffusion and not clustering, our operator $I + W - \text{diag}(W \cdot u)$ changes the nodal positions, based on attraction and repulsion, from which clustering naturally arises.

Property 3 *The two parameters in the matrix W in (5.10) satisfy the bounds*

$$0 \le \alpha \le \frac{d_{\max} - 1}{d_{\max} - \frac{1}{2}\left(1 + \frac{d_{\min}}{d_{\max}}\right)} \le 1 \tag{5.15}$$

$$0 \le \delta \le \frac{1}{d_{\max} - \frac{1}{2}\left(1 + \frac{d_{\min}}{d_{\max}}\right)} \tag{5.16}$$

Proof: Appendix D.4.3.

The influence of the attraction strength α and the repulsion strength δ on the eigenvalues β_k and the $N \times 1$ eigenvector y_2 of the $N \times N$ matrix W is analysed in Appendix D.5.

5.4. FROM THE EIGENVECTOR y_2 TO CLUSTERS IN THE NETWORK

The interplay of the attractive and repulsive force between nodes drives the nodal position in discrete time k eventually towards a steady state $\lim_{k\to\infty} x[k] = u$. However, the

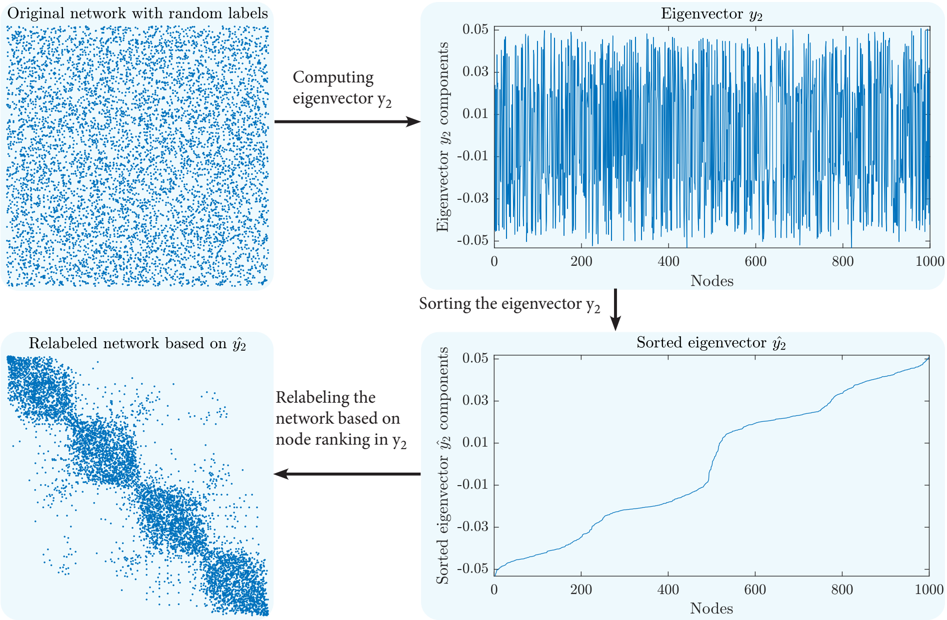


Figure 5.2: Adjacency matrix A of an SSBM network of $N = 1000$ nodes, $c = 4$ clusters and parameters $b_{in} = 26$, $b_{out} = 0.67$ (top-left). Eigenvector y_2 components (top-right). Sorted eigenvector $\hat{y}_2$ components (bottom-right). Relabeled adjacency matrix $\hat{A}$ based on the sorted eigenvector $\hat{y}_2$ (bottom-left).

scaled and shifted position vector $x[k]$ in (5.14) converges in time towards the second eigenvector y_2 with an exponentially decreasing error. In this section, we estimate the clusters in network, based on the eigenvector y_2.

By sorting the eigenvector y_2 to $\hat{y}_2$, the components of y_2 are reordered and the corresponding relabeling of the nodes of the network reveals a block diagonal structure of the adjacency matrix A. We define the $N \times N$ permutation matrix R in a way the following equalities hold:

$$\begin{gathered} \hat{y}_2 = R \cdot y_2, \\ (\hat{y_2})_i = \left(y_2\right)_{r_i} \leq (\hat{y_2})_j = \left(y_2\right)_{r_j}, \quad i < j, \end{gathered} \tag{5.17}$$

where the $N \times 1$ ranking vector $r = R \cdot w$ and $w = [1, 2, \ldots, N]$, with r_i denoting the node i ranking in the eigenvector y_2. The permutation matrix R allow us to define the $N \times N$ relabeled adjacency matrix $\hat{A}$, the $N \times 1$ relabeled degree vector $\hat{d}$ of G, and the $N \times 1$ sorted eigenvector $\hat{y}_2$ as follows:

$$\begin{cases} \hat{A} &= R^T \cdot A \cdot R \\ \hat{d} &= R \cdot d \\ \hat{y}_2 &= R \cdot y_2. \end{cases} \tag{5.18}$$

Groups of nodes that have relatively small difference in the eigenvector y_2 components, while relatively large difference compared to other nodes in the network, compose a cluster. Therefore, the community detection problem transforms into recognizing intervals of similar values in the sorted eigenvector $\hat{y}_2$.

Figure 5.2 exemplifies the idea, where the adjacency matrix A of a randomly labeled SSBM network of $N = 1000$ nodes and $c = 4$ clusters is presented in the upper-left part, as a heat map. The eigenvector y_2 is drawn in the upper-right part, while the sorted eigenvector $\hat{y}_2$ is drawn on the bottom-right side. Finally, the relabeled adjacency matrix $\hat{A}$, based on nodal ranking of y_2 is depicted on the lower-left side. The sorted eigenvector $\hat{y}_2$ reveals a stair with four segments, equivalent to four block matrices on the main diagonal in relabeled adjacency matrix $\hat{A}$.

The eigenvector y_2 represents a continuous measure of how similar neighbours of two nodes are. There are two different approaches to identify network communities for a given eigenvector y_2:

- Cluster identification based on the sorted eigenvector $\hat{y}_2$. This approach is explained in subsection 5.4.1.
- Cluster identification based on the ranking vector r. This approach does not rely on the eigenvector y_2 components, but solely on nodal ranking, as explained in subsection 5.4.2.

5

5.4.1. COMMUNITY DETECTION BASED ON NODAL COMPONENTS OF THE EIGENVECTOR y_2

To identify clusters, we observe the difference in eigenvector y_2 components between nodes with adjacent ranking. If $(\hat{y}_2)_{i+1} - (\hat{y}_2)_i < \theta$, where θ denotes a predefined threshold, then the nodes r_i and r_{i+1} belong to the same cluster, else the nodes r_i and r_{i+1} are boundaries of two adjacent clusters. The resulting cluster membership function is

$$C_{r_{i+1},r_i} = \begin{cases} 1 & (\hat{y}_2)_{i+1} - (\hat{y}_2)_i < \theta \\ 0 & \text{otherwise}, \end{cases} \tag{5.19}$$

where the threshold value θ is determined heuristically. The cluster estimation in (5.19) can be improved by using other more advanced approaches, such as the K-means algorithm.

5.4.2. MODULARITY-BASED COMMUNITY DETECTION

By implementing (5.4) and (5.18) into (5.3) we obtain:

$$m = \frac{1}{2L} \cdot u^T \cdot \left(\hat{A} \circ \hat{C} - \frac{1}{2L} \cdot \left(\hat{d} \cdot \hat{d}^T \right) \circ \hat{C} \right) \cdot u, \tag{5.20}$$

where $\hat{C} = R^T \cdot C \cdot R$. As shown in Figure 5.2, the network relabeling based on the ranking vector r reveals block diagonal structure in $\hat{A}$. Thus, the relabeled modularity matrix $\hat{C}$ has the following block diagonal structure:

$$\hat{C} = \begin{bmatrix} J_{n_1 \times n_1} & O_{n_1 \times n_2} & \cdots & O_{n_1 \times n_c} \\ O_{n_2 \times n_1} & J_{n_2 \times n_2} & \cdots & O_{n_1 \times n_c} \\ \vdots & \vdots & \cdots & \vdots \\ O_{n_1 \times n_1} & O_{n_c \times n_2} & \cdots & J_{n_c \times n_c} \end{bmatrix}, \tag{5.21}$$

where c denotes number of clusters in network, where the i-th cluster is composed of n_i nodes. We highlight that relation (5.21) holds only in the case of a classical community problem, i.e. when each node belongs to exactly one community.

Algorithm 1 Recursive algorithm for cluster estimation

Require: $\hat{A}$ and $\hat{d}$ are the relabeled adjacency matrix A and the degree vector d (5.18), while L denotes number of links. The modularity threshold is denoted by θ. The function returns the $c \times 1$ vector b, whose elements are cluster borders in a relabeled graph.

function ESTIMATECLUSTERS($\hat{A}$, $\hat{d}$, N, L, θ)
  $d_f, d_b, p, q \leftarrow O_{N\times 1}$
  $\left(d_f\right)_1 \leftarrow \hat{d}_1$
  $(d_b)_N \leftarrow \hat{d}_N$
  $p_1 \leftarrow -\frac{\hat{d}_1^2}{(2L)^2}$
  $q_N \leftarrow -\frac{\hat{d}_N^2}{(2L)^2}$
  for $i \leftarrow 2$ to N **do**
    $l \leftarrow N - i$
    $\left(d_f\right)_i \leftarrow \left(d_f\right)_{i-1} + \hat{d}_i$
    $(d_b)_{N-i+1} \leftarrow (d_b)_{N-i+2} + \hat{d}_{N-i+1}$
    $s \leftarrow \frac{\sum_{j=1}^{i} \hat{a}_{ij}}{L} - \frac{2\cdot\hat{d}_i\cdot(d_f)_{i-1}+\hat{d}_i^2}{(2L)^2}$
    $t \leftarrow \frac{\sum_{j=1}^{i} \hat{a}_{N-j+1,l+1}}{L} - \frac{2\cdot\hat{d}_{l+1}\cdot(d_b)_{l+2}+\hat{d}_{l+1}^2}{(2L)^2}$
    $p_i \leftarrow p_{i-1} + s$
    $q_{l+1} \leftarrow q_{l+2} + t$
  end for
  $r \leftarrow \arg\max_{\mathcal{N}} (p+q)$
  if $(p+q)_r > \theta$ **then**
    $\hat{A}_1$, $\hat{d}_1$, $N_1 \leftarrow$ sub-matrix(vector) corresponding to the first cluster $\{1, 2, \ldots, r\}$
    $\hat{A}_2$, $\hat{d}_2$, $N_2 \leftarrow$ sub-matrix(vector) corresponding to the second cluster $\{r+1, r+2, \ldots, N\}$
    return $\hat{b} \leftarrow \begin{bmatrix} \text{EstimateClusters}(\hat{A}_1, \hat{d}_1, N_1, L, p_r) \\ r, \\ \text{EstimateClusters}(\hat{A}_2, \hat{d}_2, N_2, L, q_r) \end{bmatrix}$
  else
    return $\hat{b} \leftarrow \emptyset$
  end if
end function

We define the $N \times 1$ vectors $\hat{e}_i$ for $i = \{1, 2, \ldots, c\}$ as

$$\hat{e}_i = \left[O_{\left(1\times\sum_{j=1}^{i-1} n_j\right)} \quad u_{(1\times n_i)} \quad O_{\left(1\times\sum_{j=i+1}^{N} n_j\right)} \right]^T, \tag{5.22}$$

that allows us to redefine $\hat{C} = \sum_{i=1}^{c} \hat{e}_i \cdot \hat{e}_i{}^T$ and further simplify (5.20):

$$m = \frac{1}{2L} \cdot \sum_{i=1}^{c} \hat{e}_i{}^T \cdot \left(\hat{A} - \frac{1}{2L} \cdot (\hat{d} \cdot \hat{d}^T) \right) \cdot \hat{e}_i. \tag{5.23}$$

Since the vector $\hat{e}_i$ consists of zeros and ones, the equation (5.23) represents the sum of elements of the matrix $(\hat{A} - \frac{1}{2L} \cdot (\hat{d} \cdot \hat{d}^T))$ corresponding to each individual cluster.

We estimate clusters for a given ranking vector r by recursively optimising the modularity m. In the first iteration, we examine all possible partitions of the network in two clusters and compute their modularity. The partition that generates the highest modularity is chosen. In the second iteration, we repeat for each subgraph the same procedure and find the best partitions into two clusters. Once we determine the best partitions for both subgraphs, we adopt them if the obtained modularity of the generated partition exceeds the modularity of a parent cluster from the previous iteration. The recursive procedure stops when the modularity m cannot be further improved, as described by pseudocode (1). This version of the proposed process is denoted as LCP in section 5.6.

5.4.3. MODULARITY-BASED COMMUNITY DETECTION FOR A KNOWN NUMBER OF COMMUNITIES

The algorithm 1 also applies for graph partition with a known number of communities c. In that case, instead of stopping the recursive procedure described in algorithm 1 when the modularity m cannot be further improved, we stop at iteration $(\log_2 c + 1)$. In each iteration, the partition with the maximum modularity is accepted, even if negative.

As a result, we obtain $2c$ estimated clusters with the $2c \times 2c$ aggregated modularity matrix M_c:

$$(M_c)_{gh} = \sum_{i \in g, j \in h} \left(\hat{A} - \frac{1}{2L} \cdot \hat{d} \cdot \hat{d}^T \right)_{ij}, \tag{5.24}$$

where $g, h \in \{1, 2, \ldots, 2c\}$ denote estimated communities. The aggregated modularity matrix M_c allows us to merge adjacent clusters, until we reach c communities in an iterative way. We observe the $(2c - 1 \times 1)$ vector μ, where $\mu_g = (M_c)_{g,g+1}$. The maximum element of μ indicates which two adjacent clusters can be merged, so that modularity index m is negatively affected the least. By repeating this procedure c times, we end up with the graph partition in c clusters. This version of the proposed process is denoted as LCP_c in Section 5.6.

5.4.4. NON-BACK TRACKING METHOD VERSUS LCP

Angel *et al.* [98, p.12] noted that the $2N$ non-trivial eigenvalues of the $2L \times 2L$ non-back tracking matrix B from (D.6) are contained in eigenvalues of the $2N \times 2N$ matrix B^*:

$$B^* = \begin{bmatrix} A & I - \Delta \\ I & O \end{bmatrix}, \tag{5.25}$$

where the $N \times N$ matrix with all zeros is denoted as O. The $2N \times 2N$ matrix B^*, written as

$$B^* = \begin{bmatrix} I + (A - \Delta) + (\Delta - I) & -(\Delta - I) \\ I & O \end{bmatrix}$$

can be considered as a state-space matrix of a process on a network, similar to our LCP process in (5.7), with the last N states storing delayed values of the first N states. The $2N \times 2N$ matrix B^* defines the set of N second-order difference equations, where the governing equation for the node i position is

$$x_i[k+1] = x_i[k] + \sum_{j \in \mathcal{N}_i} (x_j[k] - x_i[k]) + (d_i - 1) \cdot (x_i[k] - x_i[k-1])$$

We recognize the second term in (5.26) as an attraction force between neighbouring nodes with uniform intensity, while in our LCP (5.7) the attraction force intensity is proportional to the number of neighbours two adjacent nodes share. Further, while we propose a repulsive force between adjacent nodes in (5.7), node i in (5.26) is repulsed from its previous position $x_i[k]$ in direction of the last position change $(x_i[k] - x_i[k-1])$.

We implement the weighted intensity of the attractive force as in (5.7), ignoring the repulsive force by letting $\delta = 0$, and define the $2N \times 2N$ matrix W^*, corresponding to B^*,

$$W^* = \begin{bmatrix} I + \alpha \cdot \left(A \circ A^2 + A - \text{diag}\left(\left(A \circ A^2 + A \right) \cdot u \right) \right) + (\Delta - I) & -(\Delta - I) \\ I & O \end{bmatrix}. \tag{5.26}$$

5

We estimate the number of clusters c in a network from W^* similarly as in the non-back tracking method in Sec. D.1.4 by counting the number of eigenvalues in W^* with real component larger than $\sqrt{\lambda_1(W^*)}$. This approach is denoted as LCP$_n$ in Section 5.6.

5.5. REDUCING INTENSITY OF FORCES BETWEEN CLUSTERS

The idea behind a group of methods in community detection, called divisive algorithms, consists of determining the links between nodes from different clusters. Once these links have been identified, they are removed and thus only the intra-community links remain [96]. We invoke a similar idea to our linear clustering process.

An outstanding property of our approach is that the LCP defines the nodal position as a metric, allowing us to perform clustering in multiple ways. The position distance between any two, not necessarily adjacent nodes indicates how likely the two nodes belong to the same cluster. Then, the position metric also allows us to classify links as either intra- or inter-community. Thus, we iterate the linear clustering process (5.7) and, in each iteration, we identify and scale the weights of the inter-community links.

The attraction and repulsive forces are defined as linear functions of the position difference between two neighbouring nodes, as presented in Figure 5.1. While linear functions greatly simplify the complexity and enable a rigorous analysis, the linearity of forces introduces some difficulties in the process. Firstly, as two adjacent nodes are further away, both the attractive and the repulsive force between them increase in intensity. Similarly, as the neighbouring nodes are closer on a line, both forces decrease in intensity and converge to zero as the nodes converge to the same position. Secondly, the attractive force between any two neighbouring nodes is always of higher intensity than the repulsive force, causing the process to converge towards the trivial steady-state.

Non-linearity in the forces can be introduced in the proposed linear clustering process iteratively by scaling the weights of inter-community links between iterations, which artificially decreases the strength of forces between the two nodes from different

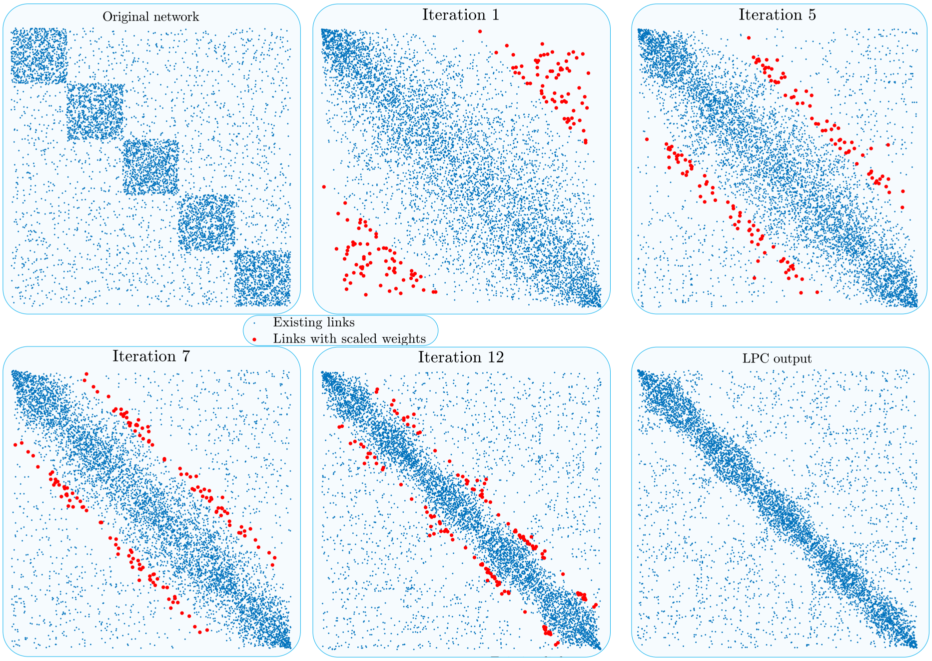


Figure 5.3: Adjacency matrix A of an SSBM network of $N = 1000$ nodes, $c = 5$ clusters of equal size, with parameters $b_{in} = 26$ and $b_{out} = 2.25$ (top-left). The following 4 subfigures present the relabeled adjacency matrix based on the ranking vector r in iterations 1, 5, 7 and 12, respectively. In each iteration, the weights of 2% links are scaled (red colour). The weight of each link is allowed to be scaled once. The relabelled adjacency matrix $\hat{A}$ after 15 iterations of scaling weights of links between clusters (bottom-right).

clusters. In other words, we reduce the importance of links between nodes from different clusters based on the partition from the previous iteration.

5.5.1. SCALING THE WEIGHTS OF INTER-COMMUNITY LINKS

The difference $|(y_2)_i - (y_2)_j|$ in the eigenvector y_2 components of nodes i and j indicates how similar neighbourhoods of these nodes are. A normalized measure for the difference in neighbouring nodes i and j is the difference $(|r_i - r_j|)$ of their rankings in the sorted eigenvector $\hat{y}_2$. Thus, links that connect nodes with the highest ranking difference are most likely inter-community links. We define the $N \times N$ scaling matrix S as follows:

$$s_{ij} = \begin{cases} 1, & \text{if } |r_j - r_i| < \theta_r \\ v, & \text{otherwise}, \end{cases} \tag{5.27}$$

where the ij-th element equals 1 if the absolute value of the ranking difference between nodes i and j is below a threshold θ_r, otherwise some positive value $0 \le v \le 1$. Based on the $N \times N$ scaling matrix S in (5.27), we update the governing equation as follows:

$$x[k+1] = \left(I + \tilde{W} - \text{diag}\left(\tilde{W} \cdot u\right)\right) \cdot x[k],$$

where $\tilde{W} = S \circ W$. Scaling the link weights in (5.27) only impacts the clustering process in (5.9), as defined in the equation above. However, modularity-based community detection, explained in Section 5.4.2, operates on the $N \times N$ adjacency matrix A in each iteration. Therefore, our implementation of scaling the weights of inter-community connections in network helps the process to better distinguish between clusters (i.e. eventually provides better relabeling in (5.18)), without modifying the $N \times N$ adjacency matrix A and, hence, without negatively affecting the modularity m optimisation in Algorithm 1. An example of removing links (i.e. $v = 0$) is depicted on Figure 5.3, where in each iteration weights of $\frac{15}{4}$% identified inter-cluster links are scaled. Scaling the weights of links between clusters significantly improves the quality of the identified graph partition.

5.6. Benchmarking LCP with other clustering methods

5

Computational complexity of the entire proposed clustering process equals $O(N \cdot L)$, as derived in Appendix D.6. In this section, we benchmark the linear clustering process (5.7) against popular clustering algorithms (introduced in Appendix D.1), both on synthetic and real-world networks. The non-back tracking algorithm (Appendix D.1.4) and our LCP$_n$ (Sec 5.4.4) estimate only number of clusters, Newman's method (Appendix D.1.3), the Leiden method (Appendix D.1.2) the Louvain method (Appendix D.1.1) and our LCP (Sec 5.4.2) estimate both number of clusters and the cluster membership of each node, while LCP$_c$ (Sec 5.4.3) requires the number of communities c to perform graph partitioning. The attractive strength $\alpha = 0.95$ and the repulsive strength $\delta = 10^{-3}$ are used in all simulations. Weights of 60% links in total are scaled using (5.27), evenly over 30 iterations, where in i-th iteration scaled weight is $\frac{0.05 \cdot i}{30}$.

5.6.1. Clustering performances on stochastic block generated graphs

We compare the clustering performance of our LCP with that of clustering methods introduced in Appendix D.1, on the same graph generated by the symmetric stochastic block model (SSBM) with clusters of equal size. All graphs have $N = 1000$ nodes. We vary the parameters b_{in} and b_{out} using (D.7) in a way to keep the average degree $d_{av} = 7$ fixed. For each SSBM network, we execute the clustering methods 10^2 times and present the mean number of estimated clusters and mean modularity of produced partitions in Figures (5.4-5.5).

The clustering performance on SSBM graphs with $c = 2$ clusters ($c = 4$ clusters) is presented on the left-hand side (right-hand side) of Figure 5.4, respectively. The non-back tracking algorithm and our LCP$_n$ achieve the best performance in estimating the number of communities c, as shown in the upper part of Figure 5.4. Further, our LCP outperforms each considered modularity-based method in identifying the number of communities c and in modularity m. Furthermore, when clusters are visible (i.e. above the detectability threshold), the NMI value (presented in the bottom figures) of our LCP and our LCP$_c$ significantly outperforms other clustering algorithms. Figure 5.4 illustrates a significant difference in performance between our LCP and the non-back tracking matrix (NBT) method. Our LCP (in blue) and the other three modularity-based methods perform

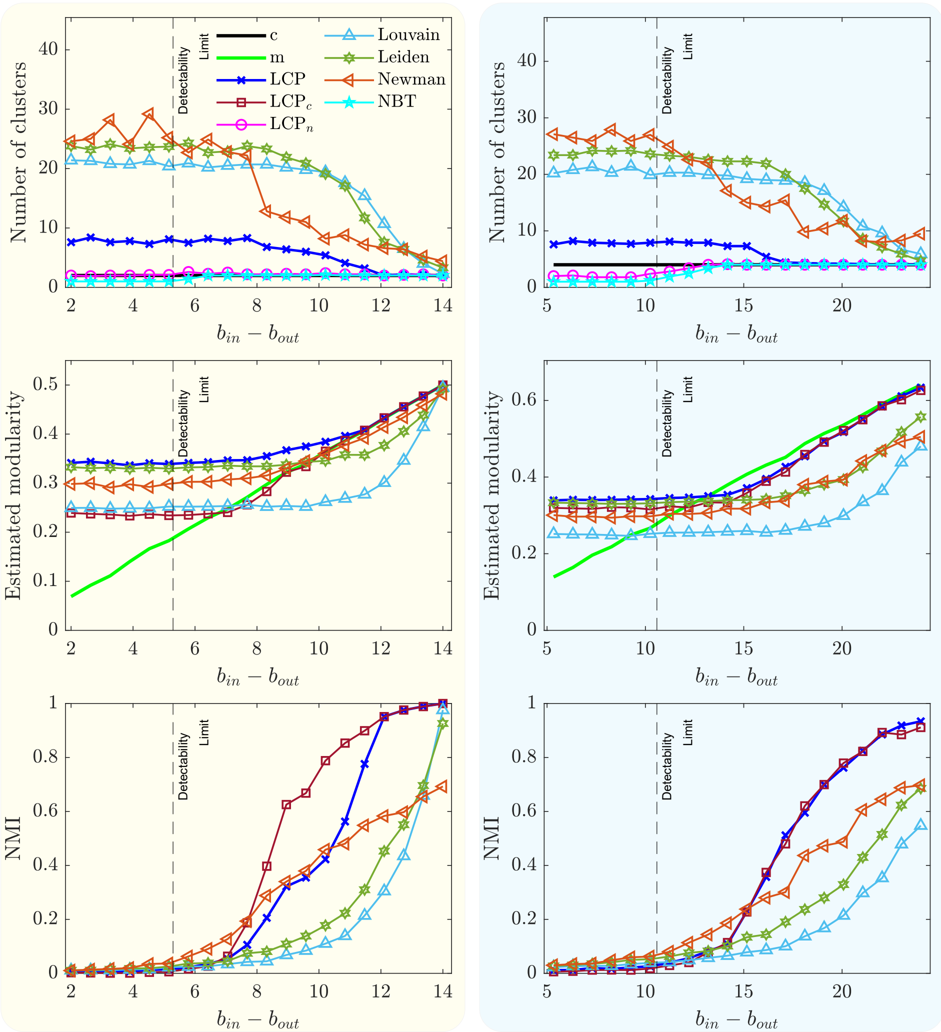


Figure 5.4: The estimated number of clusters (upper figures) in SSBM graphs with $N = 1000$ nodes, average degree $d_{av} = 7$, $c = 2$ (left-hand side) and $c = 4$ (right-hand side) clusters, respectively, for different values of parameters b_{in} and b_{out}. The modularity of the estimated partitions is presented in the central figures, while the NMI measure per each clustering algorithm is provided at the bottom figures. The vertical dashed line indicates the clustering detectability threshold.

poorly in recognising the number c of clusters for a wide range of $b_{in} - b_{out}$ (around and below the detectability threshold). Poor performance occurs because modularity-based methods generate partitions of higher modularity than the original network (in black) but with different communities! Consequently, the NMI measure deteriorates in these regimes. Our LCP_n (in red), for a given number of communities c, identifies partitions with higher modularity m than of the original network, even within the theoretically detectable regime.

Figure 5.5 illustrates results for SSBM graphs of $N = 1000$ nodes, with $c = 8$ (left-hand side) and $c = 20$ (right-hand side) clusters. Our LCP consistently outperforms the other three methods in estimated modularity m over the entire range of $b_{in} - b_{out}$ values. Except for $b_{in} - b_{out}$ values around and below the detectability threshold, the NMI measure of our LCP is superior to the other three methods (bottom figures).

5.6.2. CLUSTERING PERFORMANCES ON LFR BENCHMARK GRAPHS

Figure 5.6 illustrates clustering results on LFR benchmark graphs of $N = 500$ nodes with $c = 5$ (left-hand part) and $c = 11$ (right-hand part) communities. Compared to Newman, Louvain and Leiden algorithm, our LCP is among the best in estimating the number of clusters c (upper figures) while outperforming each considered method in estimated modularity m (middle figures). In addition, our LCP provides the highest NMI measure when the clusters are visible (i.e. for low μ value). For relatively large values of μ, our LCP identifies partitions different from the original one but with considerably higher modularity. Therefore, the NMI measure deteriorates in this regime (lower figures). When a graph is generated by the LFR benchmark, the non-backtracking method (NBT) and our LCP_n fail to estimate the number of clusters c.

5

5.6.3. CLUSTERING PERFORMANCES ON REAL-WORLD NETWORKS

Table 5.6.3 summarises the clustering performance of our LCP and those considered existing algorithms on seven real-world networks of different sizes, number of links and community structure. In five out of seven cases, our LCP provides partition with the highest modularity m, compared to other algorithms. LCP's superiority in achieved modularity m aligns with the results obtained on synthetic benchmarks. While the estimated number of clusters c of each method cannot be judged as the ground truth is unknown, LCP's estimated number of communities c is, on average, the closest to that of the non-back tracking matrix, known as one of the best predictors in the literature.

5.7. CONCLUSION

In this chapter, we propose a linear clustering process (LCP) on a network consisting of an attraction and repulsion process between neighbouring nodes, proportional to how similar or different their neighbours are. Based on nodal positions, we are able to estimate both the number c and the nodal membership of communities. Our LCP outperforms modularity-based clustering algorithms, such as Newman's, Leiden and the Louvain method, on both synthetic and real-world networks while being of the same computational complexity. The proposed LCP allows estimating the number c of clusters as accurately as the non-back tracking matrix in case of SSBM graphs. A potential

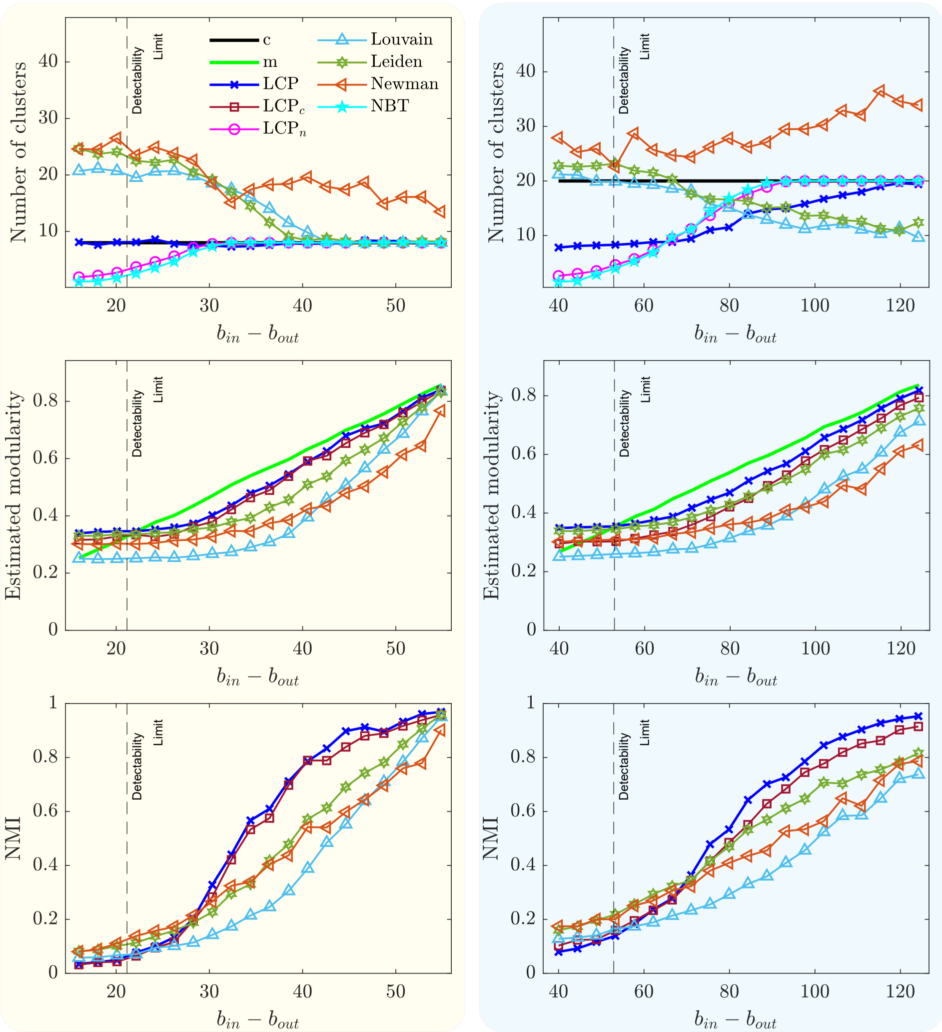


Figure 5.5: The estimated number of clusters (upper figures) in SSBM graphs with $N = 1000$ nodes, average degree $d_{av} = 7$, $c = 8$ (left-hand side) and $c = 20$ (right-hand side) clusters, respectively, for different values of parameters b_{in} and b_{out}. The modularity of the estimated partitions is presented in the central figures, while the NMI measure per each clustering algorithm is provided at the bottom figures. The vertical dashed line indicates the clustering detectability threshold.

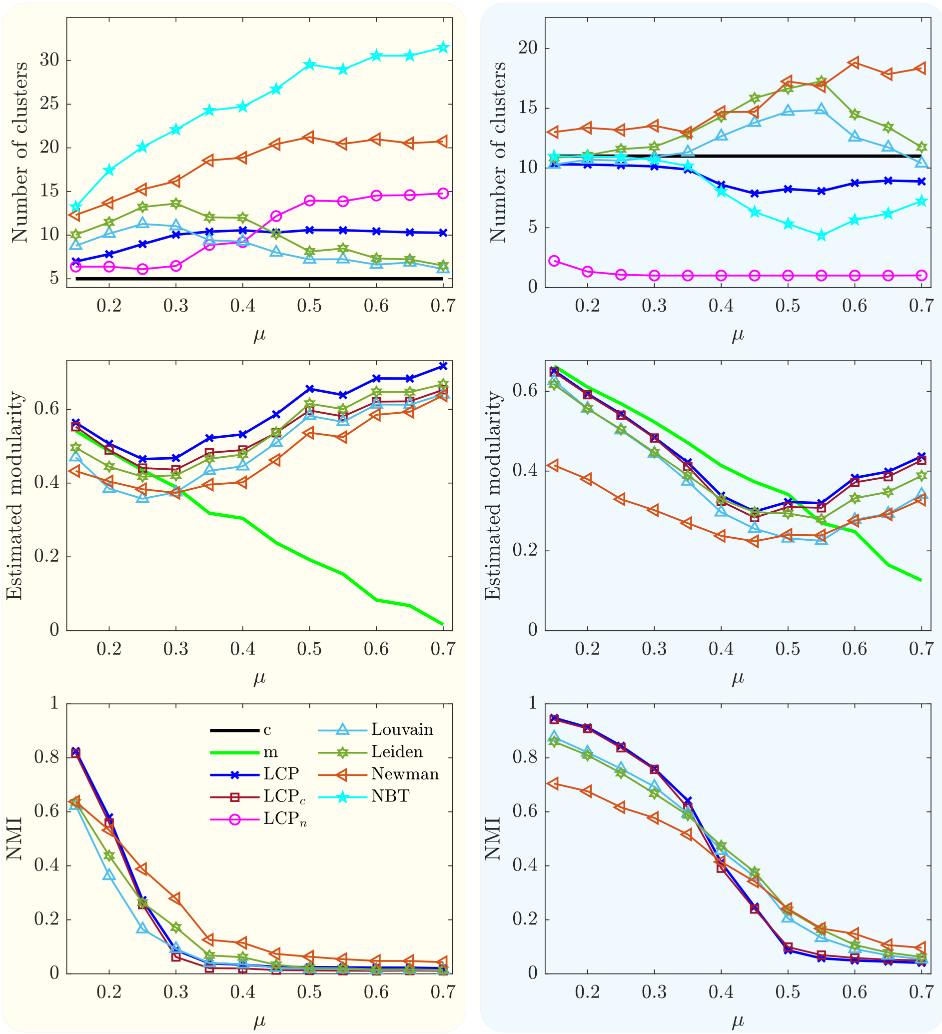


Figure 5.6: The estimated number of clusters (upper figures) in LFR benchmark graphs of $N = 500$ nodes with the average degree $d_{av} = 12$, consisting of $c = 5$ (left-hand side, with $\gamma = 1$ and $\beta = 2$) and $c = 11$ (right-hand side with $\gamma = 2$ and $\beta = 3$) clusters, respectively, for different values of parameter μ. The modularity of the estimated partitions is presented in the central figures, while the NMI measure per each clustering algorithm is provided at the bottom figures.

Table 5.1: Clustering performance of our LCP and considered existing clustering algorithms on real-world networks.

Real-world networks			LCP		Louvain		Leiden		Newman		NBT	LCP_c
Network name	N	L	c	m	c	m	c	m	c	m	c	c
Karate Club	34	78	3	**0.3922**	4	0.3565	4	0.3729	5	0.3776	2	1
Dolphins	62	159	4	0.5057	4	0.4536	5	**0.5105**	6	0.4894	2	2
Polbooks	105	441	3	**0.5160**	4	0.4897	4	0.5026	8	0.4160	3	2
Football	115	613	7	**0.5894**	7	0.5442	7	0.5635	11	0.4623	10	5
Facebook	347	2519	8	**0.4089**	16	0.3726	18	0.3792	23	0.3770	8	4
Polblogs	1490	19090	19	**0.4224**	7	0.3385	11	0.3117	4	0.3459	8	1
Co-autorship	1589	2742	40	0.9296	272	**0.9423**	270	0.9410	28	0.7393	23	16

improvement of the proposed linear clustering process lies in a more effective way of scaling inter-community link weights between successive iterations.

The linear clustering process LCP is described by a matrix $I + W - \text{diag}(W \cdot u)$, which can be regarded as an operator acting on the position of nodes, comparable to quantum mechanics (QM). In QM, an operator describes a dynamical action on a set of particles. Since quantum mechanical operators are linear, the dynamics are exactly computed via spectral decomposition. In a same vein, our operator $I + W - \text{diag}(W \cdot u)$ is linear and describes via attraction and repulsion a most likely ordering of the position of nodes that naturally leads to clusters, via spectral decomposition, in particular, via the eigenvector y_2 in Section 5.3.3.

5

6

Time Dynamics of the Dutch Municipality Network

Science is the poetry of reality.

Richard Dawkins

Based on data sets provided by Statistics Netherlands and the International Institute of Social History, we investigate the Dutch municipality merging process and the survivability of municipalities over the period 1830 – 2019*. We examine the dynamics of the population and area per municipality and how their distributions evolved during the researched period. We apply a Network Science approach, where each node represents a municipality and the links represent the geographical interconnections between adjacent municipalities via roads, railways, bridges or tunnels which were available in each specific yearly network instance. Over the researched period, we find that the distributions of the logarithm of both the population and area size closely follow a normal and a logistic distribution respectively. The tails of the population distributions follow a power-law distribution, a phenomenon observed in community structures of many real-world networks. The dynamics of the area distribution are mainly determined by the merging process, while the population distribution is also driven by the natural population growth and migration across the municipality network. Finally, we propose a model of the Dutch Municipality Network that captures population increase, population migration between municipalities and the process of municipality merging. Our model allows for predictions of the population and area distributions over time.*

This chapter is based on [99].

6.1. INTRODUCTION

The process of urbanization by which large numbers of people permanently resided in cities[1] has marked our history. The technological changes that enabled the urbanization and industrialization of our society took centuries to shape our cities, villages and rural areas. Identifying relevant governing factors and understanding the influence of different processes, such as population evolution, people migration, and urban growth, is essential for urban planning and policy-making.

Marchetti revealed in [100] how people's movements and commuting time depend on technological innovations in transport. In addition, Gonzales *et al.* [101] observed that human motion is characterised by both temporal and spatial regularity while obeying simple, reproducible patterns. Human movement is often modelled as a random walk, known as a Levy flight [102], where the distribution of travelling distance follows a power law [103]. In contrast to human mobility patterns, job-related and socioeconomic well-being variables govern migration flows of people [104], whose trends can be age-specific, as observed by Johnson and Fuguitt in [105]. Migration patterns further shape the development of urban and rural areas. Makse *et al.* [106] proposed a percolation-based model for city growth, following the principle that urban area development leads to further development. In addition, they found that the area distribution of towns surrounding a city follows a power law in the case of Berlin and London in years 1920, 1945 and 1981 respectively. Schlapfer *et al.* [107] empirically confirmed the scale-invariant increase of interactions between humans with city size.

6

When researching phenomena related to geographical urban areas, most often cities are considered [106, 108–111] as a basic unit, thus limiting the analysis to only a part of the entire urbanization spectrum of a country. Consequently, the geographical influence between neighbouring areas cannot be adequately considered [112]. In this research, a set of municipal[2] units is chosen rather than cities, for two reasons:

1 City boundaries are unofficial and often ambiguously defined compared to municipalities that enclose their localities and rural area situated on a particular part of national land and account for their particular part of the total national population.

2 All cities belonging to one country do not cover together the entire national surface and do not comprise the entire national population. In contrast, municipalities together constitute an entire country in terms of land surface and population, allowing for analysis of a country as a network of interconnected municipalities. To the best of our knowledge, the evolution of municipalities over time has not been analysed from a network perspective before.

A large system of elements (nodes) and their interactions or relations (links) can be represented by a network. The characterization of networks has been extensively investigated for classification purposes and for understanding the effects of the network

[1] Urbanization|Britannica.

[2] A municipality is a city or a town or a set of localities having a dedicated local government (Municipality|Cambridge Dictionary).

structure on its functioning [2, 113, 114]. From a network science perspective, this research focuses on understanding the underlying processes that influence the evolution and survivability of geographical areas of a country.

This chapter concentrates on the population and area size distributions at the municipality level and the processes that change their characteristics over time. We argue that the population and area size, together with the underlying topology, sufficiently correlate with the probability of a municipality being annexed by a neighbouring municipality. We demonstrate that the municipality merging process changes the area size distribution. The population distribution is also influenced by the continuous (inter)national migration of people across the municipality network. Regarding migration we distinguish two different types of migration flows:

- People moving from small(er) to large(r) municipalities in terms of population size. This migration flow occurs due to more attractive characteristics such as urban infrastructure, better facilities, employment and economic opportunities available in large(r) municipalities [107, 109],

- People moving from municipalities with a large(r) population to municipalities with small(er) population. This migration flow, enabled by mass-commuting[3] since the 1960s, occurs due to a more attractive cost of living, more space per person and affordable housing, thus avoiding the drawbacks of densely populated urban municipalities.

These two opposite migration flows take place simultaneously and shape the migration patterns across the municipality network. These migration flows can be regarded as an optimisation process in which people aim to obtain advantages of both small(er) and large(r) municipalities as much as they can. People tend to live close enough to large urban areas to enjoy the benefits but, on the contrary, distant enough to also enjoy the additional living space and nature in smaller localities. Consequently, individual citizens decide about the trade-off between their commuting time [100] and geographical distance[4] between their specific household situation and their work locations. As a result of all these individual decisions, the two migration flows directly govern the population distribution while indirectly influencing the merging process, the topology change and area distribution over time.

After collecting and combining approximately 200 years of Dutch statistical data into one large multi-layer network where each layer contains both the population and area per municipality per year, we focus on the Netherlands and we design a method and model allowing for quantitative network analysis. However, our research approach can be applied to any urbanized country if lengthy time series of statistical data are available from the respective national statistical offices.

From densely populated urban areas to the smallest villages in The Netherlands, more than 2000 localities are grouped into municipalities. Continuing today, a major

[3] The increase of the number of privately owned cars in The Netherlands is described in the 2019 publication of Statistics Netherlands; De groei van het Nederlandse personenautopark [115].

[4] While in 1947 only 15% of working population in The Netherlands worked outside the municipality where they lived, in 2006 that percentage reached 56% [115].

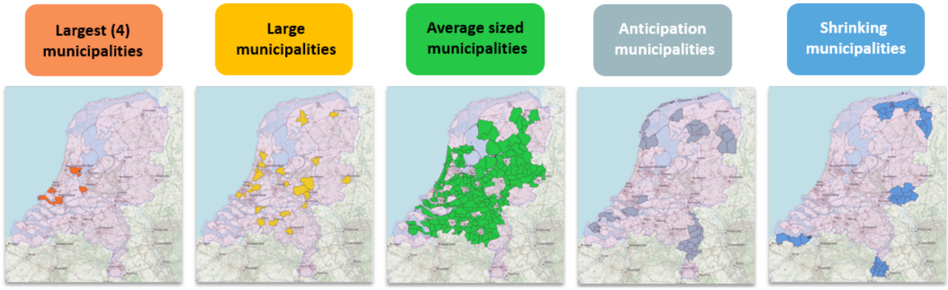


Figure 6.1: Classification of the Dutch municipalities in five categories of population.

development observed from recorded population-related time series[5] is the municipality merging process, also referred to as municipal restructuring [116]. Figure 6.1 shows a classification of all Dutch municipalities in five[6] population categories. Although sometimes newly established municipalities occur in municipal restructuring, due to coalitions, renaming and in a few cases creating land from water, in this chapter, we use the term *municipality merging process* for all processes influencing CBS-codes (explained in Appendix E.3). The number of Dutch municipalities decreased since the beginning of industrialization (the end of the first half of the 19th century), while the population steadily increased. For example, at the beginning of the industrial revolution, The Netherlands consisted of 1228 mainly rural municipalities, gradually decreasing to 1016 in 1947 towards 355 mainly urbanized municipalities in 2019. In the meantime, the national population density tripled between 1905 and 2010. According to the Ministry of Interior Affairs, The Netherlands has 9 shrinking areas[7] and 11 anticipation areas[8] as shown in Figure 6.1. A shrinking area [116] is defined as an area where the population is expected to decrease by at least 12.5% until 2040, while the decrease in the number of households is expected at least 5%. Areas, where the population is declining less rapidly, are called anticipation areas. In anticipation areas, the population is forecast to decrease by at least 2.5% until 2040.

In Section 6.2 we define the Dutch Municipality Network. We analyse over time its topology changes: how its population and area sizes evolved at the national, province and municipality level. Section 6.3 examines the governing processes behind population and area distribution changes over time from which we propose a model for the Dutch Municipality Network in Section 6.4. In Section 6.5 we conclude.

6.2. Dutch Municipality Network

We construct the Dutch Municipality Network (DMN) from a dataset consisting of geographical municipality-related polygons for each year between 1830 and 2019. As a result, Figure 6.2 shows examples of the planar graphs for the years 1830, 1924 and 2019,

[5] Population development; live births, deaths and migration by region| CBS
[6] Areas of shrinkage and anticipation areas|CBS
[7] In Dutch: krimpgebieden or krimpregio's
[8] In Dutch: anticipeergebieden or anticipeerregio's

in which the position of a node is determined by the geographic coordinates of the town hall of the corresponding municipality. The set of Dutch municipalities in year k constitutes a temporal network $G(\mathcal{N}[k], \mathcal{L}[k])$, defined by the set $\mathcal{N}[k]$ of $N[k] = |\mathcal{N}[k]|$ nodes and set $\mathcal{L}[k]$ of $L[k] = |\mathcal{L}[k]|$ links. Each municipality in year k is represented by a unique node and the underlying topology is defined by the $N[k] \times N[k]$ symmetric adjacency matrix $A[k]$. Nodes i and j share a link (i.e. $a_{ij}[k] = 1$) if there are geographical interconnections between adjacent municipalities i and j via roads, railways, bridges or tunnels[9], which were available in year k, otherwise $a_{ij}[k] = 0$.

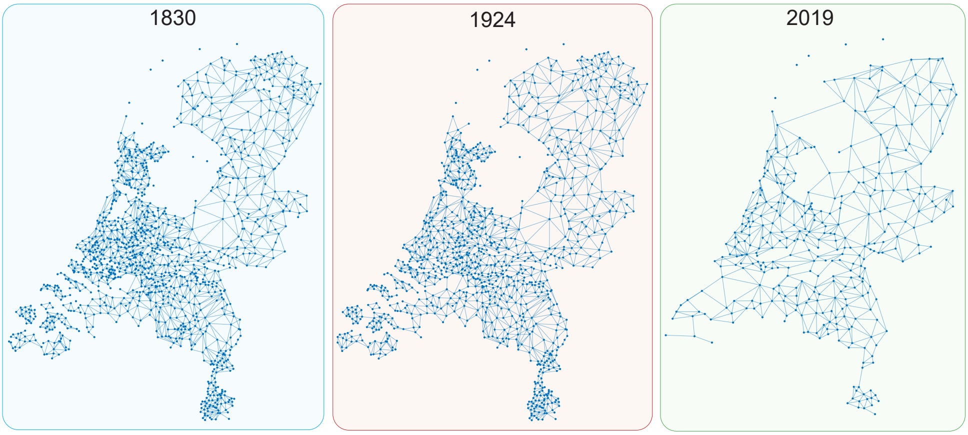


Figure 6.2: Dutch Municipality Network topology in the years 1830, 1924 and 2019.

In addition to the set of DMN graphs, a DMN research construct[10] was setup, containing the municipality area size, the population size and the merging data. Appendix E.1 describes the datasets used in this research. We applied two complementary municipality identification coding schemes to connect the yearly instances, as explained in Appendix E.2. The time series of data containing the population and area per municipality were collected from the International Institute of Social History[11] recorded in the Historical Database of Dutch Municipalities [117] and from Statistics Netherlands[12].

To better understand the survivability of Dutch municipalities, we analyse different underlying governing processes of the Dutch Municipality Network over time. Subsection 6.2.1 analyses the municipality network topology evolution per year, while the time

[9] If a pair of adjacent municipalities is exclusively connected in year k via water, we record $a_{ij} = 0$ in $A[k]$. Although there can be a ferry service connecting two adjacent municipalities, we record $a_{ij} = 0$ because a ferry service can connect more than two municipality nodes in contrast to one link exclusively connecting two nodes. Another characteristic that complicates analysis is the fact that some ferry services are not available during an entire year k.

[10] The DMN research construct comprises: (I) from 1830 on, the municipality area and merging data for each year k, (II) from 1851 on, the population vectors for each year k and (III) two population vectors derived from the 1809 and 1830 censuses.

[11] In Dutch: Internationaal Instituut voor Sociale Geschiedenis.

[12] In Dutch: Centraal Bureau voor de Statistiek (CBS).

dynamics of the area and population distribution per municipality are inspected in subsections 6.2.2 and 6.2.3, respectively.

6.2.1. TOPOLOGY EVOLUTION OVER TIME

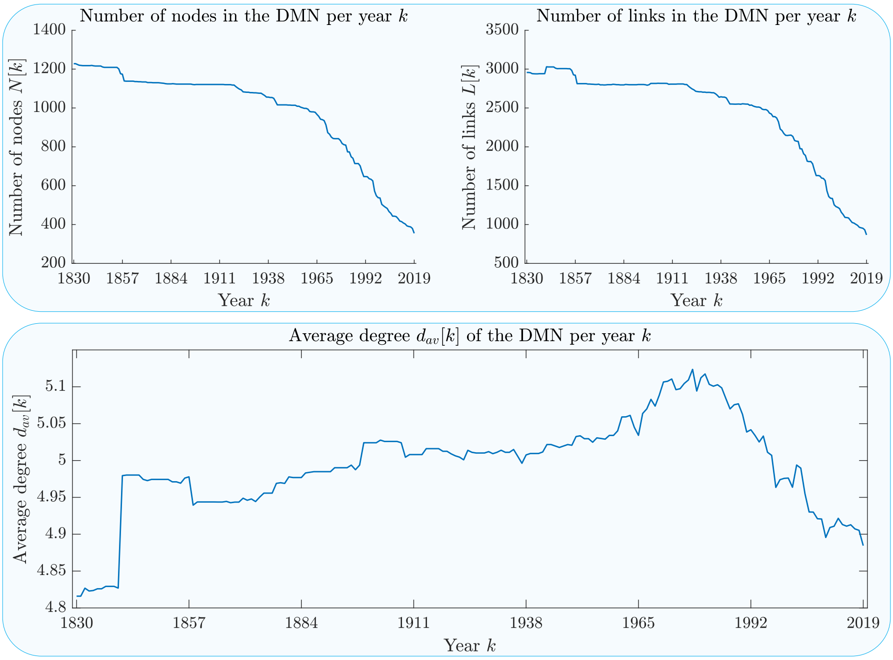


Figure 6.3: Number of nodes $N[k]$ (upper left-hand side), number of links $L[k]$ (upper right-hand side) and average degree $d_{av}[k]$ (lower part) in the DMN during the period 1830 – 2019.

When municipality i merges into an adjacent municipality j in year k, municipality node i disappears and becomes inactive in the $k+1$ instance of the DMN. When a new municipality is created in year k, an additional municipality node appears and becomes active in the $k+1$ instance of the DMN. The upper left-hand side of Figure 6.3 shows a decrease of 873 municipality nodes $N[k]$ as a results of municipality merging process between 1830 and 2019. In addition, the right-hand side of Figure 6.3 depicts the number of links $L[k]$ evolution over the researched period. The average degree $d_{av}[k] = \frac{1}{N[k]}\sum_{j=1}^{N[k]} d_j[k] = 2\frac{L[k]}{N[k]}$, with $d_j[k] = \sum_{i=1}^{N[k]} a_{ij}[k]$ denoting the degree of the j-th node in year k, remained almost unchanged during the research period, as depicted in the lower part of Figure 6.3

$$d_{av}[k] \approx 5,\ k \in \{1830, \ldots, 2019\}. \tag{6.1}$$

In other words, a typical Dutch municipality in the period 1830 – 2019 was surrounded by five neighbouring municipalities on average. Appendix E.4 provides a conservation law for the average degree $d_{av}[k]$ on a planar geographical network. The conservation equation (E.1) explains the changes in $d_{av}[k]$ over the year k as shown in the lower part of Figure 6.3. When pairs of municipalities merge, the average degree $d_{av}[k]$ slightly increases. Upward spikes in $d_{av}[k]$ occur in the DMN when newly built infrastructure[13] connects pairs of municipalities which were previously separated by water. However, in the period after 1960, the merging process intensified and often took place in waves which involved multiple municipalities per merger. As shown in Appendix E.4, the conservation relation (E.2) indicates a decreasing trend in $d_{av}[k]$, when mergers of clusters of three or more municipalities occur. As a result, $d_{av}[k]$ started decreasing after 1975.

6.2.2. AREA PER DUTCH MUNICIPALITY

In this subsection, we consider area measurements per municipality in the period 1830–2019 as realisations of the area random variable S of a municipality and examine how the area distribution per municipality changed over time. We show that the random area per municipality on a logarithmic scale, denoted by $Y = \log S$, allows for better insight into the underlying governing processes compared to a linear scale.

The area of each Dutch municipality in year k is a component of the $N[k] \times 1$ vector $s[k]$, where $s_i[k]$ denotes the area of municipality i in year k. The average area per Dutch municipality in year k is denoted as $s_{av}[k]$

$$s_{av}[k] = \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} s_i[k] = \frac{1}{N[k]} \cdot u^T \cdot s[k], \tag{6.2}$$

where $u^T = [1, 1, \cdots, 1]$ is the all-one vector. The $N[k] \times 1$ vector $y[k]$ contains the logarithm of area per municipality:

$$y[k] = \begin{bmatrix} \log(s_1[k]) & \log(s_2[k]) & \ldots & \log(s_{N[k]}[k]) \end{bmatrix}^T. \tag{6.3}$$

The total land surface of The Netherlands increased due to the process of building dikes, creating polders and draining land from the North sea and (after 1932) the IJsselmeer[14]. Figure 6.4 shows that the national area size has increased by 9% between 1830 and 2019.

AREA DISTRIBUTION

The logarithm Y of the area of a typical Dutch municipality closely follows a Gaussian or normal distribution and a logistic or Fermi-Dirac[15] distribution [118], which are reviewed in Appendix E.5. Instead of applying lognormal and log-logistic distributions on the area random variable $S[k]$ in year k, the fitting accuracy with a normal and logistic distribution of the *logarithm* of the area random variable $Y[k] = \log(S[k])$ is higher. In Appendix E.6.1, we apply the Anderson-Darling (AD) and the Kolmogorov-Smirnov (KS) statistical tests to examine to which extent the hypothesis, that the random variable $Y[k]$ follows a Gaussian or a logistic distribution, holds.

[13] Due to road and railway infrastructure development the number of nodes in disconnected components of the DMN decreased from 191 in the year 1830 to only 5 disconnected island municipalities in the Waddenzee in the year 2019.

[14] The Flevoland province, established in 1986, has been created almost entirely from water and includes the municipalities of Almere, Zeewolde, Dronten, Lelystad, Noordoostpolder and the former island Urk.

[15] The Fermi-Dirac distribution was introduced to describe energy states of particles.

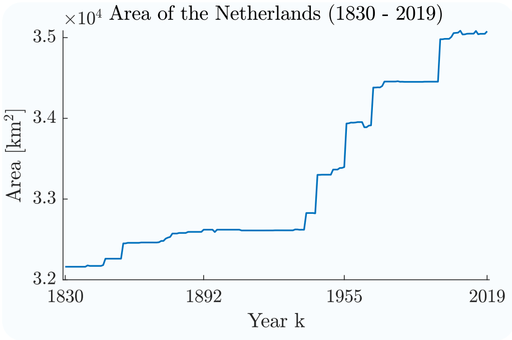


Figure 6.4: The total land surface of The Netherlands as the summation of the square kilometres from all municipalities over the period 1830 – 2019.

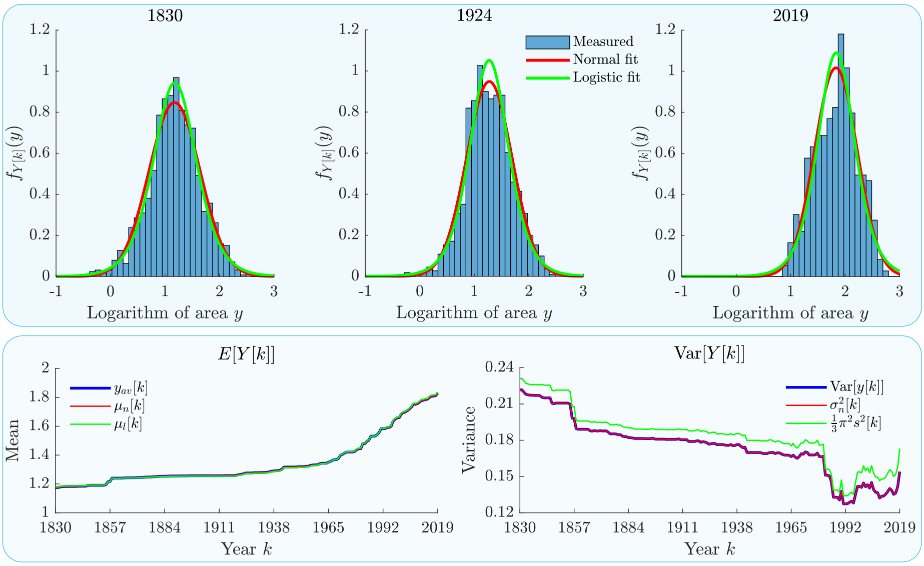


Figure 6.5: The probability density function $f_{Y[k]}(y)$ of the logarithm of the measured area $Y[k]$ per Dutch municipality (blue bars), fitted with a normal distribution (defined in (E.4); red colour) and a logistic distribution (defined in (E.9); green colour) for the years 1830, 1924 and 2019 (upper part). The mean $y_{av}[k]$ (lower left-hand side) and variance Var$[y[k]]$ (lower right-hand side) of the measured logarithm of area vector $y[k]$ versus the mean and the variance by the normal fit (red colour) and the logistic fit (green colour) in the period 1830 – 2019.

The upper part of Figure 6.5 illustrates the probability density function $f_{Y[k]}(y)$ of the logarithm of measured area per municipality (blue bars), fitted with a normal (red) and a logistic (green) distribution, for the years 1830, 1924 and 2019. The mean $\mu_n[k]$ and the variance $\sigma_n^2[k]$ of the normal distribution (defined in Section E.5.1), together with the mean $\mu_l[k]$ and variance $\sigma_l^2[k] = \frac{1}{3}\pi^2 s^2[k]$ of the logistic distribution (defined in Section E.5.3), per year k, are compared with the measured mean $y_{av}[k]$ and the variance $\text{Var}(y[k])$ in the lower part of Figure 6.5. The mean $E[Y[k]]$ of the logarithm of area $Y[k]$ is estimated equally precisely with a normal and logistic distribution. On the contrary, the variance $\text{Var}[Y[k]]$ is better fitted with a normal distribution. The lognormal distribution (defined in Sec E.5.2) possesses a weaker right tail than a log-logistic distribution (defined in Sec E.5.4), which follows more realistically the geographical boundary that areas of municipalities obey. In general, the area of a municipality can increase only at the cost of another municipality annexation, because the total area is almost[16] constant over time.

As will be derived in (6.11) in Section 6.3.2, the mean[17] $y_{av}[k]$ monotonically increases over time due to the merging process, with a pace depending on the merging rate and the area of abolished municipalities. The variance $\text{Var}(y[k])$ mainly decreases over time, except in the last two decades, where fluctuations occur. The decreasing trend of $\text{Var}(y[k])$ with time k reveals the nature of the merging process. In order to visualise how the merging process affected the distribution of the logarithm of the area, the probability density function $f_{Y[k]}(y)$ of the logistic distribution fit (left-hand side of Figure 6.6) of the logarithm of random area $Y[k]$ and the probability density function $f_{S[k]}(s)$ of the log-logistic distribution fit (right-hand side of Figure 6.6) of the area random variable $S[k]$ for each year k in the period 1830 – 2019.

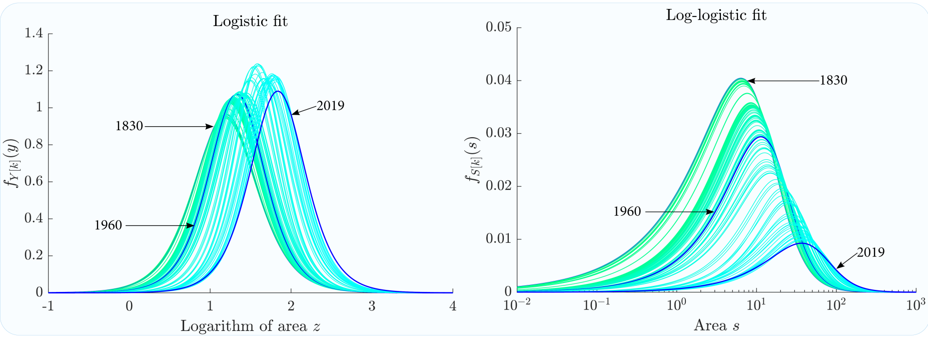


Figure 6.6: Probability density functions $f_{Y[k](y)}$ of the logistic fit of the logarithm of the area distribution in the period 1830 – 2019 (left-hand part). Probability density functions $f_{S[k]}(s)$ of the log-logistic fit of the area distribution in the period 1830 – 2019 (right-hand part).

The left-hand side of Figure 6.6 reveals that due to the merging process, municipalities are predominantly abolished from the left-hand side of the distribution curve and were annexed by a neighbouring municipality with a larger area. As a result, the left-hand side of the distribution curve is constantly shifting towards the right-hand side

[16]Total area of the mainland of The Netherlands is constant over time, except for the newly built land, as presented in Figure 6.4.

[17]The mean $y_{av}[k] = \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} \log(s_i[k]) = \log\left(\prod_{i=1}^{N[k]} (s_i[k])^{\frac{1}{N[k]}}\right)$ represents the logarithm of the geometric mean of the $N[k] \times 1$ area vector $s[k]$.

at a faster pace than the right-hand side of the distribution[18]. Therefore, the variance $\mathrm{Var}(y[k])$ decreases, while the mean $y_{av}[k]$ increases over time. The merging process reduces the diversity of municipalities in area size, while the fluctuations in $\mathrm{Var}(y[k])$ indicate the outliers (such as island municipalities) on the left tail.

6.2.3. POPULATION PER DUTCH MUNICIPALITY

In this subsection, we analyse the population distribution per municipality over time, where the collected population values per municipality are considered a realisation of the population random variable. Similar to the area size in Section 6.2.2, we find that the population random variable P reveals less information about underlying governing processes than its logarithm $Z = \log P$.

The population of Dutch municipalities in year k is represented by the $N[k] \times 1$ vector $p[k]$, where the population of municipality i in year k is denoted by $p_i[k]$. The total Dutch population $T[k]$ in year k is obtained by summing the population of each active municipality

$$T[k] = \sum_{i=1}^{N[k]} p_i[k], \tag{6.4}$$

or $T[k] = u^T p[k]$. The $N[k] \times 1$ vector $z[k]$ contains the logarithm $z_i[k] = \log(p_i[k])$ of the population of municipality i in year k,

$$z[k] = \begin{bmatrix} \log(p_1[k]) & \log(p_2[k]) & \dots & \log(p_{N[k]}[k]) \end{bmatrix}^T. \tag{6.5}$$

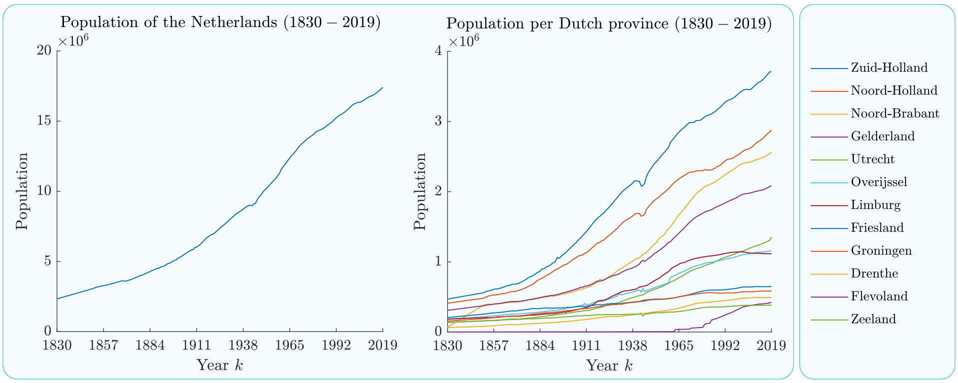


Figure 6.7: Population development of The Netherlands (left-hand side figure) and population development per Dutch province (right-hand side figure) in the period 1830 – 2019.

In 1830, The Netherlands had a population of 2.33 million people living in 1048 municipalities, which increased to 17.41 million citizens in 2019. Although the total population of The Netherlands, shown in Figure 6.7, has steadily increased during the period

[18]While the left distribution tail is shifted towards the right-hand side over time because municipalities with the relatively small area are abolished, the right distribution tail is shifted due to the increase in the area of municipalities that absorbed the abolished ones.

1830 – 2019, the population increase per province significantly varies (right-hand side of Figure 6.7). The impact of the Second World War on the population per Dutch province also varies in intensity: the population of the provinces South Holland[19] and North Holland[20] temporarily decreased most significantly.

POPULATION DISTRIBUTION

The logarithm of the Dutch municipality population random variable $Z[k]$ closely follows a normal and a logistic distribution in the period 1830 – 2019. Similarly, the population random variable $P[k]$ follows a lognormal and a log-logistic distribution. The

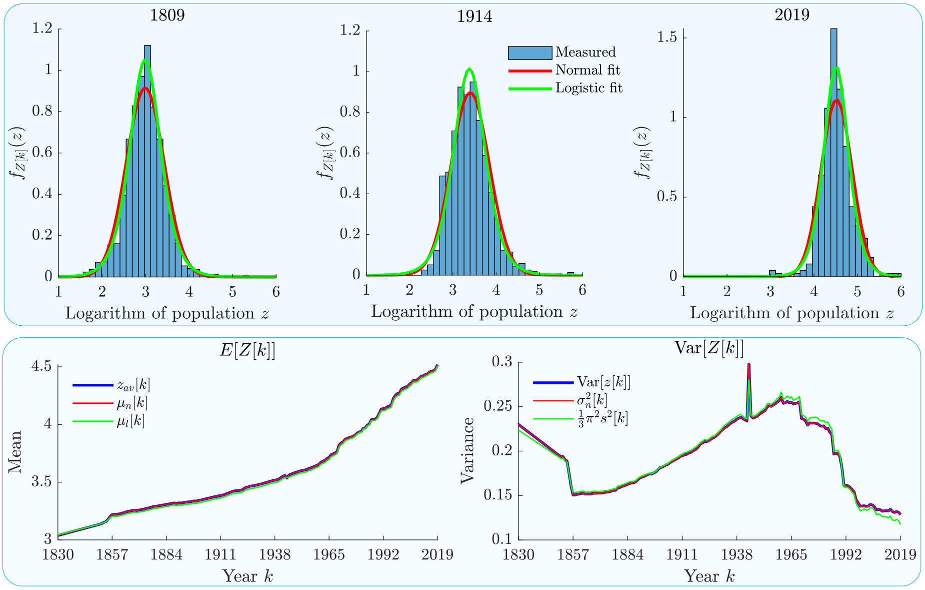


Figure 6.8: The probability density function of the logarithm of the population vector $Z[k]$ per Dutch municipality (blue bars), fitted with a normal distribution (defined in (E.4); red colour) and a logistic distribution (defined in (E.9); green colour) for the years 1830, 1914 and 2019 (upper part). Mean $z_{av}[k]$ (left-lower part) and variance Var$[z[k]]$ (right-lower part) versus the mean and the variance by the normal fit (red colour) and the logistic fit (green colour) in the period 1830 – 2019.

upper part of Figure 6.8 depicts the probability density function $f_{Z[k]}(z)$ of the logarithm of the population per Dutch municipality $Z[k]$ (blue bars), fitted with a normal (red) and a logistic (green) distribution, for the years 1830, 1914 and 2019. In Appendix E.6.2, we apply the Anderson-Darling (AD) and the Kolmogorov-Smirnov (KS) statistical tests to examine to which extent the hypothesis of the random variable $Z[k]$ following a normal or a logistic distribution holds.

To understand how the population distribution evolved over the researched period, we analyse the mean $E[Z[k]]$ and the variance $E\left[(Z[k]-E[Z[k]])^2\right]$ trends over time.

[19] In Dutch: Zuid-Holland

[20] In Dutch: Noord-Holland

The lower left-hand side of Figure 6.8 illustrates the average logarithm of the population $z_{av}[k] = \frac{1}{N[k]} \sum_{i=1}^{N[k]} z_i[k]$ in the period 1830 – 2019, together with the mean $E[Z[k]]$ of the normal fit $\mu_n[k]$ (red colour) and the logistic fit $\mu_l[k]$ (green colour). Both considered distributions precisely fit the measured mean $z_{av}[k]$ over time. The monotonically increasing mean $z_{av}[k]$ reveals the national population growth, but also comprises the effects of the migration and merging process, as will be discussed in Section 6.3.

The variance Var(z[k]) $= \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} (z_{av}[k] - z_i[k])^2$ over time is compared in the lower right-hand side of Figure 6.8 with the expected variance $E\left[(Z[k] - E[Z[k]])^2\right]$ of the normal fit $\sigma_n^2[k]$ (red colour) and of the logistic fit $\frac{1}{3}\pi^2\sigma_l^2[k]$ (green colour). Different trends of the variance Var(z[k]) over time reveal opposite migration patterns[21] across the geographical network of municipalities, which is derived in Section 6.3.

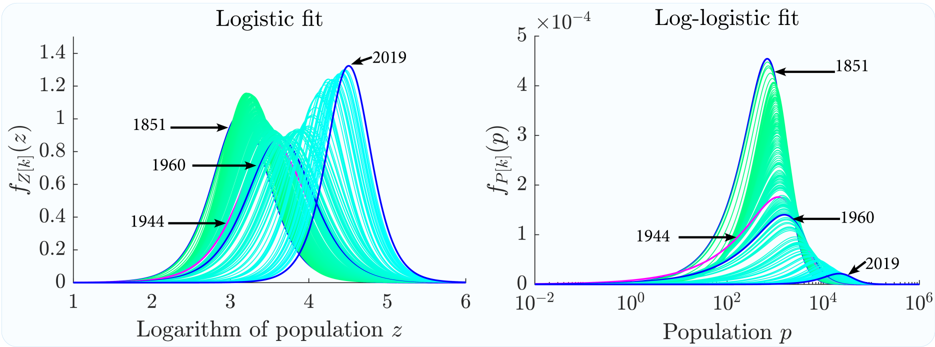


Figure 6.9: Probability density functions $f_{Z[k]}(z)$ of the logistic fit of the logarithm of the population distribution in the period 1830 – 2019 (left-hand part). Probability density functions $f_{P[k]}(p)$ of the log-logistic fit of the population distribution in the period 1830 – 2019 (right-hand part).

The left-hand side of Figure 6.9 illustrates the probability density functions $f_{Z[k]}(z)$, from annual logistic fits of the logarithm of the population $Z[k]$ in the period 1830–2019. On the right-hand side of Figure 6.9, we provide the probability density functions $f_{P[k]}(p)$ of the log-logistic annual fit of the population random variable $P[k]$. Figure 6.9 reveals the following general trends:

- The mode of the probability density function mode$(f_{Z[k]}(z)) = \frac{1}{\sigma_l[k]}$ is inversely proportional to the square root of variance $\sqrt{\text{Var}(Z[k])}$, as the enclosed surface under an bell-shaped curve obeys $\int_{-\infty}^{\infty} f_{Z[k]}(z)dz = 1$. Therefore, the increasing mode$(f_{Z[k]}(z))$ over time reflects a decreasing diversity in population on a logarithmic scale.

- Both probability density functions $f_{Z[k]}(z)$ and $f_{P[k]}(p)$ are continuously shifted to the right-hand side during the entire researched period, reflecting an almost 8 times increase in the total population of The Netherlands between 1830 and 2019 (see left-hand side of Figure 6.7).

[21]We refer here to two opposite migration flows, from small(er) to large(r) municipalities and from large(r) to small(er) municipalities, as defined in the Introduction.

Since the mean $E[Z[k]]$ and the variance $\text{Var}[Z[k]]$ of the logarithm of the random population $Z[k] = \log P[k]$ are fitted precisely by both distributions, the difference lies in the deep tails, where normal and logistic distributions behave considerably different. As derived in (E.14), the probability density function $f_{Z[k]}(z)$ of a logistic distribution on a double logarithmic scale decays linearly with population p, while the probability density function $f_{Z[k]}(z)$ of a normal distribution decreases as a square function of the population p, as derived in (E.15). Linear decay in the probability density function $f_{Z[k]}(z)$ of the logistic distribution indicates that the population distribution in the deep tail follows a power-law distribution.

Power-Law Fitting

The population per random municipality $P[k]$ in year k follows a power law if it obeys the distribution function $F_{P[k]}(p) = \Pr(P[k] \geq p)$ obeys

$$F_{P[k]}(p) = \left(\frac{p}{p_{min}[k]}\right)^{-\tau[k]+1}, \tag{6.6}$$

where the distribution parameter $\tau[k]$ in year k is known as the exponent or scaling parameter, while $p_{min}[k]$ denotes the minimum population value in year k that obeys the power law. In empirical datasets, a power law often fits only a subset of a vector, explaining the rare occurrence of large outcomes [119]. The corresponding probability density function $f_{P[k]}(p) = \frac{dF_{P[k]}(p)}{dp}$ of the power-law distribution in (6.6) is as follows

$$f_{P[k]}(p) = C[k] \cdot p^{-\tau[k]}, \tag{6.7}$$

where $C[k] = (\tau[k]-1) \cdot p_{min}[k]^{(\tau[k]-1)}$ denotes the normalisation constant in year k. For each year k in the period 1830 – 2019, the distribution function $F_{P[k]}(p) = \Pr(P[k] \geq p)$ is fitted by a power-law distribution for a subset of medium and larger municipalities and the power law exponent $\tau[k]$ is estimated. Figure 6.10 shows the distribution function $F_{P[k]}(p) = \Pr(P[k] \geq p)$ for the years 1851, 1960 and 2019 on the left-hand side, while the percentage of the total number of municipalities, that approximately follow a power law over time, is drawn on the upper right-hand part. In the lower right-hand part of Figure 6.10, the power law exponent $\tau[k] \in (2.1, 2.7)$ roughly decreases up to 1960 and increases after 1960. Apparently, during the Dutch urbanisation period, featured by a dominant migration flow of people towards large(r) municipalities, the power law exponent $\tau[k]$ decreased towards about $\tau[1961] = 2,17$ at 1961. Subsequently, the opposite migration flow dominated after 1960 and caused an increasing trend in the power law exponent $\tau[k]$. Probability functions in years 1851, 1960 and 2019, presented on the left-hand side of Figure 6.10, depict the effect of different migration flows on the population distribution per municipality.

In this section, we showed that the average degree $d_{av}[k]$ slightly increases when two municipalities are merged while $d_{av}[k]$ decreases due to a merger of more than two municipalities. Irrespective of the merger type, the average area size per municipality on a logarithmic scale $y_{av}[k]$ monotonically increased. In contrast, the variability in the area size $\text{Var}(y[k])$ decreased over the entire research period. Multiple underlying processes

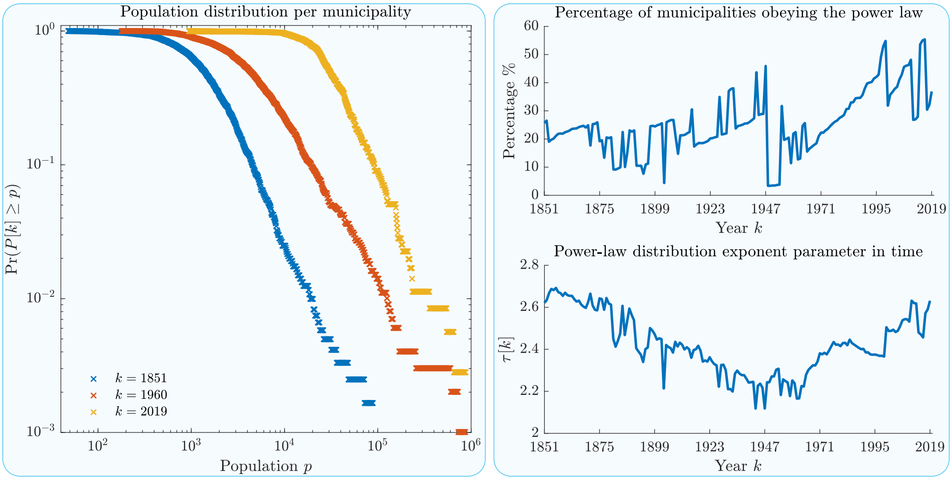


Figure 6.10: Distribution function $\Pr(P[k] \geq p)$ of the population per municipality in years 1851, 1960 and 2019 (left-hand side). The percentage of the $N[k] \times 1$ vector $p[k]$ that is fitted by a power-law fit in the period 1851 – 2019 (upper right-hand part). Estimated exponent $\tau[k]$ of the power-law fit of population distribution per municipality in the period 1851 – 2019 (lower right-hand part).

govern the time dynamics of the population. The increasing trend of the average logarithm $z_{av}[k]$ of population per municipality reveals a national population increase, but also the opposite trends of variability in population size $\text{Var}(z[k])$.

6.3. Dynamic Processes on the Dutch Municipality Network

In this section, we identify underlying processes in the Dutch municipality network and how they impacted population and area distribution in the period 1830 – 2019. Overall, we show that taking the logarithmic of the relevant quantities simplifies the analysis of the governing processes, which is a quite remarkable observation[22]. While both population and area distributions are heavy-tailed on a linear scale, they are bell-shaped on a logarithmic scale. Thus, we find that the mean and variance on a logarithmic scale describe the population and area distribution more precisely than on a linear scale.

6.3.1. Municipality merging process

At the end of each year k, a number – possibly none – of municipalities is abolished and annexed by one or more neighbouring municipalities. We denote the set of abolished municipalities at the end of year k as $\mathscr{A}[k]$, with the number of abolished municipalities denoted by $N_a[k] = |\mathscr{A}[k]|$. The evolution of the number of abolished municipalities

[22] Often human behaviour seems to follow a lognormal distribution as in Twitter [120] and online social platforms like Digg [121]. To the best of our knowledge, there is no rigorous theory of why the *logarithm* of quantities related to human behaviour (as here population and area) appears so often.

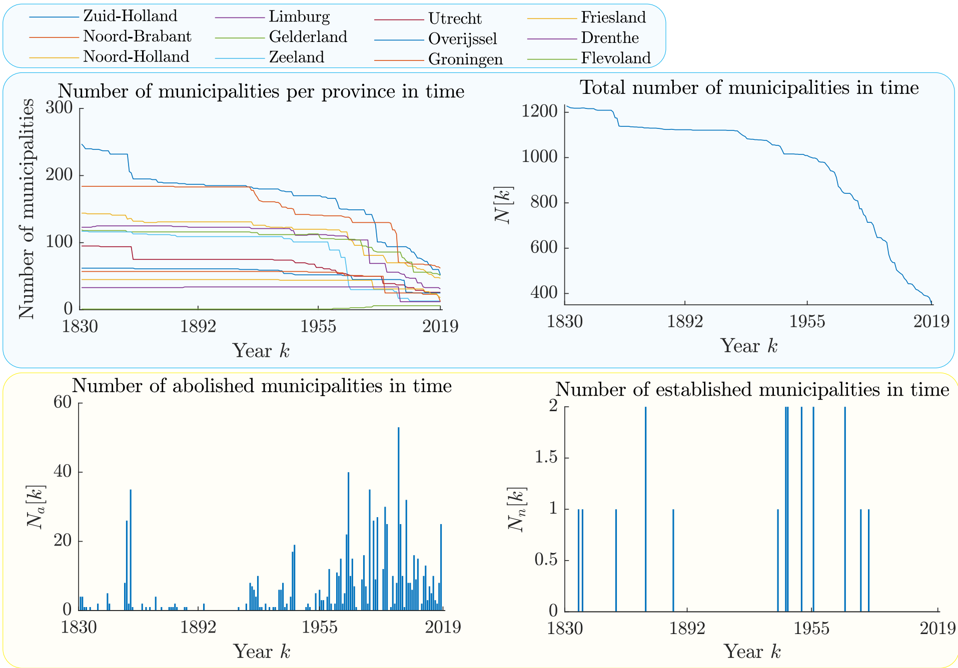


Figure 6.11: Number of municipalities per each of the 12 Dutch provinces in the period 1830 – 2019 (upper left-hand side). The total number of Dutch municipalities $N[k]$ in the period 1830 – 2019 (upper right-hand side). Number of abolished $N_a[k]$ (lower left-hand side) and newly established municipalities $N_n[k]$ (lower right-hand side) in the period 1830 – 2019.

$N_a[k]$ over time is depicted in the lower left-hand side of Figure 6.11. Appendix E.3 provides an overview of the merger types in the Dutch Municipality Network during the research period. The merging process became most intensive in the second part of the 20th century, decreasing population and area diversity while increasing the average size per municipality.

Newly established municipalities additionally modify the Dutch Municipality Network topology over time. The total area of The Netherlands increased since 1830 due to the reclaimed land from the sea on which new municipalities have been established. We denote the number of newly established municipalities at the end of year k as $N_n[k]$. The lower right-hand side of Figure 6.11 depicts how often new municipalities were established in the period 1830 – 2019. The evolution of the number of municipalities $N[k]$ over time, as presented in the upper left-hand side of Figure 6.11, obeys the following conservation law

$$N[k+1] = N[k] + N_n[k] - N_a[k]. \tag{6.8}$$

However, since very few new municipalities[23] have been established during 1830 – 2019,

[23] As presented in the lower right-hand side of Figure 6.11, since 1830 until 2019 in total $\sum_{i=1830}^{2019} N_n[i] = 19$ new municipalities have been established.

the municipality merging process predominantly drives the changes in the DMN topology. Thus, for the following analysis, we approximate (6.8) as

$$N[k+1] \approx N[k] - N_a[k]. \tag{6.9}$$

The difference equation (6.9) appears commonly in literature and can be solved by iteration[24] over k. The general exact solution is found via generating functions (see, e.g. [122, p. 123]).

6.3.2. GOVERNING PROCESSES OF THE AREA DYNAMICS

The area distribution per municipality evolves due to merging and establishing new municipalities. Since the latter occurs relatively rarely, we focus on how the merging process impacts the area distribution. We show that the area dynamics on a linear scale depend solely on the number of abolished municipalities $N_a[k]$, while the analysis on a logarithmic scale reveals additional information about the merging process.

In Figure 6.12, we provide the mean $s_{av}[k]$ (upper left-hand side) and the variance $\mathrm{Var}(y[k])$ (lower left-hand side) of the $N[k] \times 1$ area vector $s[k]$, as well as the mean y_{av} (upper right-hand side) and the variance $\mathrm{Var}(y[k])$ (lower right-hand side) of the $N[k] \times 1$ logarithm of area vector $y[k]$ in the period 1830 – 2019.

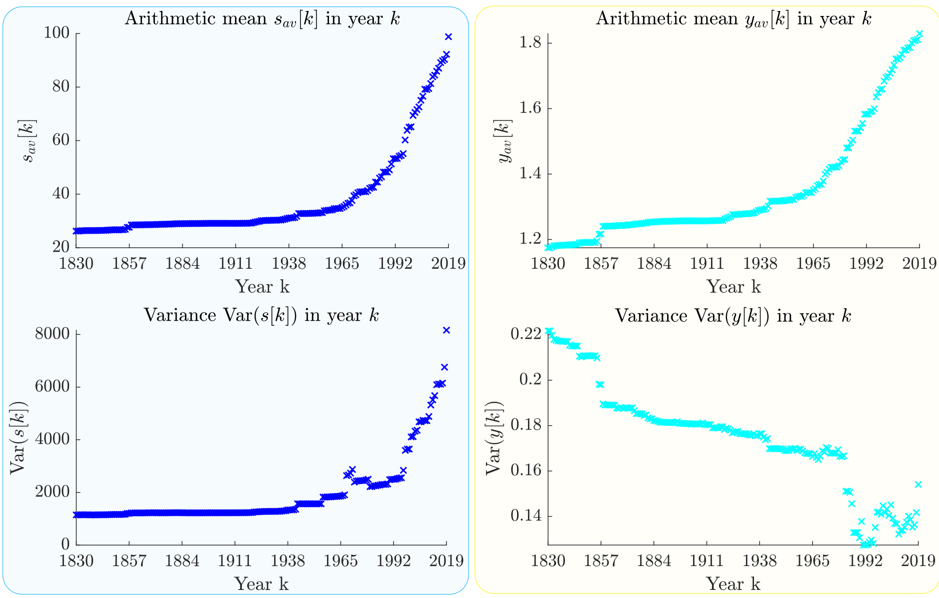


Figure 6.12: Mean $s_{av}[k]$ (upper left-hand part) and Variance $\mathrm{Var}(s[k])$ (lower left-hand part) of the $N[k] \times 1$ area vector $s[k]$ in the period 1830 – 2019. Mean $y_{av}[k]$ (upper right-hand part) and Variance $\mathrm{Var}(y[k])$ (lower right-hand part) of the $N[k] \times 1$ logarithm of area vector $y[k]$ in the period 1830 – 2019.

[24] By iteration over k, we obtain $N[k] = N[m] - \sum_{j=p}^{m-1} N_a[j]$, where we assume that year $m < k$ is known or is the initial condition.

We consider a merger case where $N_a[k] = |\mathscr{A}[k]|$ abolished municipalities in year k are annexed by an existing municipality $\eta \in \mathscr{N}[k]$. The set of municipalities in the following year becomes $\mathscr{N}[k+1] = \mathscr{N}[k] \setminus \mathscr{A}[k]$, where $\setminus$ denotes the set difference. As a result of a municipality merger, the area of the annexing municipality η grows as $s_\eta[k+1] = s_\eta[k] + \sum_{i\in\mathscr{A}[k]} s_i[k]$ in the next year $k+1$. The average area $s_{av}[k+1]$ in the following year $k+1$ evolves as follows

$$s_{av}[k+1] = \left(1 + \frac{N_a[k]}{N[k]-N_a[k]}\right) \cdot s_{av}[k]. \tag{6.10}$$

The mean $s_{av}[k]$ over time is presented in the upper right-hand side of Figure 6.12. From combining (6.9) and (6.10), we observe that the mean $s_{av}[k]$ in year k is inversely proportional to the number of municipalities $N[k]$

$$\frac{s_{av}[k+1]}{s_{av}[k]} = \frac{N[k]}{N[k+1]},$$

thus revealing solely the information about the intensity of the merging process over time. On the contrary, the conservation law for the average $y_{av}[k]$ of the $N[k]\times 1$ vector $y[k] = \log(s[k])$ of the area $s[k]$ per Dutch municipality in year k is

$$y_{av}[k+1] = \left(1 + \frac{N_a[k]}{N[k]-N_a[k]}\right) \cdot y_{av}[k] + \frac{1}{N[k]-N_a[k]} \log\left(\frac{\sum\limits_{i\in\eta\cup\mathscr{A}[k]} s_i[k]}{\prod\limits_{j\in\eta\cup\mathscr{A}[k]} s_j[k]}\right), \tag{6.11}$$

as derived in Appendix E.7.1. The second term in (6.11) reveals additional information about the mergers, compared to the conservation law in (6.10). The increase of the mean $y_{av}[k]$ over time, as depicted in the upper right-hand side of Figure 6.12, is bounded by the second term in (6.11). From the left-hand side of Figure 6.6, we observe that the left distribution tail of the logarithm of the area $Y[k]$ is impacted mostly after 1960, indicating that the municipalities with the smallest areas were often abolished. Mergers involving municipalities from the left distribution tail cause the second term in (6.11) to increase in value and consequently increase the mean $y_{av}[k]$ at a larger pace than before 1960. From the upper right-hand side of Figure 6.12, we clearly distinguish two linear patterns over time, until and after 1960.

From the decreasing trend of the variance $\mathrm{Var}(y[k])$ over time, presented in the lower right-hand side of Figure 6.12, we observe that the merging process continuously lowered the area size diversity of Dutch municipalities on a logarithmic scale. In other words, the area size of a municipality negatively correlates with the probability of its abolishment, taking into account the increasing trend of the mean $y_{av}[k]$. Therefore, the municipality area could be considered a predictor of the probability of municipality abolishment.

Since the area and population distribution per Dutch municipality follow the same distribution model, the insights into how the merging process impacted the area distribution also hold for the population, allowing for recognising the impact of other governing processes, such as population growth and people migration.

6

6.3.3. GOVERNING PROCESSES OF THE POPULATION DYNAMICS

In this section, we analyse how the population increase, population migration between municipalities and the process of municipality merging determined the population distribution per municipality.

In Figure 6.13, we present the mean $p_{av}[k]$ (upper left-hand side) and the variance $\text{Var}(p[k])$ (lower left-hand part) of the $N[k] \times 1$ population vector $p[k]$. In addition, we depict the mean $z_{av}[k]$ (upper right-hand side) and the variance $\text{Var}(p[k])$ (lower right-hand side) of the $N[k] \times 1$ logarithm of the population vector $z[k]$ in the period 1830 – 2019.

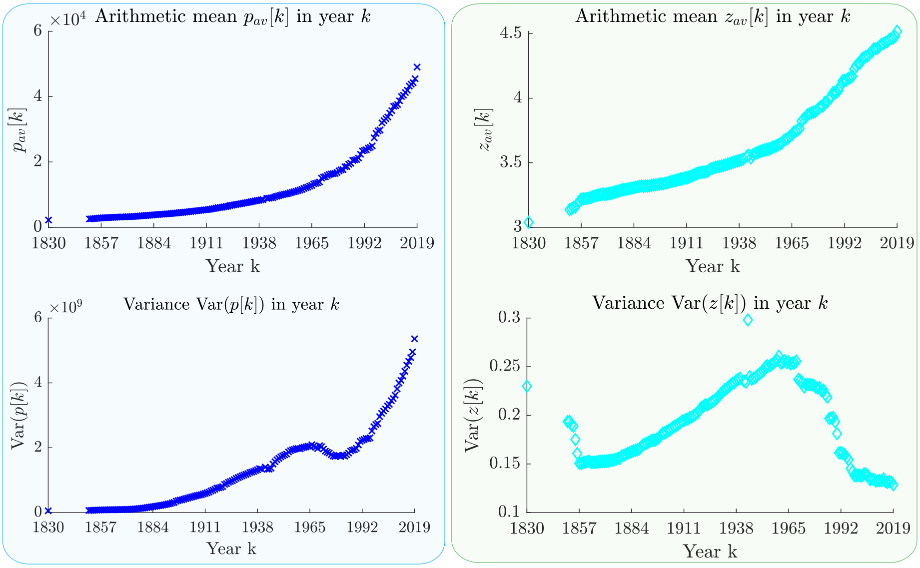


Figure 6.13: Mean $x_{av}[k]$ (upper left-hand part) and Variance $\text{Var}(x[k])$ (lower left-hand part) of the $N[k] \times 1$ population vector $x[k]$ in the period 1830 – 2019. Mean $z_{av}[k]$ (upper right-hand part) and Variance $\text{Var}(z[k])$ (lower right-hand part) of the $N[k] \times 1$ logarithm of population vector $z[k]$ in the period 1830 – 2019.

The trends of both the mean $z_{av}[k]$ and the variance $\text{Var}(z[k])$ in Figure 6.13 can be approximated by a two-segment linear function of time k, before and after 1960. Further, the variance[25] of the logarithm of the population vector $\text{Var}(z[k])$ peaks around 1960 and starts decreasing afterwards, revealing a change in the dynamic pattern of the underlying processes. A decreasing trend of the variance $\text{Var}(z[k])$ coincides with the intensified municipality merging process that took place after 1960, as presented in Figure 6.11. Another underlying process governing both the mean $z_{av}[k]$ and the variance $\text{Var}(z[k])$ over time is the population evolution per municipality.

[25] The variance $\text{Var}(z[k])$ spikes in the year $k = 1944$ due to the Second World War, and this year represents an outlier in the time dynamics of the DMN population distribution.

POPULATION RANK-SIZE DISTRIBUTION

The population distribution of a country's large(r) cities often follows Zipf's Law, which reveals a relationship between the frequency and size of a set [123]. We analyse the rank-size distribution of the population per Dutch municipality in the period 1830 – 2019. In each year k, the population vector $p[k]$ rank-size distribution is fitted with a linear function on a double logarithmic scale. The absolute value of the slope of the population rank-size distribution we denote as the population rank-size slope $\beta[k]$. In the upper part of Figure 6.14, we provide the population rank-size distribution per municipality, together with the fitted line on a double logarithmic scale for the years 1830, 1920 and 2010. In addition, we depict the slope $\beta[k]$ of the linear fit (lower left-hand part) and the population of Amsterdam $p_A[k]$ in the period 1830 – 2019.

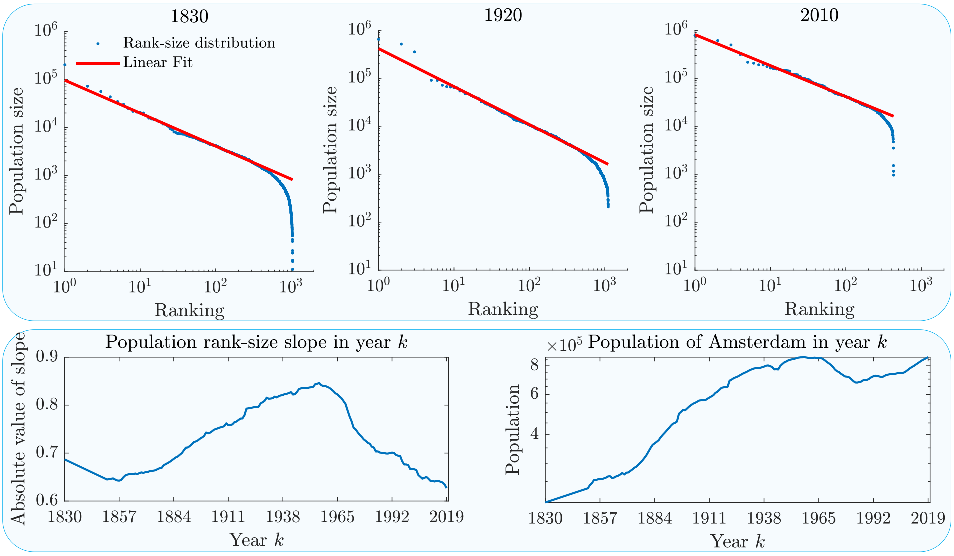


Figure 6.14: Population rank-size distribution per municipality and the linear fit in double logarithmic scale, in years 1830, 1920 and 2010 (upper part). Estimated slope of the population rank-size distribution in the period 1809 – 2019 (lower left-hand part). Population of Amsterdam in the period 1830 – 2019 (lower right-hand part).

A general trend of vertical movement[26] of the point cloud in the upper part of Figure 6.14 reflects the population increase over time, as we explain in Section 6.4.1. The rank-size distribution slope $\beta[k]$ over time reveals two opposite dynamic trends of the population evolution per Dutch municipality in the period 1851 – 2019. Since 1851, the rank-size slope $\beta[k]$ continuously increased and peaked at $\beta[1960] = 0.846$ in 1960, from when it started decreasing. The population rank-size slope $\beta[k]$ in year k can be approximated as

$$\beta[k] \approx b_1 \cdot k + b_2, \tag{6.12}$$

[26] A vertical movement of the dots in the upper part of Figure 6.14 is similar to the horizontal movement of the logistic probability density function $f_{Z[k]}(z)$ over time, presented in Figure 6.9.

where the parameters b_1 and b_2 are estimated for the periods 1851–1960 and 1960–2019 as follows

$$\begin{aligned} b_1 &= 2\cdot 10^{-3}, \qquad b_2 = -3.23, k \in \{1851, 1959\} \\ b_1 &= -3.4\cdot 10^{-3}, b_2 = 7.47, \quad k \in \{1960, 2019\}. \end{aligned} \tag{6.13}$$

A positive increase in the population rank-size slope $\beta[k+1] - \beta[k]$ between two consecutive years k and $k+1$ in the period 1851 – 1960 informs us that larger municipalities increased in population size faster than smaller municipalities on a logarithmic scale. Here we introduce an assumption important for the following analysis. We assume that the population of each municipality approximately follows Zipf's Law. Therefore, from the rank-size distribution in Figure 6.14, the logarithm of municipality i population $z_i[k]$ in year k can be approximated as

$$z_i[k] \approx z_A[k] - \beta[k]\cdot \log r_i[k], \tag{6.14}$$

where $i \in \mathcal{N}[k]$ and municipality i ranking in the $N[k]\times 1$ population vector $p[k]$ in year k is denoted as $r_i[k]$, while the logarithm of the population in Amsterdam (i.e. the largest Dutch municipality by population) in year k is denoted as $z_A[k] = \log(p_A[k])$. When assuming that the ranking of municipality i in the $N[k]\times 1$ population vector $p[k]$ does not change $r_i[k+1] = r_i[k]$ in two consecutive years k and $k+1$, we obtain the following governing equation

$$\frac{p_i[k+1]}{p_i[k]} = (r_i[k])^{-b_1}\cdot \frac{p_A[k+1]}{p_A[k]}, \tag{6.15}$$

as derived in Appendix E.7.2. The governing equation (6.15) of the population increase per municipality, for different values of the linear fit parameter b_1 in (6.13), reveal two opposite trends of people migration. Until 1960, people predominantly migrated from small(er) to large(r) municipalities. Consequently, municipalities with a large(r) population grew faster. In contrast, in the period after 1960, the largest municipalities in population no longer grew at a dominant pace, revealing the migration flow towards small(er) municipalities in population size. The two dynamic trends are also observable from the population of Amsterdam $p_A[k]$, presented on the lower right-hand side of Figure 6.14. Based on the governing equation (6.14), we derive the impact of the rank-size slope $\beta[k]$ over time onto the mean $z_{av}[k]$ of the logarithm of population vector $z[k]$

$$z_{av}[k] = z_A[k] - \beta[k]\cdot \log\left(N[k]!^{\frac{1}{N[k]}}\right). \tag{6.16}$$

which can be further simplified using Stirling's approximation [124, p. 257] of the Gamma function

$$z_{av}[k] \approx z_A[k] - \beta[k]\cdot\left(\log(N[k]) - \log(e) + \frac{1}{2N[k]}\cdot\log(2\pi N[k])\right). \tag{6.17}$$

Relation (6.16) explains two linear patterns in the mean $z_{av}[k]$ evolution over time, presented in the upper right-hand side of Figure 6.13. In the period 1830 – 1960, the increase in Amsterdam population dominantly impacted the mean z_{av}. In the next period

1960 – 2019, Amsterdam population saturated on a logarithm scale $z_A[k]$. However, both the rank-size distribution slope $\beta[k]$ and the number of municipalities $N[k]$ monotonically decreased in value, keeping the increasing trend of the mean $z_{av}[k]$.

The assumption introduced in (6.14) allows to derive the variance $\mathrm{Var}(z[k])$ as provided in Appendix E.7.2

$$\mathrm{Var}(z[k]) = \beta^2[k] \cdot g(N[k]), \tag{6.18}$$

where

$$g(N[k]) = \frac{1}{N[k]} \sum_{i=1}^{N[k]} \left(\log \frac{N[k]!^{\frac{1}{N[k]}}}{i} \right)^2,$$

explaining the behaviour of the Variance $\mathrm{Var}(z[k])$ over time. Since 1830 until 1960, the slope $\beta[k]$ monotonically increased, causing an increase of the variance $\mathrm{Var}(z[k])$. On the contrary, the decreasing trend of the slope $\beta[k]$ after 1960, together with the decreasing number of municipalities $N[k]$ due to the merging process, caused the $\mathrm{Var}(z[k])$ to decrease. Moreover, the aggressive merging process that took place after 1960 amplified the decreasing rate of the variance $\mathrm{Var}(z[k])$.

In Appendix E.7.3, we derive an explicit relation between the exponent of the power-law probability density function (depicted in Figure 6.10) and the rank-size distribution slope $\beta[k]$ (provided in the lower left-hand side of Figure 6.14)

$$\beta[k] = \frac{1}{\tau[k] - 1}. \tag{6.19}$$

6

Evolution of municipalities across fixed population size categories

To better understand how the population distribution changed over time, we analyse the evolution of municipalities over time across fixed population size categories. The first size category contains municipalities with less than 200 inhabitants. On the other side, the last category includes municipalities with more than 200.000 inhabitants. In between, we define equidistant intervals of the population size on a logarithmic scale for a given number of intervals. For each year k in the period 1830 – 2019, we correlate the percentage of the total population in The Netherlands with the ratio of the total number of municipalities per population size category. The correlation is presented in the upper part of Figure 6.15.

Except for the largest municipalities (i.e. those with more than 200.000 inhabitants), we observe a consistent correlation pattern in the two-dimensional space. A fixed population size category initially[27] contains no municipalities until their population sizes reach values determining the size category. Each category draws a path in the clockwise direction, as presented in Figure 6.15, and eventually tends back to the coordinate origin. On the contrary, a municipality advances upwards through adjacent categories as population size increases. When advancing, a municipality is the largest in the current category while becoming the new smallest element in the next larger category. Therefore, the impact of a municipality leaving the current category is negative and of considerably higher intensity than a municipality's positive impact upon entering the adjacent

[27] Under the assumption, we find an adequate time instant. However, for a given time instant, different categories are in different regions of their respective paths.

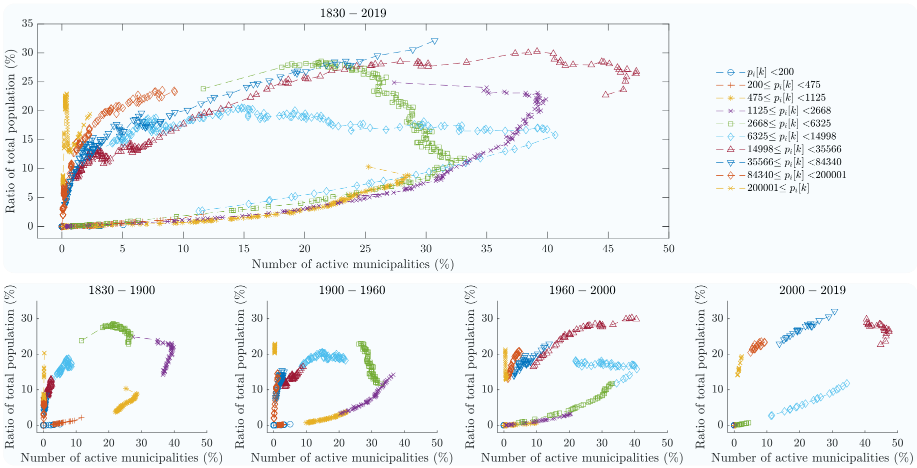


Figure 6.15: Correlation between the relative number of municipalities and the relative population per logarithmic equidistant population size category in the periods (lower part, from left to right-hand side) 1830 – 1900, 1900 – 1960, 1960 – 2000 and 200 – 2019, and during the entire researched period 1830 – 2019 (upper part).

larger category. Consequently, the correlation patterns in Figure 6.15 must always describe paths in the clockwise direction.

In the lower part of Figure 6.15, we present correlation patterns in different periods to analyse the impact of different underlying processes. The merging process negatively affects the abolished municipality's category, as its annexation is equivalent to removing that municipality from the corresponding group. On the contrary, the annexing municipality either positively influences the path of its size category or advances to a (possibly non-adjacent) larger-size group of municipalities. Indeed, the trajectory of smaller-size categories in the period 1830 – 1900 (first plot from the left) is considerably shorter than the trajectory in the following periods 1900 – 1960 (second plot) and 1960 – 2000 (third plot), respectively, which coincides with the merging intensification over time, as presented in Figure 6.11.

The dominant increase in the population ratio of the largest municipalities in the period 1830 – 1900 indicates an intensive migration of people from small(er) to large(r) municipalities, as presented in the lower left-hand side of Figure 6.15. On the contrary, the largest municipalities significantly decreased in population size in the period 1960 – 2000. The migration flow from large(r) to small(er) municipalities became dominant in this period. In addition, an intensive merging process took place, degrading small(er) size categories while further reinforcing the population increase of municipalities of large(r) sizes.

Finally, trajectories in Figure 6.15 enclose an area. Such phenomena can be explained by the fact that the distribution of the logarithm of population per municipality follows Gaussian distribution consistently over time, as depicted in Figure 6.8. In other words, the probability density function $f_{Z[k]}(z)$ defines a bell-shaped curve, being horizontally shifted over time. Therefore, a population-size category of municipalities firstly increases in the number of municipalities (and thus population ratio), peaks and starts decreasing, explaining the trajectories in Figure 6.15.

6.4. MODEL OF THE DUTCH MUNICIPALITY NETWORK

This section proposes a model which captures the time dynamics of the DMN. The purpose of the model is not to explain the evolution of a single Dutch municipality over time but rather to describe the evolution of the population and area distribution per Dutch municipality. The DMN model consists of three sequential sub-models:

1. Population increase model per municipality,
2. Inter-municipal migration model, and
3. Merging model,

6.4.1. POPULATION INCREASE MODEL

Available measurements of the population per municipality in the period 1830 – 2019 reveal a consistent correlation pattern between the population $p_i[k]$ of municipality i in year k and its increase $p_i[k+1]-p_i[k]$, which a linear function on a double logarithmic scale can approximate. In the upper part of Figure 6.16, we present the correlation[28] between values $p_i[k+1]-p_i[k]$ and $p_i[k]$, where $i \in \mathcal{N}[k]$, on a double logarithmic scale in years 1852, 1936 and 2019, respectively. A positive correlation can be approximated as follows

$$E\left[\log\left(P[k+1]-P[k]\right)\right] = c_1[k]\cdot E\left[Z[k]\right] + c_2[k], \tag{6.20}$$

where coefficients $c_1[k]$ and $c_2[k]$ represent the slope and additive constant of the linear fit in year k, as presented in the lower part of Figure 6.16. While the slope $c_1[k]$ slightly oscillates around 1 for a period of 130 years, the additive constant $c_2[k]$ decreases over time, allowing us to introduce the following approximations

$$\begin{aligned} c_1[k] &\approx 1 \\ c_2[k] &\approx 9.27 - 5.8\cdot 10^{-4}\cdot k. \end{aligned} \tag{6.21}$$

By choosing $c_1[k]=1$ and adopting the stronger assumption that the difference equation (6.20) holds not only for the mean, but also for the random variables themselves, i.e. $\log(P[k+1]-P[k]) = Z[k]$, we deduce that $\text{Cov}\left[\log\left(P[k+1]-P[k]\right), Z[k]\right] = \text{Var}\left[Z[k]\right]$, meaning that the variability in the difference (an increase of population in a random municipality $P[k]$) equals that of $P[k]$. The governing equation for the population increase model per municipality is obtained by importing (6.21) into (6.20):

$$E[P[k+1]] \approx \left(1+e^{c_2[k]}\right)\cdot E[P[k]]. \tag{6.22}$$

The slope $c_1[k]$ and the additive constant $c_2[k]$ values considerably oscillate in the last two decades of the researched time series. A reason that causes such behaviour is an

[28]A minor percentage of municipalities with negative population increase is not presented in Figure 6.16.

6

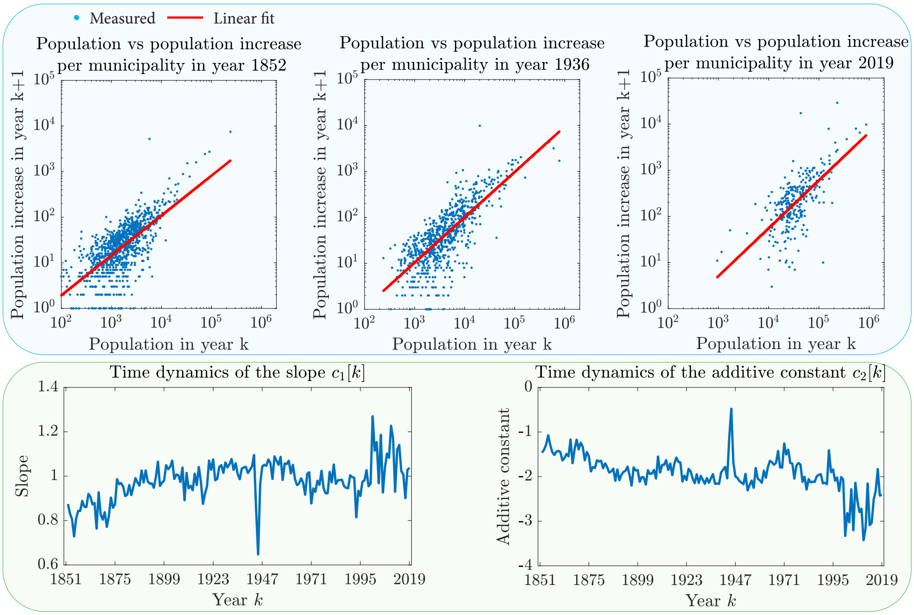


Figure 6.16: Population per municipality $p[k]$ in year k, versus population increase $(p[k+1]-p[k])$ in the following year $k+1$, on a double logarithmic scale for the years 1852, 1936 and 2019, together with a fitted linear function (upper-part). Time dynamics of the slope $c_1[k]$ and the additive constant $c_2[k]$ of a fitted line (lower part).

intensified merging process (see lower left-hand part of Fig 6.11) that took place in the mentioned period, as a result of which certain municipalities (in most cases with a relatively small population) are being abolished and annexed by a neighbour municipality with a larger population. Consequently, the population increase of annexing municipalities in the following year spikes. Indeed, on a closer look at Figure 6.16, the slope $c_1[k]$ (additive constant $c_2[k]$) exhibit only positive (negative) spikes during the last 20 years of the researched period, respectively, and return to the previous state in the subsequent year.

Under the assumption that the number of municipalities remains unchanged within two consecutive years, $N[k+1] = N[k]$, the mean $z_{av}[k]$ evolves due to the proposed population increase model in (6.22) as follows

$$z_{av}[k+1] = \frac{1}{N[k]} \sum_{i=1}^{N[k]} \left(\log\left(1+e^{c_2[k]}\right) + z_i[k]\right) = \log\left(1+e^{c_2[k]}\right) + z_{av}[k],$$

because a multiplicative increase on a linear scale is equivalent to an additive increase on a logarithmic scale. On the contrary, the variance $\mathrm{Var}(z[k])$

$$\mathrm{Var}(z[k+1]) = \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} \left(z_{av}[k+1] - \log\left(1+e^{c_2[k]}\right) - z_i[k]\right)^2 = \mathrm{Var}(z[k]),$$

is invariant to the population increase model in (6.22). An argument behind neglecting the slope $c_1[k]$ deviations around value 1 and adopting (6.21) is the idea of decoupling two population processes that occur on the DMN, namely the population increase and population migration. We argue that the variability in multiplicative population increases per municipality is a consequence of the people migrating between municipalities. Thus, in the following subsection, we introduce the migration model on a network that complements the population increase model (6.22).

6.4.2. INTER-MUNICIPAL MIGRATION MODEL

The population increase model in (6.22) reveals a common trend in population increase per Dutch municipality. Other simultaneous processes on the DMN are immigration/emigration and internal migration of people. In this subsection, we introduce a migration model of people on a geographical network and apply it to the Dutch Municipality Network.

We propose a linear model that captures the migration of people across a geographical network of municipalities and complements the population increase model. The proposed migration model is a diffusion-like process founded on the assumption that there are two opposite migration flows taking place simultaneously on a network:

- **Forward migration:** People moving from small(er) to large(r) municipalities in terms of population size, denoted as the forward migration flow with forward migration rate α. This migration flow became dominant during the urbanisation period, from approximately 1850 until 1960 (see Figure 6.18).
- **Backward migration:** People moving from large(r) to small(er) municipalities, denoted as the backward migration flow with backward migration rate δ. This migration flow became dominant after 1960.

6

We define the $N[k] \times N[k]$ migration matrix $M[k]$, with elements $m_{ij}[k]$

$$m_{ij}[k] = a_{ij}[k] 1_{\{E[p_i[k]] < E[p_j[k]]\}}, \tag{6.23}$$

with the indicator function denoted as 1_x, which is defined as 1 if the statement x is true, otherwise equals 0. Relation (6.23) transforms the undirected DMN into a directed network, in which each link points to the *adjacent* municipality with a larger population, from where we conclude

$$A[k] = M[k] + M^T[k].$$

The $N[k] \times N[k]$ migration matrix $M[k]$ allows for introducing a model of people migrating on a municipality network

$$E[P[k+1]] = \left(I + \alpha \cdot M^T[k] + \delta \cdot M[k] - \alpha \cdot \text{diag}(M[k] \cdot u) - \delta \cdot \text{diag}\left(M^T[k] \cdot u\right)\right) \cdot E[P[k]], \tag{6.24}$$

where the $N[k] \times N[k]$ matrix I denotes the identity matrix. Each matrix term in (6.24) allows for a physical interpretation. The $N \times N$ identity matrix I indicates that the proposed migration model describes an additive process. The second term $\alpha \cdot M^T[k]$ calculates arrivals of people per municipality due to the forward migration flow (i.e. from smaller to larger adjacent municipality). The same migration flow, away from a smaller adjacent municipality, is accounted for in the matrix term $\alpha \cdot \text{diag}\left(M[k]^T \cdot u\right)$. The third matrix term $\delta \cdot M[k]$ computes the arrivals of people per municipality due to backward migration (i.e. from large(r) to small(er) adjacent municipality). As each migration flow has an origin and a destination municipality, the number of departures due to the backward migration is accounted for by $\delta \cdot \text{diag}(M[k] \cdot u)$. The sum of the opposite forward and backward migration flows provides the resulting migration flow from municipality i to a larger adjacent municipality j (i.e. $p_j[k] > p_i[k]$) $\alpha \cdot p_i[k] - \delta \cdot p_j[k]$. In the particular case when $\alpha = \delta$, the governing equation (6.24) describes a diffusion process

$$E[P[k+1]] = (I - \alpha \cdot Q[k]) \cdot E[P[k]],$$

where the $N \times N$ Laplacian $Q = \text{diag}(d) - A$. Properties of the proposed migration model in (6.24) are provided in the Appendix E.8.

6.4.3. MERGING MODEL

The merging dynamics of the Dutch Municipality Network is a complex, government-controlled process that depends on numerous factors, such as economics, politics and social aspects, to name a few. Instead, we argue that all these aspects correlate with the population and area of municipalities. Thus, we model the Dutch municipal merging process by considering the population and area measurements per municipality and the network effect.

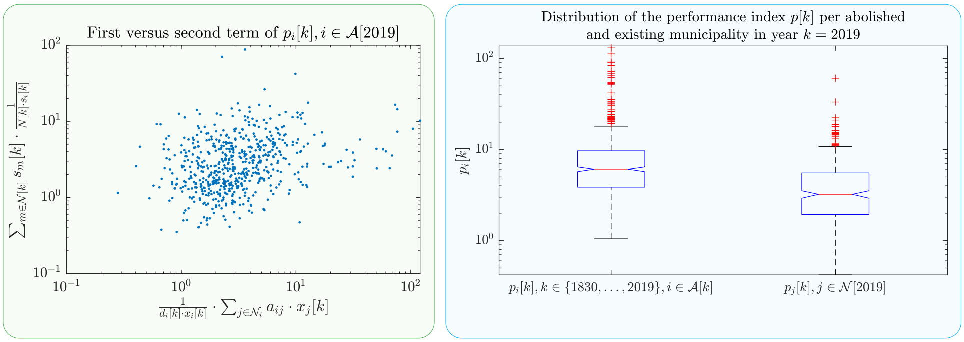


Figure 6.17: Correlation between the first and the second term of the Abolishment Likelihood index $x_i[k]$ per abolished municipality (i.e. $i \in \mathscr{A}[k]$) in the year of their abolishment (left-hand side figure). Distribution of the Abolishment Likelihood index $p[k]$ per abolished municipality in the year of their abolishment versus distribution of the Abolishment Likelihood index vector $x_i[k]$ per each municipality in year $k = 2019$ (right-hand side figure).

We propose an Abolishment Likelihood index per municipality that estimates the set of abolished municipalities $\mathscr{A}[k]$ in year k. The Abolishment Likelihood index of municipality i in year k, denoted as $x_i[k]$, is defined as

$$x_i[k] = \frac{1}{d_i[k] \cdot p_i[k]} \sum_{j \in \mathscr{N}_i[k]} a_{ij}[k] \cdot p_j[k] + \frac{1}{3} \cdot \frac{s_{av}[k]}{s_i[k]}, \tag{6.25}$$

with $\mathcal{N}_i[k]$ denoting the set of the node i neighbours in year k (i.e. $\mathcal{N}_i[k] = \{j \mid j \in \mathcal{N}[k], a_{ij}[k] = 1\}$). The first term of the Abolishment Likelihood index $x_i[k]$ in (6.25) compares the population $p_i[k]$ of municipality i with the mean population of its direct neighbours $\frac{1}{d_i[k]\cdot p_i[k]}\cdot\sum_{j\in\mathcal{N}_i[k]} a_{ij}[k]\cdot p_j[k]$, while the second term in (6.25) compares the area $s_i[k]$ of municipality i with the mean area $s_{av}[k]$ per Dutch municipality in year k. The set of abolished municipalities in year k, denoted by $\mathcal{A}[k]$, is determined as $N_a[k] = |\mathcal{A}[k]|$ municipalities with highest ranking in the $N[k]\times 1$ Abolishment Likelihood index vector $x[k]$.

On the left-hand side of Figure 6.17, we correlate terms of the Abolishment Likelihood index $x_i[k]$ in (6.25) per each abolished municipality i, in the year of its abolishment. The absence of correlation confirms the validity of our choice to consider both population and area size per municipality. In addition, the two box-whisker plots (abolished versus existing municipalities) on the right-hand side of Figure 6.17 are clearly shifted with respect to each other, confirming that the Abolishment Likelihood index indeed captures the abolishment likelihood.

6.4.4. Model validation

By combining the assumption from (6.21), the proposed migration model (6.24) and the merging model (6.25), we obtain a complete model for the time dynamics of the Dutch Municipality Network. The model is initialized by the $(N[k]\times N[k])$ adjacency matrix $A[k]$ of the DMN, the $(N[k]\times 1)$ population vector $p[k]$ and the $(N[k]\times 1)$ area vector $s[k]$ from year $k = 1851$. Starting from $k = 1852$, the input to the model is the total population of The Netherlands $T[k]$ and the number of abolished municipalities $N_a[k]$ in each year k in the period $(1851-2019)$. The DMN model is iteratively applied for each year k in the following order

- Based on the assumption in (6.21), update the population vector as $p[k+1] = \frac{T[k+1]}{T[k]}\cdot p[k]$.
- Apply the migration model, defined in (6.24).
- Compute the Likelihood Abolishment index $x[k]$ per municipality, as in (6.25). Determine $N_a[k]$ municipalities with the highest ranking in the sorted index vector $x[k]$. Assign the population and area of each abolished municipality to an adjacent municipality closest in the ranking in the sorted vector $x[k]$.
- Update the $N[k+1]\times N[k+1]$ adjacency matrix $A[k+1]$ of the DMN, the $N[k+1]\times 1$ population vector $p[k+1]$ and the $N[k+1]\times 1$ area vector $s[k+1]$.

For the migration model in (6.24), the used values of the forward migration rate $\alpha[k]$ and the backward migration $\delta[k]$ rate per year k are provided in Figure 6.18. The migration rates are determined heuristically, motivated by observations in Section 6.3.3. In the following subsection, we analyse the DMN model prediction accuracy.

6.4.5. Prediction Accuracy of the DMN Model

The measured population distribution per municipality is compared with the predicted population distribution by the DMN model. The measured and predicted population

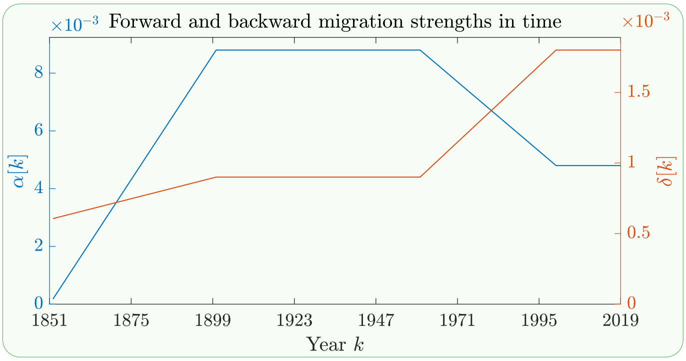


Figure 6.18: Forward migration rate $\alpha[k]$ (blue colour) and the backward migration rate $\delta[k]$ (red colour) over time k.

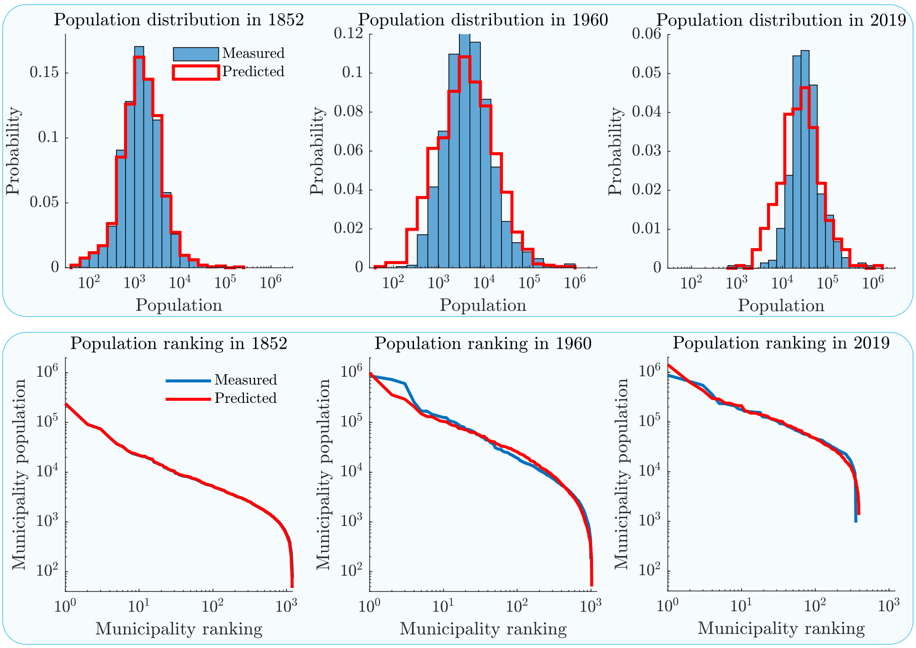


Figure 6.19: Measured versus predicted population distribution per municipality in years 1852, 1960 and 2019 (upper part). Measured versus predicted population rank-size distribution per municipality in years 1852, 1960 and 2019 (lower part).

distributions are compared for the years 1852, 1960 and 2019 in the upper part of Figure 6.19. Further, the lower part of the Figure provides the rank-size distribution of both the measured and predicted population vector.

The distribution of the predicted population vector per Dutch municipality closely follows the distribution of the measured population vector over time during the entire research period. With the proposed decoupling of the population dynamics into an equal increase per municipality in (6.21) and the migration process in (6.24), we can explain how the Dutch population distribution evolved in the last 170 years.

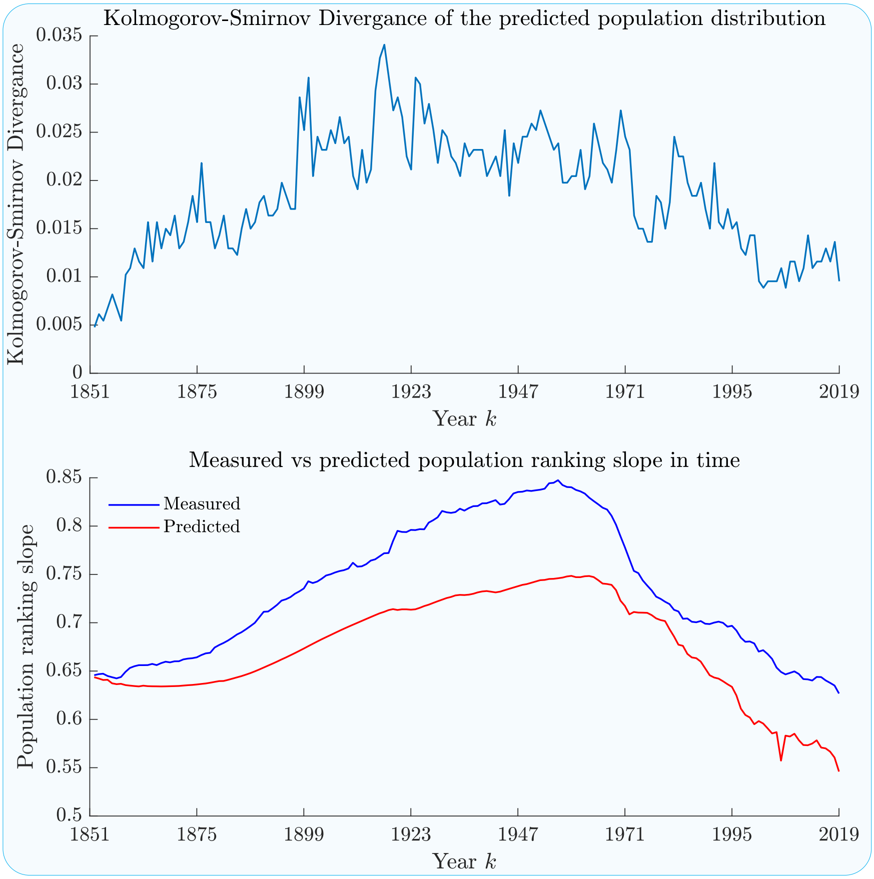


Figure 6.20: Kolmogorov-Smirnov divergence between the measured and predicted population distribution in the period (1852 – 2019) (upper figure). Predicted versus measured population rank-size slope in the period (1852 – 2019) (lower figure).

To quantify the precision of the predicted population distribution over time, we compute the Kolmogorov-Smirnov divergence between the predicted and measured population distribution and provide the results in the upper part of Figure 6.20. The divergence value remains relatively low during the entire researched period, indicating that the adopted values for the forward migration rate $\alpha[k]$ and the backward migration rate $\delta[k]$ indeed reveal the migration flows in The Netherlands.

The predicted versus the measured population rank-size distribution slope $\beta[k]$ is provided in the lower part of Figure 6.20. The rank-size slope of the predicted population vector depends solely on the migration process in (6.24) and, thus, on the forward α and the backward δ migration rate, provided in Figure 6.18. Opposite trends in migration rates $\alpha[k]$ and $\delta[k]$ until and after 1960 marked a dynamic transition in the rank-size slope $\beta[k]$ of the population vector.

In Figure 6.21, we compare the distribution of all abolished municipalities in the period (1851 – 2019) per Dutch province with predicted mergers by the DMN model. The proposed DMN model achieves a fantastic precision of 91,7%. The DMN topology in 1851 initialises the DMN model. However, the road bridges and dikes built after 1851 connected many isolated components of the Dutch Municipality Network to the mainland, as discussed in Appendix E.4. Consequently, the number of isolated components in the DMN monotonically decreased over time, which is not taken into account in the DMN model. For example, the entire province of Zeeland remains disconnected from the mainland in the DMN model, which is not the case in reality. We argue that the model precision could be even higher if the topology changes of the DMN after 1851 were taken into account in the proposed DMN model.

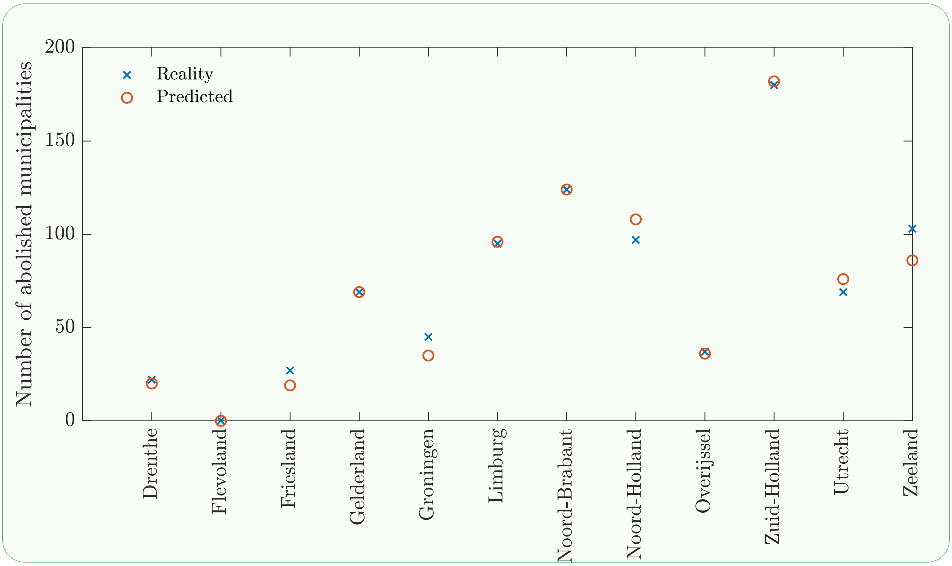


Figure 6.21: Predicted versus the measured number of abolished municipalities per Dutch province in the period 1851 – 2019.

6.5. CONCLUSION

Linking the data sets collected by Statistics Netherlands and the International Institute of Social History enabled us to investigate the Dutch municipality merging process and the survivability of municipalities over the period 1830-2019. In a geographic sense, a municipality can vary from a densely populated urbanized (city) area to a set of sparsely populated localities[29] in a rural area. In a governmental sense, a municipality is a highly autonomous administrative unit serving its population at the local government level and interacting with the overarching levels of the provincial and national government. All 1467 Dutch domestic municipalities that have ever existed between 1830 and 2019 are captured in a research construct referred to as the Dutch Municipality Network (DMN).

Compared to the first 160 years of the researched time series, the last three decades reveal considerable fluctuations, such as in the observed values of the slope of the population increase per municipality (see section 6.4.1). Can these recent fluctuations be solely attributed to the accelerated decrease in the number of Dutch municipalities due to the intensified merging process? What do these fluctuations mean for a country as a whole?

Over the entire research period, we have observed the amazing property that *the logarithm* of population size and municipality area features an almost linear difference equation over the years. This underlying "log-linearity" in the evolution process resulted in a predictive accuracy of 91.7% at the province level (see section 6.4) in spite of all municipality mergers and dynamic population fluctuations that took place in reality. The remarkable "log-linearity" in population size and municipality area asks for a scientific explanation. The collected data researched here are macroscopic statistical observations derived from many individual movements. Just as for interacting particle systems in physics, we think that the macroscopic observations can be explained if the rules or laws on the microscopic level, i.e. on individual human level, are known. Unfortunately, human behaviour is far more complicated than the already exceedingly complex interacting physical systems in nature, because the latter obey physical laws, while the governing laws – expressed in differential equations to allow computations – of human behaviour are yet unknown. Many complex networks (flock of birds, synchronization of fire-flies or heart muscles, interacting particle systems at atomic or molecule level, epidemics, etc.) possess reasonably simple local rules at the nodal or individual level, but the multitude of the interacting local rules gives rise to a surprisingly complex emerging behaviour, often characterized by phase transitions. Ab-initio calculation of a possible phase transition for the 'packing' of humans is therefore out of reach.

Nevertheless, we hypothesize on the observations of "log-linearity". The scale free power law behaviour, i.e. the linear relationship of population and area of municipalities as a function of rank-order in double logarithmic plots (figure 6.14), could be a manifestation that the system of human habitation is in a self-organised critical state, typically associated with phase transitions. The population in the Netherlands over the past 200 years has remained at or near a certain phase transition. The distribution function of the population over municipalities has similar long power-law tails (figure 6.10), although, of course, there is a cut-off at population sizes (which are too small to justify the existence of a municipality). One might speculate that a 'fully solid' phase for human habitation occurs when the entire population of the Netherlands lives in households occupying a minimum acceptable amount of space. Trying to squeeze more humans will cause repulsive forces. On the other hand, a 'fluid/gas phase' would be a fully dispersed population, in which inhabitants have a comfortable individual living space. However, the benefits

[29] In 2019, 2168 population localities were grouped into 355 Dutch domestic municipalities.

of being close to other people for social interaction as well as the mutual exchange of goods or services constitutes an attraction for humans, that pulls them towards the 'fully solid' phase. Therefore, the sketched hypothetical human packing process shares some qualitative properties with known dynamics in complex networks in which phase transitions occur, making it worthwhile to explore the possibility of phase transitions in the habitation of people.

7

Networked Systems with Linear Dynamics

Science is a way of thinking
much more than it is a body of knowledge.

Carl Sagan

This chapter studies the dynamics of complex networks with a time-invariant underlying topology, composed of nodes with linear internal dynamics and linear dynamic interactions between them. While graph theory defines the underlying topology of a network, a linear time-invariant state-space model analytically describes the internal dynamics of each node in the network. By combining linear systems theory and graph theory, we provide an explicit analytical solution for the network dynamics in discrete-time, continuous-time and the Laplace domain. The proposed theoretical framework is scalable and allows hierarchical structuring of complex networks with linear processes while preserving the information about network, which makes the approach reversible and applicable to large scale networks.

This chapter is based on [49].

7.1. Introduction

Networks are everywhere. Real-world examples of networks are electric power networks, transportation networks, water networks, economic networks, the Internet, the World Wide Web, social networks and biological networks. Dorfler *et al.* [125, 126] applied network concepts on electrical networks. Van Mieghem *et al.* [23] examined resistive networks and provided best spreaders, based on a weighted Laplacian matrix, while Cetinay *et al.* [127] analysed the vulnerability of power networks under targeted attacks. Guimera *et al.* [128] found that the world-wide air transportation network is a small-world network, while Dunne *et al.* [129] discovered that food-web networks are generally not small-world networks. Newman *et al.* [43] used the theory of random graphs with arbitrary degree distribution to model the behavior of a collaboration network of scientists. Topology of the Internet and the World Wide Web was discovered by Faloutsos *et al.* [130]. In the past two decades, the network topology has been deeply studied, for which we refer to the books by Newman [31]; Boccaletti *et al.* [131] and by Van Mieghem [51].

Each network is defined by its underlying topology and the dynamics that take place on the network. The interplay between the network topology and dynamics has been an active field of scientific research in the past two decades [45]. However, Newman [31] observed that the progress in analyzing the structural properties of the network has been faster than the one related to the dynamics taking place on the network. Barzel, Harush *et al.* [32–34] showed that, while many real networks tend to have similar (universal) structural properties, there exist classes of dynamical processes that exhibit fundamentally different flow patterns. The network dynamics depend on both the network topology and the type of dynamic interactions between the nodes.

7

During last two decades, dynamical processes on complex networks such as phase transitions [35], percolation [36], synchronization [37], diffusion [38], epidemic spreading [39–42], collective behavior [43] and traffic [44] have been intensively researched [45]. The dynamics of the real-world networks are non-linear and their underlying topology is time-varying [46]. However, complex networks with linear dynamics have been intensively researched recently [47, 48], which can be motivated in several ways. Firstly, non-linear dynamics on the networks can be approximated [132] or bounded [39] by the linear dynamics, in most cases. Secondly, the notion of controlling complex networks has become an important research question [48, 133]. In system theory, non-linear system control is a difficult problem, which has been developed on previously well-established linear system control theory [134]. An analogous order of research development is noticeable in network control theory. Several names for the complex networks with linear dynamics have been used in literature, such as networks of agents (dynamical systems) [135], networked multi-input-multi-output (MIMO) systems [136] and complex networked dynamical systems [47]. Mentioned approaches define models of the network with linear processes from the system/control theory point of view.

Here, a general framework for a complex network with linear processes is proposed, where nodes perform heterogeneous, higher-order linear dynamics, with multi-dimensional input and output vectors. The framework is based upon two assumptions: (1) The internal dynamics of the nodes, as well their interactions, are linear and (2) The underlying topology of the network is time-invariant. The framework allows each dynamic interaction between the nodes to be defined locally and independently and results in the most general description of a network of linear processes available in the literature, to our best knowledge. We provide the analytic solution (both in discrete-time, continuous-time and the Laplace domain) for the network dynamics as a whole, in terms of the internal dynamics of the nodes and the underlying graph that couples these linear processes. Thus, we preserve the network perspective. A major novelty is the hierarchical structuring of linear dynamics, in which the lowest level in the hierarchy describes individual linearly interacting processes. After a certain clustering, subnetting or grouping of linear processes (i.e. nodes on the lowest hierarchical level), these clusters can be aggregated on the next higher level of the hierarchy again as a linear process, though with a different linear dynamics. The key property of such nodal aggregation is that no information by condensation is lost! In other words, the aggregated node precisely shows the same linear dynamics as the lower level group of individual nodes. Thus, the linearity preserves information, but allows to shield the lower level interconnection details and enables very large networks to be condensed into a smaller network of interacting aggregated nodes that preserves exactly the linear dynamics! In fact, a network with linear processes of any size can be iteratively condensed into a set of hierarchical layers, in which each layer still presents a desired, aggregated network structure. An example is traffic flows (steered by a linear process) in a small neighbourhood, condensed into a city, while cities can be condensed to countries etc. Another example are different measurement techniques of a same phenomenon, where each technique has its own granularity. As long as those techniques are linear, finer-detail measurement can be aggregated with coarser ones by choosing the proper hierarchical layer that combines them. Although the spread of Corona has not a linear dynamics (but can be linearized [137]), mobile individual traces can be combined with aggregated flows measured by sensor, telecom base-stations, WIFI hotspot and so on.

7

The present work does not directly contribute to the control theory. However, the proposed model preserves the network perspective in developing the governing equations. The generality of the proposed model (i.e. the type of one-node dynamics and the interactions between nodes), the reversible scalability of the hierarchical structuring as well the network perspective based governing equations are novelties of this chapter. Another important application of the proposed model for networks with linear processes is identification. Suppose that input and output sequences can be measured at certain places in the network during a long enough time period. Linear system identification then allows to determine the exact governing equations (see [138]). Our general framework for linear networked processes hierarchically groups the subnetworks between measurement nodes and the aggregated linear dynamics of these subnetworks can be identified.

The chapter is structured as follows. Section 2 introduces basic terminology and notations, while the network dynamics and hierarchical structuring are analysed in Section 3. The concept of Extended graph is introduced in Section 4, while the analytical solution for the network dynamics in the discrete-time domain is provided in Section 5. Finally, we conclude in Section 6.

7.2. BASIC NOTATIONS

Complex networks have two general features: a graph and a service or function, specified by dynamic processes [139].

7.2.1. NETWORK TOPOLOGY

The underlying structure (topology) of the network is assumed to be time-invariant and is represented by a graph $G(\mathscr{N},\mathscr{L})$. The graph G is defined by a set $\mathscr{N}$ of $N = |\mathscr{N}|$ nodes, representing N systems[1], and by a set $\mathscr{L}$ of L links, that interconnect the systems. The link existence of the graph G is specified by the $N \times N$ adjacency matrix W, where $w_{ij} = 1$ means that there exists a link between node i and node j, otherwise $w_{ij} = 0$. The graph G is assumed to be directed, which implies that the adjacency matrix W is not symmetric in general, i.e. $W \neq W^T$.

A node i of the graph G can also be connected to external nodes. We distinguish two types of external nodes: *input* and *output* nodes. The input nodes provide links to the nodes of the graph G and have zero in-degree, while the output nodes receive links from the nodes of the graph G and have zero out-degree. In contrast to external nodes, we call the nodes and the links of the graph G *internal nodes* and *internal links*, respectively.

There are r input nodes, defined by the set $\mathscr{M}$. The input nodes connect to the internal nodes via *input links*, specified by the $r \times N$ matrix Φ:

$$\Phi = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1N} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \phi_{r1} & \phi_{r2} & \cdots & \phi_{rN} \end{bmatrix} \tag{7.1}$$

where $\phi_{ij} = 1$ defines the existence of an input link between the i-th input and j-th internal node, otherwise $\phi_{ij} = 0$.

There are q output nodes, defined by the set $\mathscr{P}$. We refer the links connecting the internal and output nodes *output links*. The existence of the output links is defined by the $N \times q$ matrix Ψ:

$$\Psi = \begin{bmatrix} \psi_{11} & \psi_{12} & \cdots & \psi_{1q} \\ \psi_{21} & \psi_{22} & \cdots & \psi_{2q} \\ \vdots & \vdots & \vdots & \vdots \\ \psi_{N1} & \psi_{N2} & \cdots & \psi_{Nq} \end{bmatrix} \tag{7.2}$$

where element ψ_{ij} indicates whether the i-th internal node provides an output link to the j-th output node ($\psi_{ij} = 1$), or not ($\psi_{ij} = 0$).

Finally, each input node can be directly connected to an output node as well. We refer to such a link as *external link* and define their existence with the $r \times q$ matrix Z:

$$Z = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1q} \\ z_{21} & z_{22} & \cdots & z_{2q} \\ \vdots & \vdots & \vdots & \vdots \\ z_{r1} & z_{r2} & \cdots & z_{rq} \end{bmatrix} \tag{7.3}$$

[1] In this work, the words node and system have been used interchangeably

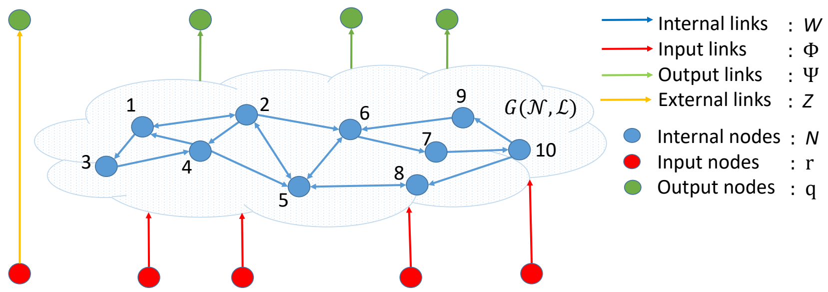


Figure 7.1: Different types of nodes and links, in case of a network of 10 nodes

where element z_{ij} defines whether there is an external link between the input node i and the output node j ($z_{ij} = 1$) or not ($z_{ij} = 0$).

The in-degree of the i-th output node is $(u^T\Psi)_i + (u^T Z)_i$, while the j-th input node has the out-degree $(\Phi u)_j + (Zu)_j$, where u is the all-one vector. All types of nodes and links defined above are presented in Fig. 7.1 and labelled by a different colour, for a graph G of 10 nodes, with additional 5 input and 4 output nodes.

7

7.2.2. PROCESSES ON THE NETWORK

Each node in the network is a linear time-invariant (LTI) system, whose dynamics are defined by a discrete-time linear state space (DLSS) model [140]. The dynamics within the i-th node/system obey the DLSS governing equations:

$$\begin{cases} x_i[k+1] & = A_i \cdot x_i[k] + B_i \cdot u_i[k] \\ y_i[k] & = C_i \cdot x_i[k] + D_i \cdot u_i[k] \end{cases} \tag{7.4}$$

where the discrete time is modelled by k. The $n_i \times n_i$ state matrix A_i defines how the $n_i \times 1$ state vector x_i depends on its previous value, while the $n_i \times m_i$ input matrix B_i determines the relation between the state vector x_i and the previous value of the $m_i \times 1$ input vector u_i. The relation between the $p_i \times 1$ output vector y_i and the state vector x_i is defined by the $p_i \times n_i$ output matrix C_i. Finally, direct relation between the output vector y_i and the input vector u_i is defined by the $p_i \times m_i$ feedforward matrix D_i.

The interconnected DLSS dynamics are sketched in Fig. 7.2, in case of a network with three nodes. We define the $N \times 1$ vector n, containing the number of states for each node/system of the network:

$$n = \begin{bmatrix} n_1 & n_2 & \dots & n_i & \dots & n_N \end{bmatrix}^T \tag{7.5}$$

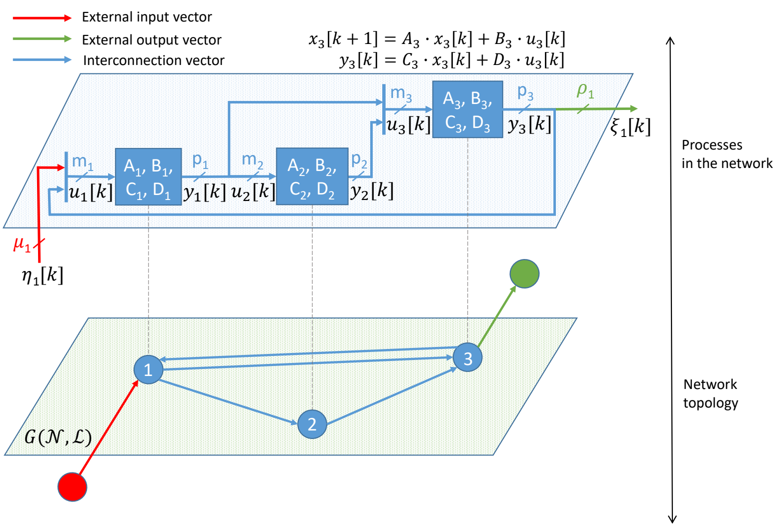


Figure 7.2: DLSS dynamics of a simple network with $N = 3$ nodes/systems

7

Similarly, we define the $N \times 1$ vector m that contains the dimension of the input vector u_i for each system ($i \in \mathcal{N}$):

$$m = \begin{bmatrix} m_1 & m_2 & \dots & m_i & \dots & m_N \end{bmatrix}^T \tag{7.6}$$

where m_i represents the dimension of the input vector u_i of the node/system i. Analogously, the $N \times 1$ vector p defines the dimension of the output vector y_i for each system in the network ($i \in \mathcal{N}$):

$$p = \begin{bmatrix} p_1 & p_2 & \dots & p_i & \dots & p_N \end{bmatrix}^T \tag{7.7}$$

where p_i represents the dimension of the output vector y_i of the i-th system.

The input vector u_i of the i-th system of the network can be composed of the output vectors from other systems (due to interconnections) and of the external input vectors. In other words, only internal and input links can be connected to an internal node.

The i-th external input vector is denoted by η_i and has dimension $\mu_i \times 1$. We define the $r \times 1$ vector μ that contains the dimension of each external input vector:

$$\mu = \begin{bmatrix} \mu_1 & \mu_2 & \dots & \mu_i & \dots & \mu_r \end{bmatrix}^T \tag{7.8}$$

In addition, we define the $M \times 1$ vector η, by concatenating r external input vectors:

$$\eta = \begin{bmatrix} \eta_1 & \eta_2 & \dots & \eta_i & \dots & \eta_r \end{bmatrix}^T \tag{7.9}$$

where $M = \sum_{j=1}^{r} \mu_j$.

An external output vector can be composed of the output vectors of the systems from the network, as well as of the external input vectors. The i-th external output vector is denoted by ξ_i and has dimension $\rho_i \times 1$. We define the $q \times 1$ vector ρ containing the dimension of each external output vector ξ_i ($i \in \mathscr{P}$):

$$\rho = \begin{bmatrix} \rho_1 & \rho_2 & \dots & \rho_i & \dots & \rho_q \end{bmatrix}^T \tag{7.10}$$

In addition, we define the $P \times 1$ vector ξ, composed by concatenating q external output vectors:

$$\xi = \begin{bmatrix} \xi_1 & \xi_2 & \dots & \xi_i & \dots & \xi_q \end{bmatrix}^T \tag{7.11}$$

where $P = \sum_{j=1}^{q} \rho_j$. We introduce the $N \times 1$ vectors l_ϕ and l_w, as well as the $q \times 1$ vectors

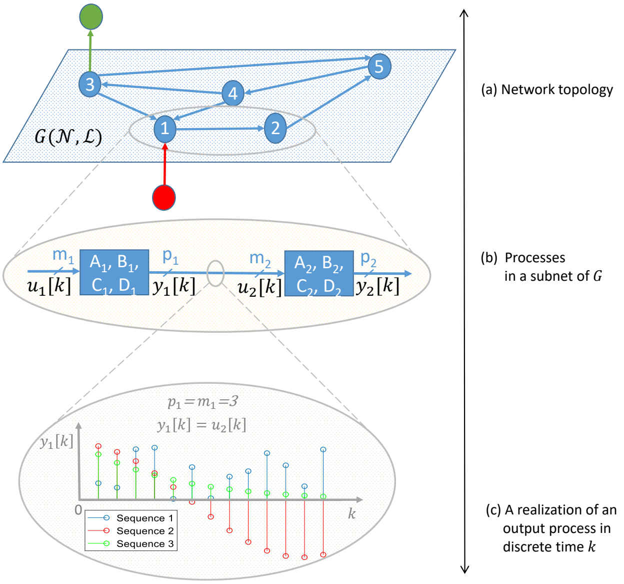


Figure 7.3: Network topology, processes and time realization of a process

l_z and l_ψ as follows:

$$\begin{aligned} l_\phi &= \Phi^T \cdot u_{r\times 1} \qquad & l_w &= W^T \cdot u_{N\times 1} \\ l_z &= Z^T \cdot u_{r\times 1} \qquad & l_\psi &= \Psi^T \cdot u_{N\times 1} \end{aligned} \tag{7.12}$$

where $(l_w)_i$ defines the number of internal links connected to the internal node i, while the number of input links that internal node i receives is defined by $(l_\psi)_i$. Additionally, the output node i receives $(l_\psi)_i$ links from the internal nodes, as well as $(l_z)_i$ external links. Total number of the internal, input, output and external links is defined as follows:

$$\begin{aligned} L_w &= (l_w)^T \cdot u_{N\times 1} \qquad & L_\phi &= (l_\phi)^T \cdot u_{N\times 1} \\ L_\psi &= (l_\psi)^T \cdot u_{q\times 1} \qquad & L_z &= (l_z)^T \cdot u_{q\times 1} \end{aligned} \tag{7.13}$$

respectively.

A graph G of 5 nodes, together with the one input and one output node is presented in Fig. 7.3(a). The processes within the first and second node of G are sketched in Fig. 7.3(b). Finally, a realization of the output vector of the first node is presented in Fig. 7.3(c).

The dimension m_i of the input vector u_i of each node in G ($i \in \mathcal{N}$) must obey:

$$m = W^T \cdot p + \Phi^T \cdot \mu \tag{7.14}$$

Analogously, the dimension ρ_i of each external output vector ξ_i ($i \in \mathcal{P}$) must obey:

$$\rho = \Psi^T \cdot p + Z^T \cdot \mu \tag{7.15}$$

Relations (7.14) and (7.15) can be written together in a matrix form:[2]

$$\begin{bmatrix} m_{N\times 1} \\ \rho_{q\times 1} \end{bmatrix} = \begin{bmatrix} W^T_{N\times N} & \Phi^T_{N\times r} \\ \Psi^T_{q\times N} & Z^T_{q\times r} \end{bmatrix} \cdot \begin{bmatrix} p_{N\times 1} \\ \mu_{r\times 1} \end{bmatrix} \tag{7.16}$$

7.3. NETWORK DYNAMICS

A complex network is composed of N nodes/systems, with internal DLSS dynamics defined by (7.4). We now would like to find the dynamics between the aggregated external output vector ξ defined in (7.11) and the aggregated external input vector η defined in (7.9), by following DLSS governing equations:

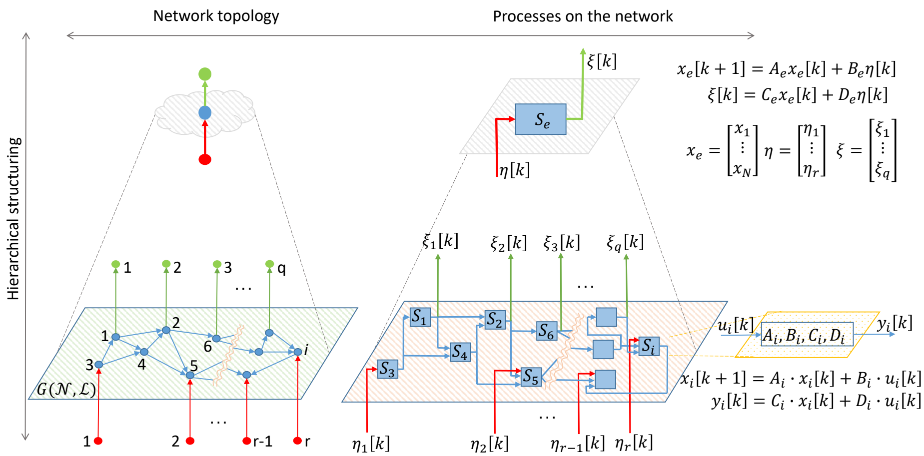


Figure 7.4: Underlying topology vs linear processes on the complex network

[2]Determining the rank of the matrix in (7.16) is a problem similar to the problem of determining the rank of the adjacency matrix of a directed graph, see e.g. [141].

$$\begin{cases} x_e[k+1] & = A_e \cdot x_e[k] + B_e \cdot \eta[k] \\ \xi[k] & = C_e \cdot x_e[k] + D_e \cdot \eta[k] \end{cases} \tag{7.17}$$

where the $\sum_{j=1}^{N} n_j \times 1$ vector x_e contains states of each system in the network:

$$x_e[k] = \begin{bmatrix} x_1[k] \\ x_2[k] \\ \vdots \\ x_N[k] \end{bmatrix} \tag{7.18}$$

The matrices A_e, B_e, C_e and D_e will be determined in terms of network topology and the dynamics of individual nodes/systems.

7.3.1. HIERARCHICAL STRUCTURING OF COMPLEX NETWORKS

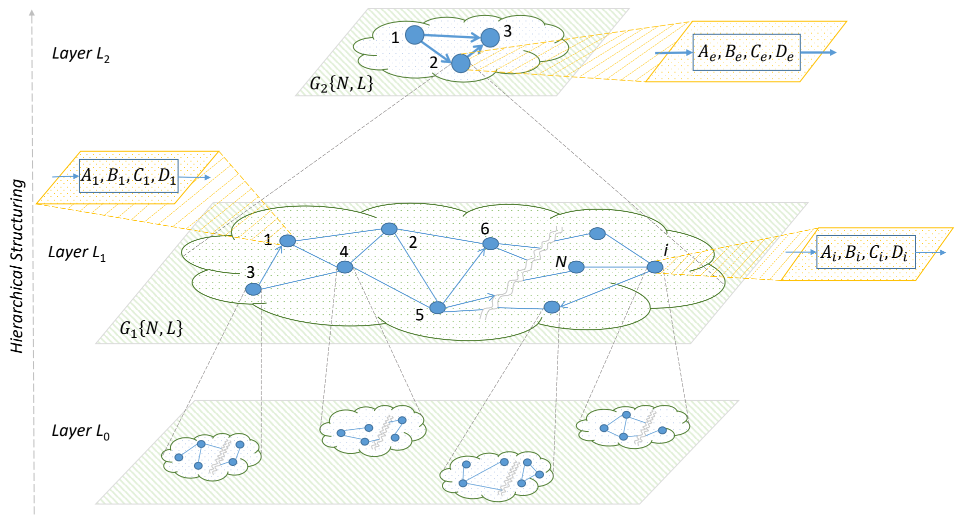


Figure 7.5: Hierarchical Structuring of Complex Networks with Linear Processes

The underlying topology of the network, together with the input and output links is sketched in the left lower part of Fig 7.4, while the processes within each node/system of the network are presented in the right lower part. By determining the DLSS process in (7.17), we determine the network dynamics. Thus, we can abstract the network dynamics with a DLSS process, as provided in the right upper part of Fig 7.4. This abstraction is analogous to abstracting the network topology by a node, as shown in the left upper part of Fig 7.4.

An example of hierarchical structuring is provided in Fig 7.5. We use three layers of abstraction, namely Layer L_0, Layer L_1 and Layer L_2. A network G_1 of N interconnected nodes with internal dynamics is presented in the Layer L_1. The dynamics of the network G_1 are abstracted by the dynamics within the node 2 of the network G_2, in a higher abstraction layer L_2. There are two additional nodes in G_2 and they abstract the dynamics of another two networks from the Layer L_1. An external impact on the network dynamics from the layer L_1 represents an interconnection between the nodes/abstracted networks in G_2.

In the same time, the internal dynamics of a node from G_1 abstract the dynamics of a network from a lower abstraction layer L_0, as presented in Fig 7.5. An external impact on the dynamics of a network in abstraction layer L_0 represents an interconnection in abstraction layer L_1 and a mode of the dynamics within a node from abstraction layer L_2.

Thus, the proposed theoretical framework allows the hierarchical structuring of complex networks with linear processes. By using the same type of governing equations (DLSS governing equations) to describe both the internal dynamics within a node/system from the network and the network dynamics, we enable hierarchical structuring of complex networks.

7.4. EXTENDED GRAPH

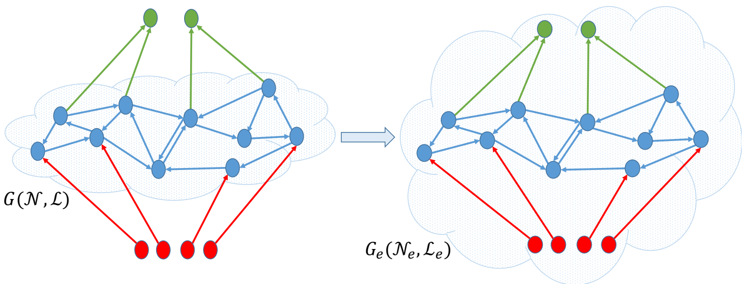


Figure 7.6: Concept of the *extended* graph G_e

The underlying topology of the network is defined by a graph G. Beside nodes of the graph G, input and output nodes are also defined, as source of input and external links and as destination of output and external links, respectively. Therefore, we introduce the *extended graph* $G_e(\mathscr{N}_e, \mathscr{L}_e)$, that is composed of $N_e = |\mathscr{N}_e|$ nodes:

$$N_e = r + N + q \quad \mathscr{N}_e = \mathscr{M} \cup \mathscr{N} \cup \mathscr{P} \tag{7.19}$$

and of L_e links:

$$L_e = L_\phi + L_w + L_\psi + L_z \tag{7.20}$$

The relation between the graph G and the extended graph G_e is presented in Fig. 7.6. The input nodes of the extended graph G_e are labelled first, before the internal nodes, while the output nodes are labelled as the last q nodes of G_e. Extended graph G_e from Fig. 7.6 with labelled nodes is presented in Fig. 7.7.

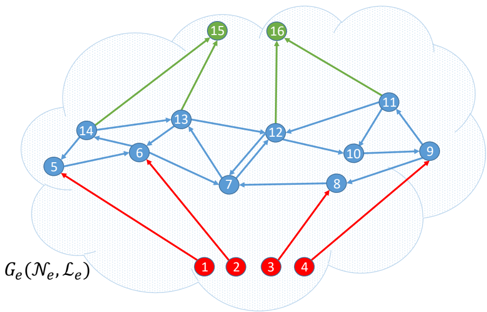


Figure 7.7: Labelled nodes of the extended graph G_e from Fig. 7.6

The adjacency matrix W_e of the extended graph G_e has a block structure:

$$W_e = \begin{bmatrix} O_{r\times r} & \Phi_{r\times N} & Z_{r\times q} \\ O_{N\times r} & W_{N\times N} & \Psi_{N\times q} \\ O_{q\times r} & O_{q\times N} & O_{q\times q} \end{bmatrix} \tag{7.21}$$

Since the input nodes have zero in-degree, the first block column of W_e is composed of zero block matrices. Similarly, since the output nodes have zero out-degree, the third block row contains zero block matrices as well.

The links whose destination is the first **internal** node of G_e are labelled first, in ascending order, relative to the source node. Next, the links connected to the second internal node are labelled. After labelling all the incoming links to the internal nodes, links whose destination is the first **output** node are labelled, in ascending order, relative to the source node. Then the incoming links of the second output node are labelled, in ascending order relative to the source node. The incoming links of the q-th output node are labelled last. The links of the extended graph G_e from Fig. 7.6 have been labelled by our convention and presented in Fig. 7.8.

We introduce the $N_e \times L_e$ incidence matrix Λ of extended graph G_e in block form:

$$\Lambda = \begin{bmatrix} (\Lambda_{11})_{r\times(L_w+L_\phi)} & (\Lambda_{12})_{r\times(L_\psi+L_z)} \\ (\Lambda_{21})_{N\times(L_w+L_\phi)} & (\Lambda_{22})_{N\times(L_\psi+L_z)} \\ (\Lambda_{31})_{q\times(L_w+L_\phi)} & (\Lambda_{32})_{q\times(L_\psi+L_z)} \end{bmatrix} \tag{7.22}$$

The first block column of Λ refers to the links whose destination is an internal node. There are $L_w + L_\phi$ such links. The second block column of Λ refers to the links whose

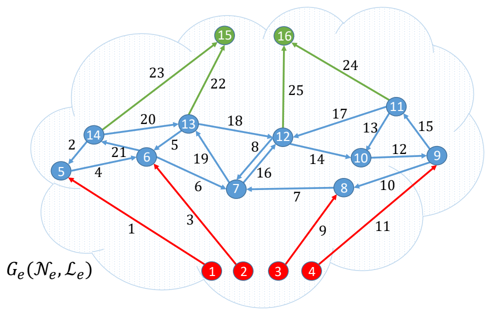


Figure 7.8: Labelled links of the extended graph G_e from Fig. 7.6

destination is an output node. There are $L_\psi + L_z$ such links. The first block row of Λ refers to the r input nodes. Further, the second block row is related to the N internal nodes, while the third block row regards the q output nodes.

We define the $L_e \times N_e$ matrix Γ as follows:

$$\Gamma = \frac{\Lambda^T + |\Lambda^T|}{2} \tag{7.23}$$

where $|\Lambda^T|$ denotes the absolute value of each element of Λ^T. Matrix Γ has a block structure:

$$\Gamma = \begin{bmatrix} (\Gamma_\phi)_{(L_w+L_\phi)\times r} & (\Gamma_w)_{(L_w+L_\phi)\times N} & O_{(L_w+L_\phi)\times q} \\ (\Gamma_z)_{(L_\psi+L_z)\times r} & (\Gamma_\psi)_{(L_\psi+L_z)\times N} & O_{(L_\psi+L_z)\times q} \end{bmatrix} \tag{7.24}$$

where each block element of Γ is of same dimensions as the according block element of transposed incidence matrix Λ^T of G_e. The negative entries of Λ^T define the destination node for each link of G_e and are not contained in Γ. Therefore, the third block column of Γ that is related to output nodes, contains zero block matrices. We observe that matrix Γ is a zero-one matrix. Each row of Γ regards certain link in G_e and contains exact one non-zero component, that refers to the source node of that link.

We further introduce the $(L_w + L_\phi) \times 1$ vectors s_ϕ and s_w, as well as the $(L_\psi + L_z) \times 1$ vectors s_z and s_ψ as follows:

$$\begin{bmatrix} (s_\phi)_{(L_w+L_\phi)\times 1} & (s_w)_{(L_w+L_\phi)\times 1} \\ (s_z)_{(L_\psi+L_z)\times 1} & (s_\psi)_{(L_\psi+L_z)\times 1} \end{bmatrix} = \Gamma \cdot \begin{bmatrix} \mu_{r\times 1} & O_{r\times 1} \\ O_{N\times 1} & p_{N\times 1} \\ O_{q\times 1} & O_{q\times 1} \end{bmatrix} \tag{7.25}$$

where $(s_w)_i$ defines the dimension of the i-th internal link, $(s_\phi)_i$ defines the dimension of the i-th input link, while $(s_\psi)_i$ and $(s_z)_i$ define the dimensions of the i-th output and i-th external link, respectively. The total number of links that are connected to the internal nodes is $L_\phi + L_w$, while $L_\psi + L_z$ is the total number of links that have the output nodes as destination. Total dimensions S_w, S_ϕ, S_ψ and S_z of all internal, input, output and external links, respectively, are defined as follows:

$$\begin{array}{ll} S_w = s_w^T \cdot u_{(L_w+L_\phi)\times 1} & S_\phi = s_\phi^T \cdot u_{(L_w+L_\phi)\times 1} \\ S_\psi = s_\psi^T \cdot u_{(L_\psi+L_z)\times 1} & S_z = s_z^T \cdot u_{(L_\psi+L_z)\times 1} \end{array} \tag{7.26}$$

Since the input and internal links are connected to internal nodes, while the output and external links have output nodes as a destination, next identities hold:

$$S_w + S_\phi = \textstyle\sum_{i=1}^N m_i \qquad S_\psi + S_z = \sum_{i=1}^q p_i \tag{7.27}$$

Additionally, we define the diagonal block matrices containing DLSS matrices of each system of the network, namely $(A_d)_{\sum_{i=1}^N n_i \times \sum_{i=1}^N n_i}$, $(B_d)_{\sum_{i=1}^N n_i \times \sum_{i=1}^N m_i}$, $(C_d)_{\sum_{i=1}^N p_i \times \sum_{i=1}^N n_i}$ and $(D_d)_{\sum_{i=1}^N p_i \times \sum_{i=1}^N m_i}$:

$$\begin{cases} A_d & = \text{diagonal}\begin{bmatrix} A_1 & A_2 & \dots & A_N \end{bmatrix} \\ B_d & = \text{diagonal}\begin{bmatrix} B_1 & B_2 & \dots & B_N \end{bmatrix} \\ C_d & = \text{diagonal}\begin{bmatrix} C_1 & C_2 & \dots & C_N \end{bmatrix} \\ D_d & = \text{diagonal}\begin{bmatrix} D_1 & D_2 & \dots & D_N \end{bmatrix} \end{cases} \tag{7.28}$$

Matrices A_d, B_d, C_d and D_d enable us to define the dynamics of each system of the network in a compact block diagonal form:

$$\begin{cases} x_e[k+1] & = A_d \cdot x_e[k] + B_d \cdot u_d[k] \\ y_d[k] & = C_d \cdot x_e[k] + D_d \cdot u_d[k] \end{cases} \tag{7.29}$$

where the $\sum_{i=1}^N m_i \times 1$ aggregated input vector u_d and the $\sum_{i=1}^N p_i \times 1$ aggregated output vector y_d are defined as follows:

$$u_d = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix} \quad y_d = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \tag{7.30}$$

Definition 20 *The aggregated input vector u_d, aggregated output vector y_d, aggregated external input vector η and the aggregated external output vector ξ are related as follows:*

$$\begin{cases} u_d[k] & = F_w \cdot y_d[k] + F_\phi \cdot \eta[k] \\ \xi[k] & = F_\psi \cdot y_d[k] + F_z \cdot \eta[k] \end{cases} \tag{7.31}$$

where the $(S_w+S_\phi)\times\sum_{i=1}^N p_i$ matrix F_w, the $(S_w+S_\phi)\times M$ matrix F_ϕ, the $(S_\psi+S_z)\times\sum_{i=1}^N p_i$ matrix F_ψ and the $(S_\psi+S_z)\times M$ matrix F_z, are composed of $(L_w+L_\phi)\times N$, $(L_w+L_\phi)\times r$,

$(L_\psi + L_z) \times N$ and $(L_\psi + L_z) \times r$ block elements, respectively, that are defined as follows:

$$\left(F_w\right)_{ij} = \begin{cases} I_{(s_w+s_\phi)_i} & if \left(\Gamma_w\right)_{ij} = 1 \\ O_{(s_w+s_\phi)_i \times p_j} & otherwise \end{cases} \quad \left(F_\phi\right)_{ij} = \begin{cases} I_{(s_w+s_\phi)_i} & if \left(\Gamma_\phi\right)_{ij} = 1 \\ O_{(s_w+s_\phi)_i \times \mu_j} & otherwise \end{cases}$$
$$\left(F_\psi\right)_{ij} = \begin{cases} I_{(s_\psi+s_z)_i} & if \left(\Gamma_\psi\right)_{ij} = 1 \\ O_{(s_\psi+s_z)_i \times p_j} & otherwise \end{cases} \quad \left(F_z\right)_{ij} = \begin{cases} I_{(s_\psi+s_z)_i} & if \left(\Gamma_z\right)_{ij} = 1 \\ O_{(s_\psi+s_z)_i \times \mu_j} & otherwise \end{cases} \tag{7.32}$$

The definition is elaborated in Appendix F.2. Matrices F_w, F_ϕ, F_ψ and F_z are defined similarly as the Kronecker products. However, each block element of these matrices is of different dimensions, which is not the case in the Kronecker product. Dimensions of the block elements vary because each vector in the network is in general of a different dimension, which is thoroughly explained in [49]. Furthermore, in Appendix F.4, we analyse homogeneous networks with identical dynamic interactions, which is a special case of the network, that allows applying the Kronecker product. Therefore, governing equations for the time dynamics of the entire network are based on the Kronecker product.

7.5. MAIN RESULTS

Theorem 21 *The matrices A_e, B_e, C_e and D_e from the DLSS governing equations in (7.17):*

$$\begin{cases} x_e[k+1] & = A_e \cdot x_e[k] + B_e \cdot \eta[k] \\ \xi[k] & = C_e \cdot x_e[k] + D_e \cdot \eta[k] \end{cases}$$

provided the matrix $(I - D_d \cdot F_w)$ is non-singular or $(D_d \cdot F_w)$ has not an eigenvalue 1, are explicitly determined as follows:

$$\begin{cases} A_e = (B_d \cdot F_w) \cdot (I - D_d \cdot F_w)^{-1} \cdot C_d + A_d \\ B_e = (B_d \cdot F_w) \cdot (I - D_d \cdot F_w)^{-1} \cdot (D_d \cdot F_\phi) + B_d \cdot F_\phi \\ C_e = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot C_d \\ D_e = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot (D_d \cdot F_\phi) + F_z \end{cases} \tag{7.33}$$

Proof. Appendix F.3

Corollary 6 *When there is no direct interaction between the input vector u_i and the output vector y_i of each system in the network (i.e. $D_i = O_{p_i \times m_i}, i \in \mathcal{N}$), the matrices A_e, B_e, C_e and D_e are explicitly determined as follows:*

$$\begin{cases} A_e & = B_d \cdot F_w \cdot C_d + A_d \\ B_e & = B_d \cdot F_\phi \\ C_e & = F_\psi \cdot C_d \\ D_e & = F_z \end{cases} \tag{7.34}$$

When the feedforward matrix D_i of each node/system of G is a non zero matrix (i.e. $D_i \neq O_{p_i \times m_i}, i \in \mathcal{N}$), the state vector x_i impacts the state vector x_j (i.e. $\left(A_e\right)_{ji} \neq O_{n_j \times n_i}$) if and only if there is a path from the node i to the node j in G (i.e. iff $\left(\sum_{k=1}^{N} W^k\right)_{ij} > 0$).

On the other side, when there is no direct relation between the input vector u_i and the output vector y_i for each node/system of the network (i.e. $D_i = O_{p_i \times m_i}, i \in \mathcal{N}$), the state vector x_i influences the state vector x_j (i.e. $\left(A_e\right)_{ji} \neq O_{n_j \times n_i}$) if and only if the node/system i and node/system j are direct neighbours (i.e. $w_{ij} = 1$). Thus, the relation (7.34) is significantly simpler than the solution of the general case (7.33). The further explanation of the matrices A_e, B_e, C_e and D_e in terms of paths in G_e is provided in [49].

The analysis of the continuous-time process on complex networks is provided in Appendix F.5. The solution for the network dynamics is provided both in the time domain and in the complex Laplace domain.

7.5.1. A NUMERICAL EXAMPLE

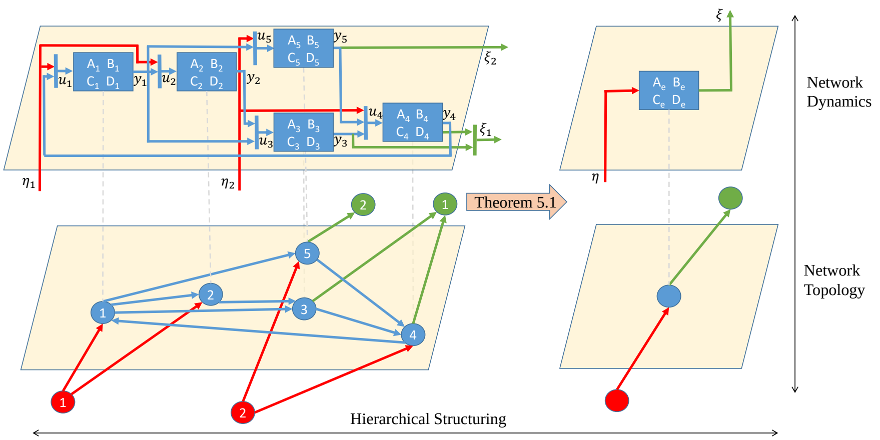


Figure 7.9: Network topology and the block diagram of the network dynamics, where $N = 5$, $r = 2$, $q = 2$

We provide a numerical example of a network model with linear processes, on which we apply the results of the chapter. Therefore, we provide a network of $N = 5$ nodes/systems, with $r = 2$ input nodes and $q = 2$ output nodes. Further, the $N \times N$ adjacency matrix W, the $r \times N$ matrix Φ, the $N \times q$ matrix Ψ and the $r \times q$ matrix Z are defined as follows:

$$W = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \Psi = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{7.35}$$
$$\Phi = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad Z = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Further, the $N \times 1$ vector p containing output dimensions of each system, the $N \times 1$ vector

n with number of states for each system and the $r \times 1$ vector μ that contains the dimension of external inputs are defined as follows:

$$p = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad n = \begin{bmatrix} 2 & 2 & 2 & 2 & 2 \end{bmatrix} \quad \mu = \begin{bmatrix} 1 & 1 \end{bmatrix} \tag{7.36}$$

while the $N \times 1$ vector m with dimensions of inputs per each node/system and the $q \times 1$ vector ρ containing dimensions of external outputs are computed using (7.16):

$$m = \begin{bmatrix} 2 & 2 & 2 & 3 & 2 \end{bmatrix} \quad \rho = \begin{bmatrix} 2 & 1 \end{bmatrix} \tag{7.37}$$

Parameters of the DLSS model of each node of the graph are defined below:

$$\begin{aligned}
A_1 &= \begin{bmatrix} 0.1227 & -0.0733 \\ 0.0733 & 0.1227 \end{bmatrix} & B_1 &= \begin{bmatrix} 0.0232 & 0.1070 \\ 0.1019 & 0.1026 \end{bmatrix} \\
C_1 &= \begin{bmatrix} 0.6547 & 0.1913 \end{bmatrix} & D_1 &= \begin{bmatrix} 0 & 0.5000 \end{bmatrix} \\
A_2 &= \begin{bmatrix} 0.3796 & -0.3920 \\ 0.3920 & 0.3796 \end{bmatrix} & B_2 &= \begin{bmatrix} 0.2371 & 0.1215 \\ 0.4789 & 0.3491 \end{bmatrix} \\
C_2 &= \begin{bmatrix} 0.0089 & 0.1603 \end{bmatrix} & D_2 &= \begin{bmatrix} 0.5000 & 0 \end{bmatrix} \\
A_3 &= \begin{bmatrix} -0.3438 & -0.2597 \\ -0.2597 & -0.7647 \end{bmatrix} & B_3 &= \begin{bmatrix} 0.0597 & 0.3568 \\ 0.4729 & 0.7413 \end{bmatrix} \\
C_3 &= \begin{bmatrix} 0.0664 & 0.2628 \end{bmatrix} & D_3 &= \begin{bmatrix} 0.5000 & 0.3394 \end{bmatrix} \\
A_4 &= \begin{bmatrix} -0.3773 & -0.0779 \\ -0.0779 & -0.9613 \end{bmatrix} & B_4 &= \begin{bmatrix} 0.7038 & 0.6134 & 0.4158 \\ 0.2989 & 0.1345 & 0.5020 \end{bmatrix} \\
C_4 &= \begin{bmatrix} 0.4997 & 0.2145 \end{bmatrix} & D_4 &= \begin{bmatrix} 0 & 0.5000 & 0 \end{bmatrix} \\
A_5 &= \begin{bmatrix} 0.5796 & -0.0619 \\ -0.0619 & 0.7033 \end{bmatrix} & B_5 &= \begin{bmatrix} 0.1072 & 0.6859 \\ 0.4328 & 0.1584 \end{bmatrix} \\
C_5 &= \begin{bmatrix} 0.1089 & 0.0430 \end{bmatrix} & D_5 &= \begin{bmatrix} 0.5000 & 0 \end{bmatrix}
\end{aligned} \tag{7.38}$$

Network topology, with input and output nodes, is presented in the lower-left part of Fig. 7.9, while the network dynamics in from of the interconnected block diagrams are presented in the upper-left part of the Figure. By applying Theorem 21 we provide the dynamics of the entire network in the form of a DLSS system, as presented in the upper-right part of the Figure. Finally, the Theorem 21 allows representing entire network topology as a node, on a higher hierarchy level.

$$\begin{aligned}
A_e &= \begin{bmatrix}
0.1402 & -0.0682 & 0.0002 & 0.0029 & 0.0040 & 0.0158 & 0.0601 & 0.0258 & 0 & 0 \\
0.0901 & 0.1276 & 0.0002 & 0.0028 & 0.0038 & 0.0152 & 0.0577 & 0.0248 & 0 & 0 \\
0.0895 & 0.0261 & 0.3797 & -0.3903 & 0.0020 & 0.0080 & 0.0341 & 0.0147 & 0 & 0 \\
0.2571 & 0.0751 & 0.3922 & 0.3843 & 0.0058 & 0.0229 & 0.0981 & 0.0421 & 0 & 0 \\
0.0440 & 0.0128 & 0.0032 & 0.0580 & -0.3428 & -0.2558 & 0.0168 & 0.0072 & 0 & 0 \\
0.3483 & 0.1017 & 0.0069 & 0.1253 & -0.2518 & -0.7336 & 0.1329 & 0.0570 & 0 & 0 \\
0.2259 & 0.0660 & 0.0021 & 0.0375 & 0.0458 & 0.1813 & -0.3006 & -0.0451 & 0.0453 & 0.0179 \\
0.0495 & 0.0145 & 0.0005 & 0.0082 & 0.0100 & 0.0398 & -0.0611 & -0.9541 & 0.0547 & 0.0216 \\
0.5052 & 0.1476 & 0.0005 & 0.0093 & 0.0114 & 0.0451 & 0.1928 & 0.0827 & 0.5796 & -0.0619 \\
0.1167 & 0.0341 & 0.0001 & 0.0022 & 0.0026 & 0.0104 & 0.0445 & 0.0191 & -0.0619 & 0.7033
\end{bmatrix} &
B_e &= \begin{bmatrix}
0.0323 & 0 \\
0.1106 & 0 \\
0.2423 & 0 \\
0.4937 & 0 \\
0.1809 & 0 \\
0.3907 & 0 \\
0.1171 & 0.9117 \\
0.0257 & 0.5499 \\
0.0291 & 0.1072 \\
0.0067 & 0.4328
\end{bmatrix} \\
C_e &= \begin{bmatrix}
0.3683 & 0.1076 & 0.0034 & 0.0612 & 0.0747 & 0.2956 & 0.1406 & 0.0603 & 0 & 0 \\
0.1841 & 0.0538 & 0.0017 & 0.0306 & 0.0374 & 0.1478 & 0.5622 & 0.2413 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1089 & 0.0430
\end{bmatrix} &
D_e &= \begin{bmatrix}
0.1909 & 0 \\
0.0954 & 0 \\
0 & 0.5000
\end{bmatrix}
\end{aligned} \tag{7.39}$$

7.6. CONCLUSION

In this chapter, we propose a general theoretical framework for modeling complex networks with time-invariant topology, composed of nodes with linear internal dynamics and with linear interactions between them.

Nodes perform heterogeneous higher-order internal dynamics, with multi-dimensional input and output vectors. The proposed framework allows to independently define each dynamic interaction between the nodes. Proper notations have been introduced for network topology and the internal dynamics of nodes. The external processes that influence the network dynamics are included in the proposed framework. The analytic solution for the network dynamics is provided in the discrete-time domain, continuous-time domain and the Laplace domain.

The assumption about linear processes on networks allows scalability of the proposed model to large-scale networks and preserves network information. Finally, the reversible hierarchical structuring of complex networks with linear processes is introduced.

8

CONCLUSION

This thesis aims to enhance understanding of the impact of connections between node pairs on network properties and their functioning through the analysis of various topological and spectral properties of networks, as well as linear processes taking place on networks. The new knowledge presented in this thesis highlights the generality and significance of the network concept, as well as how complex interactions at a network level arise from simple node pair connections. Our findings indicate that linear processes on networks, which are proportional to the underlying graph, can be successfully used to deduce the topological properties of networks.

8.1. MAIN CONTRIBUTIONS

Our findings in Chapter 2 demonstrate that every undirected graph, with the exception of the empty graph, can be retrieved from the orthogonal eigenvectors of its corresponding adjacency matrix, though not necessarily in a unique manner. This non-uniqueness arises from the presence of co-eigenvector graphs that we have identified. These graphs possess the same set of eigenvectors, yet they have different eigenvalues. Moreover, our findings reveal that regular graphs are instances of co-eigenvector graphs, and the first non-regular pair of co-eigenvector graphs arises for $N = 6$.

As previously established, the adjacency matrix raised to a certain power contains the number of walks between node pairs of that length. Encouraged by this straightforward solution, we endeavoured to find an equivalent solution for the number of paths between node pairs, where a path entails a walk with no repeated nodes. In Chapter 3, our research provides three matrix-based analytical solutions for the number of paths of a certain length between node pairs. These solutions utilise different types of walks and offer varying degrees of computational efficiency depending on the sparsity of the graph. However, unlike the number of walks, a simple analogue to the solution for the number of paths does not appear to exist.

Effective resistance measures the dissipation of energy transmitted between two nodes over the network. Originating from electrical system theory, this graph metric provides a comprehensive view of the network from the perspective of the two nodes. In Chapter 4, we utilise the information conveyed by effective resistance to propose an iterative algorithm that solves the inverse all shortest path problem while adhering to a fixed link budget. Our approach commences with the complete graph and iteratively eliminates links until the upper bound on the shortest paths is exceeded. Our algorithm demonstrates superior performance when the given shortest path weight bounds are calculated for a sparse graph. Furthermore, we propose iterative algorithms for deterministic graph sparsification with the goal of minimising or maximising the effective graph resistance or minimising deviations in the eigenvalues of the corresponding Laplacian Q.

The second part of this thesis examines linear processes on complex networks at different aggregation levels. In Chapter 5, we demonstrate that a simple node embedding on a one-dimensional line can accurately identify clusters, surpassing the performance of the most effective modularity-based and spectral clustering algorithms in the literature, such as Louvain, Leiden, and Newman, while having comparable computational complexity. Our proposed node embedding is generated through a linear process of attraction and repulsion between adjacent nodes, based on the similarity of their neighbourhoods. Nodes converge to the eigenvector corresponding to the second-largest eigenvalue of the governing matrix of our linear clustering process before reaching a trivial steady state. We estimate the number of clusters and the cluster membership of each node by optimising modularity using double recursion based on this eigenvector.

Chapter 6 considers the time dynamics of a country as a network of interconnected municipalities. Using data sets containing population and area size information for Dutch municipalities from the period spanning 1830-2019, we observe that the logarithm of both area and population adheres to a normal and logistic distribution throughout the entire period studied. Furthermore, in the tails, the population distribution conforms to the power law, a phenomenon previously noted in the literature for the largest cities of a country. The changes in area distribution are mainly attributed to the merging process, whereby neighbouring municipalities are merged. Meanwhile, population distribution over time is also influenced by the processes of population increase and migration across the country. We investigate each of these processes using empirical measurements and propose the Dutch Municipality Network model, which involves three iterative steps: modelling population increase, population migration, and merging. Remarkably, the proposed model achieves high accuracy in predicting municipality mergers on a province level over a span of two centuries.

Chapter 7 examines the scenario where each node in a network exhibits internal linear dynamics, which is characterised by a state space model of a particular order, and the dynamic interactions between nodes are linear. In this context, we propose an analytic solution for the governing equation at the network level by merging the linear systems associated with individual nodes according to the underlying topology. We demonstrate that merging interconnected linear systems into a set of master governing equations is feasible without losing any information regarding the individual dynamic processes of each node. Thus, the hierarchical structuring of a network is achievable if each node exhibits linear dynamics.

8.2. DIRECTIONS FOR FUTURE WORK

In Chapter 2, we uncover the existence of co-eigenvector graphs, which are graphs that possess the same set of eigenvectors while having different eigenvalues. Using the algorithm outlined in Figure 2.6, we discover the first pairs of co-eigenvector graphs for $N = 6$ among non-regular graphs. However, it remains unclear why co-eigenvector graphs do not appear for $N < 5$ and how the number of pairs of co-eigenvector graphs scales with the increasing size of the network, N.

The linear clustering process on networks, proposed in Chapter 5, involves clustering based on the eigenvector y_2, which corresponds to the second largest eigenvalue of the $N \times N$ matrix $W - \text{diag}(W \cdot u)$, as motivated by (5.14). To enhance the clustering performance of our linear clustering process, we adopt an iterative approach that involves computing the $N \times 1$ eigenvector y_2, estimating the partition by optimising the modularity m (Section 5.4.2), classifying the links, and scaling down the weights of inter-community links (Section 5.5). The efficacy of our linear clustering process heavily relies on the scaling step, making an improved scaling approach a promising avenue for further research.

In Chapter 7, we present the solution for the dynamics of the entire network under the assumption that each node exhibits internal linear dynamics and the interactions between nodes are also linear. This result motivates the opposite approach: identifying the dynamics of each node separately from the given input-output measurements. Given a known underlying topology and estimated dynamics for each node, we employ Theorem 21 to obtain the estimated dynamics of the entire network. One possible real-world scenario where Theorem 21 can be applied is highway road networks. In such networks, each road segment can be abstracted as a node, and speed and the number of vehicles per unit of time can be used as corresponding measurements.

ACKNOWLEDGEMENTS

Undertaking the research and writing of this thesis during my PhD studies has been an exceptional and transformative journey, one that would not have been possible without the support of many individuals. First and foremost, I express my deep gratitude to my supervisor, Professor Piet Van Mieghem, who provided me with the opportunity to conduct research at the Technical University of Delft and nurtured my growth as an independent researcher under his guidance. I am grateful for his continuous support, honest and direct feedback, and for treating me as his 'research child', a role in which he invested significant time and effort. Prof. Van Mieghem granted me the freedom to explore research topics that fascinated me while recognising the relevance and further directing my research ideas.

I am grateful to Dr Edgar van Boven for initiating our joint research project during the first year of my PhD studies. This project allowed us to collaborate with experts Hans van Hooff, Gert Buiten, Professor Frank Pijpers, and Dr Trivik Verma and supervise Ioannis Manolopoulos in his MSc research. Our joint efforts resulted in a publication of which we are proud. Edgar has provided me with substantial support throughout my doctoral journey, and I am grateful for his positive feedback and all the help. I am honoured to have been a member of the NAS research group, which provided a positive and supportive atmosphere during meetings. I would like to express my gratitude to the senior staff, particularly Professor Rob Kooij, Dr Eric Smeitink, Dr Remco Litjens, and Dr Maksim Kitsak, for their encouragement and positive attitude during NAS meetings. Guiding Songlei Fang and Beichen Wang in their MSc research was an enjoyable experience for me. I would like to extend my gratitude to the committee members who evaluated my PhD thesis for their valuable time, effort, and feedback. I also want to thank our secretaries, Joyce van Velzen, Laura de Groot, Trisha de Jonge, and Francis Bulters, for their administrative support throughout my PhD.

When I first joined the NAS group, I shared an office with senior PhD colleagues, which made my initial research steps less daunting and more enjoyable. I am thankful to Bastian Prasse for sharing his knowledge, ideas, and materials and for inspiring me with his impressive research skills. I enjoyed discussing research ideas with Zhidong He and appreciated his eagerness to communicate. It was also a pleasure sharing the office with Peng Sun, who was always kind and approachable. Throughout my PhD, I shared my office with Fenghua Wang, Gabriel Budel, and Elizaveta Evmenova, and I cherish the interesting discussions we had, as well as the little moments we shared, such as manually preparing coffee. Gabriel deserves special recognition for his kindness, willingness to help, and dedication to the group. I also appreciated being in the same group with Rogier Noldus, Qiang Liu, Long Ma, Massimo Achterberg, Maria Raftopoulou, Scott Dahlgren, Sergey Shvydun, Robin Persoons, Zhihao Qiu, Yingyue Ke, and Xinhan Liu.

I would like to express my heartfelt gratitude to my brothers, Radule and Ilija, and my parents, Vladimir and Ranka, for their love and emotional support throughout my

academic journey. Even though we were physically apart, talking with them over the phone on a daily basis provided me with a sense of comfort and made me feel like I was never too far away from home. I have always looked up to my parents for their sincerity and dedication, which has been a great source of motivation for me.

Throughout my PhD journey, I had the unconditional support of my girlfriend, Jelena, who stood by me from the very beginning and even lived with me for most of it. Her love and encouragement were crucial in motivating me to persevere through various research process stages and overcome the unique challenges we faced during the pandemic. I am forever grateful for her love, immense support and commitment to our relationship.

APPENDICES

A

HADAMARD PRODUCT IN GRAPH THEORY

Spectral decomposition of the adjacency matrix A of an undirected graph G with N nodes, defined in (2.1), consists of N eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$, contained in the $N \times 1$ vector λ and $\Lambda = \text{diag}(\lambda)$. The $N \times N$ eigenvector matrix X comprises N orthogonal eigenvectors $x_1, x_2, \cdots, x_N$ in the columns. The double orthogonality of the eigenvectors, derived in (2.3) and (2.4), implies that $X^T X = X X^T = I$, allowing to rewrite (2.1) as follows

$$A = \sum_{i=1}^{N} \lambda_i \cdot x_i \cdot x_i^T. \tag{A.1}$$

The orthogonality of the eigenvectors of A can be rewritten as $I = X \cdot I \cdot X^T$, leading to an alternative representation of the identity matrix I using eigenvectors of the adjacency matrix A

$$I = \sum_{i=1}^{N} x_i \cdot x_i^T. \tag{A.2}$$

Furthermore, any set of N orthogonal eigenvectors with unity eigenvalues define the identity matrix I. Because the $N \times N$ identity matrix I is composed of ones on the main diagonal and zeros elsewhere, it holds that $I = I \circ I$, leading to an alternative representation:

$$I = \sum_{i=1}^{N} \sum_{j=1}^{N} \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T.$$

The $N \times 1$ all-one vector $u = I \cdot u$, after importing the above relation, translates into

$$u = \sum_{i=1}^{N} \sum_{j=1}^{N} \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T \cdot u.$$

A

The inner product $\left(x_i \circ x_j\right)^T \cdot u = x_j^T \cdot x_i$ equals 0 if $i \neq j$, otherwise 1, transforming the relation above further

$$u = \sum_{i=1}^{N} (x_i \circ x_i). \tag{A.3}$$

The equation (A.3) can be formulated as $\Xi \cdot u = u$, which is already derived in Appendix (B.1), because the $N \times 1$ all-one vector u is an eigenvector of the $N \times N$ Hadamard product $\Xi = X \circ X$, corresponding to eigenvalue 1. Finally, the $N \times N$ all-one matrix $J = u \cdot u^T$ can be obtained from (A.3)

$$J = \sum_{i=1}^{N} \sum_{j=1}^{N} (x_i \circ x_i) \cdot \left(x_j \circ x_j\right)^T. \tag{A.4}$$

A.1. GRAPH WALKS AND PATHS

The adjacency matrix A of an undirected, unweighted graph is composed of either zeros or ones, from where it follows that $A = A \circ A$, which after importing (A.1) transforms into

$$A = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \cdot \lambda_j \cdot \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T. \tag{A.5}$$

The k-th power of the adjacency matrix A,

$$A^k = X \cdot \Lambda^k \cdot X^T = \sum_{i=1}^{N} \lambda_i^k \cdot x_i \cdot x_i^T. \tag{A.6}$$

comprises the number of length k walks between node pairs, as proved in Theorem 6, where $(A^k)_{ij}$ denotes the number of length k walks between node i and node j. Diagonal elements $(A^k)_{ii}$ define the number of closed walks of length k (see Definition 9) and compose the $N \times 1$ vector

$$\text{diag}(A^k) = \left(I \circ A^k\right) \cdot u.$$

After importing (A.2) and (A.1) into the relation above, we obtain

$$\text{diag}\left(A^k\right) = \sum_{i=1}^{N} \lambda_i^k \cdot (x_i \circ x_i). \tag{A.7}$$

Using the Hadamard product $\Xi = X \circ X$, defined in (2.5), we provide an alternative representation of (A.7)

$$\text{diag}\left(A^k\right) = \Xi \cdot \lambda^k. \tag{A.8}$$

On the other side, the Hadamard product $A^k \circ A^m$ of the k-th power A^k and the m-th power A^m of the adjacency matrix A,

$$A^k \circ A^m = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i^k \cdot \lambda_j^m \cdot \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T,$$

after right multiplying with the all-one vector u and using the orthogonality of eigenvectors, transforms into

$$\left(A^k \circ A^m\right) \cdot u = \sum_{i=1}^{N} \lambda_i^{k+m} \cdot (x_i \circ x_i)\,. \tag{A.9}$$

By comparing the right-hand side of relation (A.7) and relation (A.9), we conclude

$$\mathrm{diag}(A^k) = \left(A^{k-m} \circ A^m\right) \cdot u, \tag{A.10}$$

where $0 < m < k$. A straightforward explanation of the above relation is that the number of closed walks of length f for node i equals the sum of the number of length $k-m$ walks from node i to another node j, times the number of length m walks in the opposite direction, i.e. from node j to node i.

A.2. LAPLACIAN MATRIX Q

In this section we try to connect the eigenvectors of the $N \times N$ Laplacian matrix

$$Q = \Delta - A$$

to those of the adjacency matrix A. Similar to the spectral decomposition of the adjacency matrix A in (A.1), the Laplacian Q has the following spectral decomposition

$$Q = Y \cdot \mathrm{diag}(\mu) \cdot Y^T, \tag{A.11}$$

where the N orthogonal eigenvectors $y_1, y_2, \dots y_N$ are contained in the $N \times N$ eigenvector matrix Y, while the $N \times N$ diagonal matrix $\mathrm{diag}(\mu)$ contains eigenvalues $\mu_1 \geq \mu_2 \geq \dots \mu_N$ on its main diagonal. Due to orthogonality of eigenvectors of the Laplacian Q, relation (A.11) can be transformed as follows

$$Q = \sum_{i=1}^{N} \mu_i \cdot y_i \cdot y_i^T. \tag{A.12}$$

The degree diagonal matrix $\Delta = I \circ A^2$, after importing (A.1) and (A.2), transforms into

$$Q = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{2} \cdot \left(\lambda_i - \lambda_j\right)^2 \cdot \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T \tag{A.13}$$

and can be further transformed as follows

$$Q = 0 \cdot \sum_{i=1}^{N} (x_i \circ x_i) \cdot (x_i \circ x_i)^T + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \cdot \left(\lambda_i - \lambda_j\right)^2 \cdot \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T$$

that, after importing (A.3), translates into

$$Q = 0 \cdot u \cdot u^T + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left(\lambda_i - \lambda_j\right)^2 \cdot \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T. \tag{A.14}$$

A

Since it holds $\Delta = I \circ Q$ and $\Delta = I \circ A^2$, we obtain

$$\sum_{i=1}^{N}\sum_{j=1}^{N} \mu_i \cdot \left(y_i \circ x_j\right) \cdot \left(y_i \circ x_j\right)^T = \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i^2 \cdot \left(x_i \circ y_j\right) \cdot \left(x_i \circ y_j\right)^T. \tag{A.15}$$

Table A.1 provides and overview of derived identities using Hadamard product and spectral decomposition of graph-related matrices.

Table A.1: List of derived identities using Hadamard product and spectral decomposition of graph-related matrices.

$A = X \cdot \text{diag}(\lambda) \cdot X^T$	$A = \sum_{i=1}^{N} \lambda_i \cdot x_i \cdot x_i^T$
$A = A \circ A$	$A = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \cdot \lambda_j \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$A^k = X \cdot \text{diag}(\lambda)^k \cdot X^T$	$A = \sum_{i=1}^{N} \lambda_i^k \cdot x_i \cdot x_i^T$
$A^k \circ A^m$	$A^k \circ A^m = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i^k \cdot \lambda_j^m \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$(A^k \circ A^m) \cdot u$	$(A^k \circ A^m) \cdot u = \sum_{i=1}^{N} \lambda_i^{k+m} \cdot (x_i \circ x_i)$
$u^T \cdot (A^k \circ A^m) \cdot u$	$u^T \cdot (A^k \circ A^m) \cdot u = \sum_{i=1}^{N} \lambda_i^{k+m}$
$I = X \cdot I \cdot X^T$	$I = \sum_{i=1}^{N} x_i \cdot x_i^T$
$\text{diag}(\text{diag}(A^k)) = I \circ (A^k)$	$\text{diag}(\text{diag}(A^k)) = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i^k \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$\text{diag}(A^k) = (I \circ (A^k)) \cdot u$	$\text{diag}(A^k) = \sum_{i=1}^{N} \lambda_i^k \cdot (x_i \circ x_i)$
$\text{trace}(A^k) = u^T \cdot (I \circ (A^k)) \cdot u$	$\text{trace}(A^k) = \sum_{i=1}^{N} \lambda_i^k$
$I = I \circ I$	$I = \sum_{i=1}^{N} \sum_{j=1}^{N} (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$u = I \cdot u$	$u = \sum_{i=1}^{N} (x_i \circ x_i)$
$J = u \cdot u^T$	$J = \sum_{i=1}^{N} \sum_{j=1}^{N} (x_i \circ x_i) \cdot (x_j \circ x_j)^T$
$O = A \circ I$	$O = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$D = I \circ A^2$	$D = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i^2 \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$Q = Y \cdot \text{diag}(\mu) \cdot Y^T$	$Q = \sum_{i=1}^{N} \mu_i \cdot y_i \cdot y_i^T$
$Q = D - A$	$Q = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{2} \cdot (\lambda_i - \lambda_j)^2 \cdot (x_i \circ x_j) \cdot (x_i \circ x_j)^T$
$Q = Q \circ (A + I)$	$Q = \sum_{i=1}^{N} \sum_{j=1}^{N} \mu_i \cdot (\lambda_j + 1) \cdot (y_i \circ x_j) \cdot (y_i \circ x_j)^T$
$D = Q \circ I$	$D = \sum_{i=1}^{N} \sum_{j=1}^{N} \mu_i \cdot (y_i \circ x_j) \cdot (y_i \circ x_j)^T$
$A = (-Q) \circ A$	$A = \sum_{i=1}^{N} \sum_{j=1}^{N} (-\mu_i) \cdot \lambda_j \cdot (y_i \circ x_j) \cdot (y_i \circ x_j)^T$

B

Appendix for Chapter 2

B.1. Function of a symmetric matrix and the stochastic matrix Ξ

From the general relation for diagonalizable matrices (see e.g. [142, p. 526]),

$$f(A) = \sum_{k=1}^{N} f(\lambda_k)\, x_k x_k^T \tag{B.1}$$

valid for a function f defined on the eigenvalues $\{\lambda_k\}_{1\le k\le N}$ of the $N \times N$ symmetric matrix A, the element for node j equals

$$\left(f(A)\right)_{jj} = \sum_{k=1}^{N} f(\lambda_k)\,(x_k)_j^2 \tag{B.2}$$

Written in matrix form for all $1 \le j \le N$ results in

$$\begin{bmatrix} \left(f(A)\right)_{11} \\ \left(f(A)\right)_{22} \\ \left(f(A)\right)_{33} \\ \vdots \\ \left(f(A)\right)_{NN} \end{bmatrix} = \begin{bmatrix} (x_1)_1^2 & (x_2)_1^2 & (x_3)_1^2 & \cdots & (x_N)_1^2 \\ (x_1)_2^2 & (x_2)_2^2 & (x_3)_2^2 & \cdots & (x_N)_2^2 \\ (x_1)_3^2 & (x_2)_3^2 & (x_3)_3^2 & \cdots & (x_N)_3^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ (x_1)_N^2 & (x_2)_N^2 & (x_3)_N^2 & \cdots & (x_N)_N^2 \end{bmatrix} \begin{bmatrix} f(\lambda_1) \\ f(\lambda_2) \\ f(\lambda_3) \\ \vdots \\ f(\lambda_N) \end{bmatrix} \tag{B.3}$$

We write (B.3) in matrix form as $\psi = \Xi\chi$ with the vectors

$$\psi = \begin{bmatrix} \left(f(A)\right)_{11} \\ \left(f(A)\right)_{22} \\ \left(f(A)\right)_{33} \\ \vdots \\ \left(f(A)\right)_{NN} \end{bmatrix} \text{ and } \chi = \begin{bmatrix} f(\lambda_1) \\ f(\lambda_2) \\ f(\lambda_3) \\ \vdots \\ f(\lambda_N) \end{bmatrix}$$

B

where the $N \times N$ matrix $\Xi = X \circ X$ is defined in (2.5).

Since $\Xi u = u$ and $\Xi^T u = u$, by "double orthogonality" of (2.3) and (2.4), and since each element $0 \leq (x_k)_j^2 \leq 1$, the matrix Ξ with squared eigenvector components of a diagonalizable matrix A is doubly[1]-stochastic [51] with largest eigenvalue equal to 1. The latter property follows from the Perron-Frobenius Theorem of non-negative matrices. The product[2] of two doubly-stochastic matrices is also a doubly-stochastic matrix. The doubly-stochastic matrix Ξ also provides a vehicle to generate sharp inequalities, for which we refer to the book of Marshall *et al.* [143].

The $N \times N$ doubly-stochastic matrix Ξ in (2.5) can have a rank that is lower than N, in contrast to the $N \times N$ orthogonal eigenvector matrix X, whose rank always equals N. The fact that Ξ is not necessary of full rank, i.e. $\det(\Xi) = 0$ is possible, is exploited in the proof of Theorem 1 in Appendix B.2 for graph recovery.

B.1.1. THE FUNCTION $f(z) = z^k$

Let us denote the vector $\lambda^k = \left(\lambda_1^k, \lambda_2^k, \ldots, \lambda_N^k\right)$ so that, for the function $f(z) = z^k$ in (B.3) where k is a non-negative integer, we can write (B.3) as

$$\operatorname{diag}\left(\left(A^k\right)_{jj}\right) u = \Xi \lambda^k \tag{B.4}$$

where $u = (1, 1, \ldots, 1)$ is the all-one vector. From (B.4) and $u^T \Xi = u^T$, we find the well-known trace relation [51], namely that $u^T \operatorname{diag}\left(\left(A^k\right)_{jj}\right) u = \operatorname{trace}\left(A^k\right) = u^T \lambda^k = \sum_{j=1}^N \lambda_j^k$.

If the inverse Ξ^{-1} of Ξ exists, then it holds, for any integer k, that

$$\lambda^k = \Xi^{-1} \operatorname{diag}\left(\left(A^k\right)_{jj}\right) u \tag{B.5}$$

Thus, the eigenvalue λ_m of a symmetric matrix A to any, non-negative integer power k can be written as a linear combination of the diagonal elements of A^k,

$$\lambda_m^k = \sum_{i=1}^N \left(\Xi^{-1}\right)_{mi} \left(A^k\right)_{ii}$$

whereas $\Lambda^k = X^T A^k X$ shows that

$$\lambda_m^k = \sum_{i=1}^N \sum_{j=1}^N \left(A^k\right)_{ij} (x_m)_i (x_m)_j \tag{B.6}$$

[1] *Sinkhorn's theorem* (1964) states that any matrix with strictly positive entries can be made doubly-stochastic by pre- and post-multiplication by diagonal matrices.

[2] Indeed, let Ξ and Ψ be two $N \times N$ doubly-stochastic matrices. Then, left-multiplying both sides in $\Xi u = u$ by Ψ and using $\Psi u = u$ yields $\Psi \Xi u = u$. Similarly, left-multiplying both sides in $\Psi^T u = u$ by Ξ^T and using $\Xi^T u = u$ yields $(\Psi\Xi)^T u = u$. Finally, an element of $\Psi\Xi$ equals

$$0 \leq (\Psi\Xi)_{ij} = \sum_{k=1}^N \Psi_{ik} \Xi_{kj} \leq \min\left(\max_{1\leq k\leq N} \Psi_{ik}, \max_{1\leq k\leq N} \Xi_{kj}\right) \leq 1$$

which demonstrates the property.

We can write (B.4) for integers k ranging from $k=0$ up to $k=N-1$,

$$\begin{bmatrix} 1 & a_{11} & (A^2)_{11} & \cdots & (A^{N-1})_{11} \\ 1 & a_{22} & (A^2)_{22} & \cdots & (A^{N-1})_{22} \\ 1 & a_{33} & (A^2)_{33} & \cdots & (A^{N-1})_{33} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_{NN} & (A^2)_{NN} & \cdots & (A^{N-1})_{NN} \end{bmatrix} = \Xi. \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^{N-1} \\ 1 & \lambda_2 & \lambda_2^2 & \cdots & \lambda_2^{N-1} \\ 1 & \lambda_3 & \lambda_3^2 & \cdots & \lambda_3^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_N & \lambda_N^2 & \cdots & \lambda_N^{N-1} \end{bmatrix} \tag{B.7}$$

where the right-hand side matrix is an $N\times N$ Vandermonde matrix $V = V(\lambda)$, whose determinant is $\det V(\lambda) = \prod_{j>i}(\lambda_j-\lambda_i) = \prod_{i=1}^{N}\prod_{j=i+1}^{N}(\lambda_j-\lambda_i)$. The Hadamard product $V(\lambda)\circ V(\lambda) = V(\lambda^2)$ is again a Vandermonde matrix. If we denote the left-hand side $N\times N$ matrix in (B.7) by

$$Y = \begin{bmatrix} 1 & \text{diag}\left((A)_{jj}\right)u & \text{diag}\left((A)_{jj}^2\right)u & \cdots & \text{diag}\left((A)_{jj}^{N-1}\right)u \end{bmatrix}$$

then the matrix form of (B.7) is

$$Y = \Xi.V \tag{B.8}$$

From the rank property of a product of matrices, it follows that

$$\text{rank}(Y) \le \min(\text{rank}(\Xi), \text{rank}(V)) \tag{B.9}$$

The determination of rank(Ξ) based on (B.7) is, however, not obvious. Only if all eigenvalues of the matrix A are distinct, then $\det V \neq 0$ and $\Xi = Y.V^{-1}$, which shows that rank(Ξ) = rank(Y), because V is of full rank, i.e. rank(V) = N.

B.1.2. EIGENSTRUCTURE OF THE MATRIX Ξ

Let us denote the eigenvalue equation of the asymmetric[3] $N\times N$ matrix Ξ by

$$\Xi w_j = \xi_j w_j \tag{B.10}$$

Double-stochasticity combined with the Perron-Frobenius theorem tells us that $\xi_1 = 1 \ge |\xi_j|$ for any $j>1$ and $w_1 = u$. Each eigenvalue ξ_j of the asymmetric matrix Ξ thus lies within the unit circle and is either real on $[-1,1]$ or occurs in complex conjugate pairs, i.e. if $\text{Im}\,\xi_j \neq 0$, then existence of ξ_j implies existence of its complex conjugate ξ_j^*. The corresponding eigenvector w_j^* of ξ_j^* follows by taking the complex conjugate of the eigenvalue equation (i.e. replacing i by $-i$), thus $\Xi w_j^* = \xi_j^* w_j^*$. All eigenvalues of Ξ^m, i.e. ξ_j^m for $1\le j\le N$ and for any positive integer m, lie within the unit circle and the largest eigenvalue $\xi_1 = 1$ possesses the all-one vector u as eigenvector. This fact follows from (a) the above eigenvalue equation and (b), separately, from the property that the product of two doubly-stochastic matrices is also a doubly-stochastic matrix. The trace of the matrix Ξ is $\text{trace}(\Xi) = \sum_{j=1}^{N}(x_j)_j^2 \ge 0$, implying that the sum of the eigenvalues of Ξ is non-negative. It follows from $\text{trace}(\Xi^2) = \sum_{j=1}^{N}\xi_j^2 = \sum_{i=1}^{N}\sum_{k=1}^{N}(x_k)_i^2(x_i)_k^2$ that $\sum_{k=1}^{N}(\text{Re}\,\xi_k)^2 \ge \sum_{k=1}^{N}(\text{Im}\,\xi_k)^2$.

[3]Since symmetric orthogonal eigenvector matrices exist [144], their corresponding symmetric Ξ matrices have real eigenvalues in the interval $[-1,1]$.

B

Since the matrix Ξ is asymmetric, the eigenvectors are not necessarily orthogonal, but only independent (provided that Ξ is not defective and that there exist N independent eigenvectors). We find from the eigenvalue equation (B.10) that (a) $w_k^T \Xi w_j = \xi_j w_k^T w_j$ and (b) $w_j^T \Xi w_k = \xi_k w_j^T w_k$ and subtraction

$$\left(\xi_j - \xi_k\right) w_k^T w_j = w_k^T \Xi w_j - w_j^T \Xi w_k = w_k^T \left(\Xi - \Xi^T\right) w_j$$

indicates that orthogonality between w_k and w_j, for $j \neq k$, only holds for symmetric matrices. Thus, $w_j^T w_k$ is not necessarily zero if $k \neq j$.

Lemma 22 *All eigenvectors w_j of a doubly-stochastic matrix Ξ with $j > 1$ are orthogonal to $w_1 = u$.*

Proof: Right-multiplying the transpose of the eigenvalue equation (B.10) by the all-one vector yields $w_j^T \Xi^T u = \xi_j w_j^T u$. After using $\Xi^T u = u$, we find that $0 = \left(\xi_j - 1\right) w_j^T u$, which implies that any eigenvector w_j, except for $w_1 = u$ belonging to $\xi_1 = 1$, is orthogonal to the all-one vector u. □

A consequence of Lemma 22 is that the sum of the components of an eigenvector w_j with $j > 1$ of a doubly-stochastic matrix is zero.

B.1.3. APPLICATION TO THE LAPLACIAN

The Laplacian matrix $Q = \Delta - A$ is a symmetric, semi-definite matrix [51]. The eigenvalue equation $Qz_j = \mu_j z_j$ defines the normalized eigenvector z_j belonging to the eigenvalue μ_j. The set of Laplacian eigenvalues is ordered as $0 = \mu_N \leq \mu_{N-1} \leq \cdots \leq \mu_1$. The matrix equation (B.4) for $k = 1$ relates the degree vector $d = (d_1, d_2, \ldots, d_N)$ of the graph to the eigenvalue vector $\mu = \left(\mu_1, \mu_2, \ldots, \mu_N\right)$,

$$d = \Xi_Q \mu \tag{B.11}$$

where the stochastic matrix Ξ_Q in (2.5) consists of column vectors $\left((z_1)_j^2, (z_2)_j^2, \ldots, (z_N)_j^2\right)$, where $(z_k)_j$ is the j-th component of the k-th eigenvector of Q belonging to μ_k. The general relation (B.7) simplifies for the Laplacian matrix $Q = \Delta - A$ to

$$\begin{bmatrix} 1 & d_1 & \left(Q^2\right)_{11} & \cdots & \left(Q^{N-1}\right)_{11} \\ 1 & d_2 & \left(Q^2\right)_{22} & \cdots & \left(Q^{N-1}\right)_{22} \\ 1 & d_3 & \left(Q^2\right)_{33} & \cdots & \left(Q^{N-1}\right)_{33} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & d_N & \left(Q^2\right)_{NN} & \cdots & \left(Q^{N-1}\right)_{NN} \end{bmatrix} = \Xi_Q \cdot \begin{bmatrix} 1 & \mu_1 & \mu_1^2 & \cdots & \mu_1^{N-1} \\ 1 & \mu_2 & \mu_2^2 & \cdots & \mu_2^{N-1} \\ 1 & \mu_3 & \mu_3^2 & \cdots & \mu_3^{N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

Analogously to the adjacency matrix, also for the Laplacian Q the matrix Ξ_Q is singular, i.e. $\det \Xi_Q = 0$. This follows from orthogonality (2.3) of eigenvectors and the fact that $z_N = \frac{1}{\sqrt{N}} u$, because the sum of the first $N-1$ columns in Ξ_Q in (2.5) is a multiple of the last column. Hence, besides the largest eigenvalue at 1, Ξ_Q (and Ξ_Q^T) has also a zero eigenvalue. The obvious consequence is that $\Xi_Q \mu = d$ in (B.11) cannot be inverted.

However, when deleting the last column corresponding to $\mu_N = 0$ and last row, the resulting $(N-1)\times(N-1)$ matrix $\widetilde{\Xi}_Q$ (a minor of Ξ_Q) can be inverted and the $(N-1)\times 1$ eigenvalue vector $\widetilde{\mu} = \left(\mu_1, \mu_2, \ldots, \mu_{N-1}\right)$ can be determined,

$$\widetilde{\mu} = \left(\widetilde{\Xi}_Q\right)^{-1}\widetilde{d} \tag{B.12}$$

where the vector $\widetilde{d} = (d_1, d_2, \ldots, d_{N-1})$. With $\mu_N = 0$, the entire eigenvalue vector μ is found as a linear function $\mu(d)$ of the unknown degree vector d, where element $d_N = 2L - u^T\widetilde{d}$ follows from $u^T\mu = 2L$, where L denotes the number of links in the graph.

Given the orthogonal eigenvector matrix Z of the Laplacian, the Laplacian eigenvalue vector μ is obtained from (B.12) as a linear combination in the unknown degrees $d_1, d_2, \ldots, d_{N-1}, d_N$ and we can compute

$$Q = Z\text{diag}\left(\mu(d)\right)Z^T$$

Similar as in the proof of Theorem 1, the off-diagonal Laplacian elements $q_{ij} = -a_{ij}$ are zero-one elements, from which the unknown degree vector $\widetilde{d} = (d_1, d_2, \ldots, d_{N-1})$, where the integer degree $d_i \in [0, N-1]$, can be determined by enumeration. In contrast to (B.11) for the Laplacian Q, the zero-diagonal property of the adjacency matrix A, equivalent to $\Xi\lambda = 0$ in (2.8), significantly simplifies the graph recovery process, especially when rank(Ξ) is high.

B.2. Proof of Theorem 1

Section 2.3.1 has illustrated that the minimum rank$(\Xi) = 1$ can appear in the complete graph. We now prove our main result.

Proof of Theorem 1: Given the orthogonal eigenvector matrix X of the adjacency matrix A of an undirected graph, the Hadamard product $\Xi = X \circ X$ in (2.5) can be computed.

If the matrix Ξ has $n \geq 1$ eigenvectors belonging to the zero eigenvalue, then rank$(\Xi) = N - n$ and the dimension n of the kernel or null space obeys $1 \leq n \leq N-1$, because $1 \leq \text{rank}(\Xi) \leq N-1$. The kernel space corresponding to Ξ is spanned by n linearly independent, real vectors $v_1, v_2, \ldots, v_n$ and each vector v_m of the kernel space is orthogonal to all the row vectors of the matrix Ξ. The eigenvalue vector λ, which obeys $\Xi\lambda = 0$ in (2.8), can thus be written as a linear combination of the n independent kernel vectors

$$\lambda = \sum_{m=1}^{n} \beta_m v_m \tag{B.13}$$

where β_m for $1 \leq m \leq n$ are real, unknown numbers. The adjacency matrix $A = X\Lambda X^T$ is constructed with (B.13) as

$$A = \sum_{m=1}^{n} \beta_m X\text{diag}(v_m)X^T \tag{B.14}$$

and each element is $a_{ij} = \sum_{m=1}^{n} \beta_m \left(X\text{diag}(v_m)X^T\right)_{ij}$.

B

We remark that $\left(X\text{diag}(v_m)X^T\right)_{jj} = 0$ for any $1 \le j \le N$. Indeed, using $X\text{diag}(q)X^T = \sum_{k=1}^N q_k x_k x_k^T$ and $\left(x_k x_k^T\right)_{ij} = (x_k)_i (x_k)_j$, yields

$$\left(X\text{diag}(v_m)X^T\right)_{jj} = \left(\sum_{k=1}^N (v_m)_k x_k x_k^T\right)_{jj} = \sum_{k=1}^N (v_m)_k (x_k)_j^2$$

Row j of the eigenvalue equation (B.10) in Section B.1.2 of the matrix Ξ (with $\Xi_{ij} = \left(x_j\right)_i^2$) equals $\sum_{k=1}^N (w_l)_k (x_k)_j^2 = \xi_l (w_l)_j$. Since each vector v_m of the kernel space belongs to eigenvalue $\xi_l = 0$ with multiplicity n in (B.10), we find that $\left(X\text{diag}(v_m)X^T\right)_{jj} = 0$. Thus, the information that the diagonal elements, $a_{jj} = 0$ for $1 \le j \le N$, cannot be used to determine the unknowns $\beta_1, \beta_2, \dots, \beta_n$. Hence, we must invoke the off-diagonal elements of the adjacency matrix.

Any selection of n off-diagonal elements $a_{ij} = \sum_{m=1}^n \beta_m \left(X\text{diag}(v_m)X^T\right)_{ij}$, where $i \ne j$, can be chosen. Without loss of generality, we confine ourselves to n off-diagonal elements that lie on a particular row r, but also n elements a_{ij} on an upper-diagonal (with $j = i + k$ and $k > 0$) may be considered. Row r of the adjacency matrix A, up to column n, is written as the linear set, in which a_{rr} is omitted as equation and replaced by that of element $a_{r;n+1}$,

$$\begin{bmatrix} \left(X\text{diag}(v_1)X^T\right)_{r1} & \left(X\text{diag}(v_2)X^T\right)_{r1} & \cdots & \left(X\text{diag}(v_n)X^T\right)_{r1} \\ \left(X\text{diag}(v_1)X^T\right)_{r2} & \left(X\text{diag}(v_2)X^T\right)_{r2} & \cdots & \left(X\text{diag}(v_n)X^T\right)_{r2} \\ \vdots & \vdots & \ddots & \vdots \\ \left(X\text{diag}(v_1)X^T\right)_{rn} & \left(X\text{diag}(v_2)X^T\right)_{rn} & \cdots & \left(X\text{diag}(v_n)X^T\right)_{rn} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} = \begin{bmatrix} a_{r1} \\ a_{r2} \\ \vdots \\ a_{rn} \end{bmatrix} \tag{B.15}$$

The linear set (B.15) is sufficient to determine all remaining unknowns $\beta_1, \beta_2, \dots, \beta_n$, provided that the rank of the left-hand side $n \times n$ matrix, say M, is n, else a row different from r of the adjacency matrix A must be taken (or generally a different selection of n off-diagonal elements). The $n \times n$ matrix M with $\text{rank}(M) = n$ can be inverted and the unknowns $\beta_1, \beta_2, \dots, \beta_n$ can be expressed in terms of the partial row vector $(a_{r1}, a_{r2}, \dots, a_{rn})$. The only complicating factor is that the partial row vector $(a_{r1}, a_{r2}, \dots, a_{rn})$ is not precisely known, only that each element is either zero or one. A recipe for any chosen row $1 \le r \le N$ is to (i) create all $2^n - 1$ possible partial, zero-one row vectors $(a_{r1}, a_{r2}, \dots, a_{rn})$, excluding all zeros, (ii) determine all unknowns $\beta_1, \beta_2, \dots, \beta_n$ by solving the set (B.15) and (iii) compute the eigenvalue vector λ from (B.13) and (iv) check whether the resulting matrix $X\text{diag}(\lambda)X^T$ is a zero-one matrix, which is a possible adjacency matrix corresponding to the orthogonal eigenvector matrix X. Equation $\Xi\lambda = 0$ in (2.8) ensures that there must at least be one partial row vector $(a_{r1}, a_{r2}, \dots, a_{rn})$ out of the $2^n - 1$ possible combinations that leads to a zero-one matrix.

We cannot exclude, however, that only one adjacency matrix is retrieved. In other words, it may happen that $l > 1$ different adjacency matrices of l different undirected graphs are found, that all possess the same orthogonal eigenvector matrix X, but a different eigenvalue vector λ. □

C

APPENDIX FOR CHAPTER 3

C.1. $N \times N$ MATRIX F_k

We derive the first two sum terms of the $N \times N$ matrix F_k in (3.8). Firstly, we split the first term in (3.7) into two sum terms, where for the second sum term it holds $j_2 = k$

$$\sum_{i_1=0}^{k-2} \sum_{j_1=i+2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k]\right) = \sum_{i_1=0}^{k-3} \sum_{j_1=i_1+2}^{k-1} M\left(\mathscr{W}_{(i_1,j_1)}[k]\right) + \sum_{i_2=0}^{k-2} M\left(\mathscr{W}_{(i_2,k)}[k]\right).$$

Because the counters $i_1 < k$ and $j_1 < k$ of the first sum terms in the equation above are always smaller than k, we import (3.2) and further transform the equation above

$$\sum_{i_1=0}^{k-2} \sum_{j_1=i+2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k]\right) - \sum_{i_1=0}^{k-3} \sum_{j_1=i_1+2}^{k-1} M\left(\mathscr{W}_{(i_1,j_1)}[k-1]\right) \cdot A = \sum_{i_2=0}^{k-2} A^{i_2} \cdot \left(I \circ A^{k-i_2}\right).$$

Similarly, we transform the second sum term in (3.7) by excluding the case $j_2 = k$ from the sum term

$$\begin{aligned}\sum_{i_1=0}^{k-2} \sum_{j_1=i_1+2}^{k} \sum_{i_2=i_1}^{k-2} \sum_{j_2=q_2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k] \cap \mathscr{W}_{(i_2,j_2)}[k]\right) &= \sum_{i_1=0}^{k-2} \sum_{j_1=i_1+2}^{k} \sum_{i_2=i_1}^{k-2} \sum_{j_2=q_2}^{k-1} M\left(\mathscr{W}_{(i_1,j_1)}[k] \cap \mathscr{W}_{(i_2,j_2)}[k]\right) \\ &+ \sum_{i_1=0}^{k-2} \sum_{j_1=i_1+2}^{k} \sum_{i_2=i_1}^{k-2} M\left(\mathscr{W}_{(i_1,j_1)}[k] \cap \mathscr{W}_{(i_2,k)}[k]\right)\end{aligned} \tag{C.1}$$

We further exclude the case $j_1 = k$ from both sum term on the right-hand side of (C.1) and obtain

$$
\begin{aligned}
\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,j_2)}[k]\right) &= \sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k-1} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,j_2)}[k]\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k-1} M\left(\mathcal{W}_{(i_1,k)}[k]\cap\mathcal{W}_{(i_2,j_2)}[k]\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-2} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,k)}[k]\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{i_2=i_1}^{k-2} M\left(\mathcal{W}_{(i_1,k)}[k]\cap\mathcal{W}_{(i_2,k)}[k]\right)
\end{aligned}
\tag{C.2}
$$

The first sum term can be split in two sub walks as in (3.1), transforming (C.2) as follows

$$
\begin{aligned}
\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,j_2)}[k]\right) &= \sum_{i_1=0}^{k-3}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-3}\sum_{j_2=q_2}^{k-1} M\left(\mathcal{W}_{(i_1,j_1)}[k-1]\cap\mathcal{W}_{(i_2,j_2)}[k-1]\right)\cdot A\\
&+\sum_{i_1=0}^{k-2}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k-1} A^{i_1}\cdot\left(I\circ\left(A^{i_2-i_1}\cdot\left(I\circ A^{j_2-i_2}\right)\cdot A^{k-j_2}\right)\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-2} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,k)}[k]\right)\\
&+\sum_{i_1=0}^{k-4}\sum_{i_2=i_1+2}^{k-2} A^{i_1}\cdot\left(I\circ A^{i_2-i_1}\right)\cdot\left(I\circ A^{k-i_2}\right)
\end{aligned}
\tag{C.3}
$$

The third sum term on the right-hand side of the relation above can be split into three sum terms, where $i_2 < j_1$, secondly $i_2 = j_1$ and finally where $i_2 > j_1$, allowing us to transform (C.3) as follows

$$
\begin{aligned}
\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,j_2)}[k]\right) &= \sum_{i_1=0}^{k-3}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-3}\sum_{j_2=q_2}^{k-1} M\left(\mathcal{W}_{(i_1,j_1)}[k-1]\cap\mathcal{W}_{(i_2,j_2)}[k-1]\right)\cdot A\\
&+\sum_{i_1=0}^{k-2}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k-1} A^{i_1}\cdot\left(I\circ\left(A^{i_2-i_1}\cdot\left(I\circ A^{j_2-i_2}\right)\cdot A^{k-j_2}\right)\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{j_1-1} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,k)}[k]\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(j_1,k)}[k]\right)\\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=j_1+2}^{k-1}\sum_{i_2=i_1}^{k-2} M\left(\mathcal{W}_{(i_1,j_1)}[k]\cap\mathcal{W}_{(i_2,k)}[k]\right)\\
&+\sum_{i_1=0}^{k-4}\sum_{i_2=i_1+2}^{k-2} A^{i_1}\cdot\left(I\circ A^{i_2-i_1}\right)\cdot\left(I\circ A^{k-i_2}\right).
\end{aligned}
\tag{C.4}
$$

Finally, the second sum term of the $N \times N$ matrix F_k is defined as follows

$$
\begin{aligned}
\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k} M\left(\mathscr{W}_{(i_1,j_1)}[k]\cap\mathscr{W}_{(i_2,j_2)}[k]\right) = &\sum_{i_1=0}^{k-3}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{k-3}\sum_{j_2=q_2}^{k-1} M\left(\mathscr{W}_{(i_1,j_1)}[k-1]\cap\mathscr{W}_{(i_2,j_2)}[k-1]\right)\cdot A \\
&+\sum_{i_1=0}^{k-2}\sum_{i_2=i_1}^{k-2}\sum_{j_2=q_2}^{k-1} A^{i_1}\cdot\left(I\circ\left(A^{i_2-i_1}\cdot\left(I\circ A^{j_2-i_2}\right)\cdot A^{k-j_2}\right)\right) \\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=i_1}^{j_1-1} A^{i_1}\cdot\left(A^{i_2-i_1}\circ A^{j_1-i_2}\circ A^{k-j_1}\right) \\
&+\sum_{i_1=0}^{k-4}\sum_{j_1=i_1+2}^{k-2} A^{i_1}\left(I\circ A^{j_1-i_1}\right)\cdot\left(I\circ A^{k-j_1}\right) \\
&+\sum_{i_1=0}^{k-2}\sum_{j_1=i_1+2}^{k-1}\sum_{i_2=j_1+1}^{k-2} A^{i_1}\cdot\left(I\circ A^{j_1-i_1}\right)\cdot A^{i_2-j_1}\cdot\left(I\circ A^{k-i_2}\right) \\
&+\sum_{i_1=0}^{k-4}\sum_{i_2=i_1+2}^{k-2} A^{i_1}\cdot\left(I\circ A^{i_2-i_1}\right)\cdot\left(I\circ A^{k-i_2}\right).
\end{aligned}
\tag{C.5}
$$

C

C.2. COUNTING THE NUMBER OF LENGTH k PATHS RECURSIVELY

DETERMINEKPATHS (A, N, k)

Input: A, N, k
Output: P
1. $P \leftarrow O_{N\times N}$
2. **for** $i \leftarrow 1$ to N
3. $\quad T \leftarrow$ COMPUTEKPATHS ($O_{N\times 1}$, A, i, 0, k)
4. $\quad$ Store T as the i-th row of P
5. **end for**
6. **return** P

Figure C.1: Metacode for calling the recursive algorithm for determining the number of length k paths between node pairs in a graph, with the graph size N, hopcount k and the $N \times N$ adjacency matrix A as input. Algorithm returns the $N \times N$ path matrix P_k

Compared to the recursive algorithm shown in Figure 3.10, the modified recursive function in Figure C.2 increments the $N \times 1$ node-based path vector T by considering the neighbours j of the destination node n_h (line 4). This is done only if the hop count h of these paths matches the input hop count k (line 3). If they match, the recursion ends and the path vector T is returned (line 5). Otherwise, the recursion is called for any neighbour j of the destination node n_h (line 10) that has a non-zero degree (line 9), after removing all links adjacent to the destination node n_h (line 7).

The adjusted recursion is called as outlined in Algorithm C.1, where initially the $N \times N$ length k path matrix P_k is initialised as a zero matrix (line 1). Next, the recursive function is called for each node (line 3), while the $N \times 1$ node-based path vector T, obtained for

COMPUTEKPATHS (T, A, n_h, h, k)

Input: T, A, n_h, h, k
Output: T
1. $h \leftarrow h+1$
2. $\mathcal{N}_{n_h} \leftarrow \{j \mid j \in \mathcal{N}, a_{n_h,j} = 1\}$
3. **if** $h = k$
4. $\quad T_j \leftarrow T_j + 1$, where $j \in \mathcal{N}_{n_h}$
5. $\quad$ **return** T
6. **else**
7. $\quad a_{n_h,j} \leftarrow 0$ and $a_{j,n_h} \leftarrow 0$, where $j \in \mathcal{N}_{n_h}$
8. $\quad$ **for** $m \leftarrow 1$ to $|\mathcal{N}_{n_k}|$
9. $\quad\quad$ **if** $|\mathcal{N}_{j_m}| > 0$
10. $\quad\quad\quad T \leftarrow$ COMPUTEKPATHS (T, A, j_m, h, k)
11. $\quad\quad$ **end if**
12. $\quad$ **end for**
13. **end if**
14. **return** T

Figure C.2: Metacode of the recursive algorithm for determining all length k paths in a graph, with the $N \times N$ adjacency matrix A, the $(N-1) \times N$ node-based path matrix T, destination node n_k and hopcount k as input. The recursive function returns the $(N-1) \times N$ node-based path matrix T.

node i is stored as the ith row of P_k (line 4).

D

Appendix for Chapter 5

D.1. Clustering algorithms

D.1.1. Louvain Method

The Louvain method is a simple, yet powerful heuristic clustering algorithm, proposed by Blondel *et al.* [14]. The method is based on an iterative, unsupervised, two-step procedure that optimizes modularity m. Initially, a directed graph G with an $N \times N$ weighted adjacency matrix M is partitioned in N clusters, where each node constitutes its own cluster or community.

In the first stage, the algorithm examines how the graph modularity m changes if a node i would be assigned to a community of its neighbouring node $j \in \mathcal{N}_i$. The modularity gain Δm in case node i is assigned to community h of adjacent node j has been determined in [14] as

$$\begin{aligned}\Delta m = &\left(\frac{\sum_{\text{in}} + 2\sum_{l:C_{lj}=1} M_{il}}{2L} - \left(\frac{\sum_{\text{tot}} + d_i}{2L} \right)^2 \right) - \\ &\left(\frac{\sum_{\text{in}}}{2L} - \left(\frac{\sum_{\text{tot}}}{2L} \right)^2 - \left(\frac{d_i}{2L} \right)^2 \right),\end{aligned} \tag{D.1}$$

where the sum of the weights of intra-community links in h is $\sum_{\text{in}}$, while $\sum_{\text{tot}}$ denotes the sum of the weights of all links in G incident to any node in community h. Node i is assigned to the community with the largest positive gain in modularity m. In case all computed gains Δm are either negative or smaller than a predefined small positive threshold value, node i remains in its original community. The first stage ends when modularity m cannot be further increased by re-assigning nodes to communities of neighbours.

In the second stage of an iteration, the weighted graph from the first stage is transformed into a new weighted graph, where each community is presented by a node. The link weight between two nodes h and g equals the sum of weights of all links between communities h and g in the graph from the first stage. Furthermore, the weight of a self-loop of node g in the new graph equals the sum of weights of all intra-community links in cluster g of the graph from the previous stage. The new graph is provided to the first stage in the next iteration. The algorithm stops when modularity m cannot be increased further. The time complexity of the Louvain method is linear in the number of links $O(L)$ on sparse graphs [14].

D.1.2. LEIDEN METHOD

The Louvain method, while being one of the most popular clustering algorithms in the literature, suffers from identifying poorly connected or even disconnected communities. This defect was first discovered by Traag *et al.*, who proposed the Leiden algorithm in [145], an improvement of the Louvain method that estimated graph partition while guaranteeing connected communities. The Leiden algorithm consists of three iterative steps:

D

1 Local moving of nodes, an improved version of the first step of the Louvain algorithm, described in (D.1). Louvain algorithm visits each node randomly until modularity cannot be improved by moving a node to a different community. While doing so, Louvain also visits nodes that cannot be moved. On the contrary, the Leiden algorithm visits only those nodes whose adjacent nodes have been moved. It is achieved by placing nodes in a queue and iteratively checking whether it is possible to improve the quality function by updating the cluster membership of a node. When a node is moved to another community, its neighbours from other communities are placed in the queue.

2 Refinement of the partition, where each node is assigned its own community. Nodes are only locally merged, i.e. within communities estimated in the previous stage. Two nodes from the same community are merged only in case both nodes are well connected to the community from the previous stage. At the end of the refinement stage, partitions from the first stage are often split into multiple communities.

3 Aggregation of the network, based on the refined partition from the previous step, as in the second stage of the Louvain algorithm.

The Leiden algorithm performs clustering faster than the Louvain algorithm while providing in general between partitions [145]. In Section 5.6, we compare the performance of the Leiden algorithm with the proposed linear clustering process on both synthetic and real-world networks.

D.1.3. NEWMAN'S METHOD OF OPTIMAL MODULARITY

Newman [84] proposed a clustering algorithm that is based on modularity optimisation. The algorithm starts with estimating the bisection of a graph G, generating the highest modularity m from (5.3), that can be rewritten as follows:

$$m = \frac{1}{4L} y^T \cdot M \cdot y, \tag{D.2}$$

where the $N \times 1$ vector y is composed of values 1 and -1, denoting cluster membership of each node, while the $N \times N$ modularity matrix $M = A - \frac{1}{2L} \cdot d \cdot d^T$ has the following eigenvalue decomposition

$$M = \sum_{i=1}^{N} \zeta_i \cdot z_i \cdot z_i^T, \tag{D.3}$$

where the $N \times 1$ eigenvector z_i corresponds to the i-th eigenvalue ζ_i. Further, the vector $y = \sum_{j=1}^{N} (z_j^T \cdot y) \cdot z_j$ can be written as a linear combination of eigenvectors $\{z_i\}_{1 \le i \le N}$, which transforms (D.2) to

$$m = \frac{1}{4L} \sum_{i=1}^{N} \zeta_i \cdot (z_j^T \cdot y)^2. \tag{D.4}$$

In order to maximise the modularity m, Newman [84] proposed to define $y_i = 1$, in case $(z_1)_i > 0$, otherwise $y_i = -1$. In a next iteration, the same procedure of spectral division into two partitions is repeated on both sub-graphs. However, using only the block sub-matrix of M, corresponding to cluster g in next iteration would not take into account inter-community links. Instead, for the estimated cluster g, the modularity matrix M_g is updated as

$$M_g = m_{ij} - \left(\sum_{k \in g} m_{ik} \right) \cdot \delta_{ij}, \tag{D.5}$$

where Kronecker delta $\delta_{ij} = 1$ if $i = j$, otherwise $\delta_{ij} = 0$. The algorithm stops when the modularity m cannot be further improved.

D.1.4. NON-BACK TRACKING MATRIX

The non-back tracking clustering method estimates the number of clusters in a network, based on the spectrum of the non-back tracking matrix B, that contains information about 2-hop directed walks in a network G, that are not closed [19]. Given an undirected network $G(\mathcal{N}, \mathcal{L})$, for each link $i \sim j$ between nodes i and j, two directed links $(i \to j)$ and $(j \to i)$ are created. By transforming each link in G into a bi-directional link pair, we compose in total $2L$ links. The $2L \times 2L$ non-back tracking matrix B is defined as follows:

$$B_{(u \to v),(w \to z)} = \begin{cases} 1 & \text{if } v = w \text{ and } u \neq z \\ 0 & \text{otherwise,} \end{cases} \tag{D.6}$$

where $v, w, z \in \mathcal{N}$. Since the non-back tracking matrix B is asymmetric, its eigenvalues are generally complex. Furthermore, a vast majority of eigenvalues lie within a circle in complex plain, with centre at the origin and with radius equal to the square root of the largest eigenvalue. Krzakala *et al.* [19] hypothesized that the number of clusters in G equals the number of real-valued eigenvalues outside the circle. Computing the eigenvalues of the non-back tracking matrix B is of computational complexity $O(L^3)$. However, the complexity can be reduced to $O(N^3)$, as explained in [9, p. 20]. The non-back tracking matrix method is denoted as NBTM in Section 5.6.

D.2. RANDOM GRAPH BENCHMARKS

D.2.1. STOCHASTIC BLOCK MODEL

In this paper, we focus on the symmetric stochastic block model (SSBM), where only two different link probabilities are defined. Two nodes are connected via a link with probability p_{in} if they belong to the same cluster, otherwise, the direct link exists with probability p_{out}. Communities emerge when the link density within clusters is larger than the inter-community link probability $p_{in} > p_{out}$. Furthermore, we restrict clusters to be of the same size:

$$n_i = \frac{N}{c}, i \in \{1, 2, \ldots, c\},$$

which causes the expected degree to be the same for each node:

$$E[D] = \frac{b_{in} + (c-1) \cdot b_{out}}{c}, \tag{D.7}$$

irrespective of its cluster membership. We further consider a sparse and assortative SSBM. The SBMM is called sparse and assortative when the link probabilities $p_{in} = \frac{b_{in}}{N}$ and $p_{out} = \frac{b_{out}}{N}$ are defined upon positive constants $b_{in} > b_{out}$ that stay constant when $N \to \infty$. Decelle *et al.* [146, 147] found that when the difference $b_{in} - b_{out}$ is above the detectability threshold

$$b_{in} - b_{out} > c \cdot \sqrt{E[D]}, \tag{D.8}$$

it is theoretically possible to recover cluster membership of the nodes; otherwise, the community structure of a network is not distinguishable from randomness. The threshold (D.8) marks a phase transition between the undetectable and the theoretically detectable regime of the SSBM.

D.2.2. LFR BENCHMARK

Lancichinetti *et al.* proposed in [97] the LFR benchmark, providing more realistic random graphs with a built-in community structure than SSBM graphs. Opposite to SSBM graphs (where each node has the same expected degree), the authors argue that the degree distributions in real-world networks are usually heterogeneous. Furthermore, the tails of degree distributions often obey the power law [148]. Next, by restricting clusters to be the same size, we neglect the observed properties of community size distribution in real-world networks that are often heavy-tailed [149]. Therefore, the LFR benchmark produces a graph with the following characteristics:

1 Each node has a degree sampled from a power law distribution, whose exponent equals the input parameter γ.

2 The size of each community is sampled from a power law distribution, whose exponent equals the input parameter β.

3 A fraction $1-\mu$ of each node's links are intra-community.

In addition to the above-introduced parameters, the LFR benchmark assumes the network size N, the average degree d_{av} and the number of communities c as inputs.

D.3. LCP IN CONTINUOUS TIME

We explain the physical intuition of our clustering process in continuous time t, where the position $x_i(t)$ of a node i is assumed to change continuously with time t. The change $x_i(t+\Delta t) - x_i(t)$ in position of node i at time t for small increments Δt is proportional to the sum over neighbours j of the difference $x_j(t) - x_i(t)$ in position weighted by the resultant force between attraction and repulsion:

$$\begin{aligned}\frac{dx_i(t)}{dt} = \sum_{j\in\mathcal{N}_i} \Bigg(&\frac{\alpha\cdot\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right|+1\right)}{d_j d_i} - \\ &\frac{\frac{1}{2}\cdot\delta\cdot\left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{d_j d_i}\Bigg)\cdot\left(x_j(t) - x_i(t)\right)\end{aligned} \tag{D.9}$$

where α and δ are, in the continuous-time setting, the rates (with units s^{-1}) for attraction and repulsion, respectively. The law (D.9) of the nodal positioning $x_i(t)$ for each $i \in \mathcal{N}$ deviates from physical repulsion between charged particles, where the force is proportional to $\left(x_j(t) - x_i(t)\right)^{-b}$ for some positive number b. The important advantage of the law (D.9) is its linearity that allows an exact mathematical treatment. The linear dynamic process (D.9) is proportional to the underlying graph, which we aim to cluster; a non-linear law depends intricately on the underlying graph and may result in a lesser clustering. The drawback of the linear dynamic process (D.9), as investigated below in Section 5.3.3, lies in the steady state, where the attractive and repulsive forces are precisely in balance.

After dividing both sides by δ,

$$\frac{dx_i(t)}{d(\delta t)} = \sum_{j\in\mathcal{N}_i} \left(\frac{\frac{\alpha}{\delta}\cdot\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right|+1\right)}{d_i d_j} - \frac{\frac{1}{2}\cdot\left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{d_i d_j}\right)\cdot\left(x_j(t) - x_i(t)\right)$$

and defining the normalized time by $t^* = \delta t$ and the effective attraction rate $\tau = \frac{\alpha}{\delta}$, the governing equation (D.9) reduces to

$$\frac{dx_i(t^*)}{dt^*} = \sum_{j\in\mathcal{N}_i} \left(\frac{\tau\cdot\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right|+1\right)}{d_i d_j} - \frac{\left(\frac{\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right|}{2} - 1\right)}{d_i d_j}\right)\cdot\left(x_j(t^*) - x_i(t^*)\right) \tag{D.10}$$

The position $x_i(t^*)$ of node i is now expressed in the dimensionless time t^*, where the actual time $t = \frac{t^*}{\delta}$ is measured in units of $\frac{1}{\delta}$. By scaling or normalizing the time, the repulsion strength or rate δ has been eliminated, illustrating that the clustering process only depends upon one parameter, the effective attraction rate τ. Relation (5.1) indicates that the weight of the position difference

$$w_{ij} = \frac{\tau\cdot\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right|+1\right) - \left(\frac{\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right|}{2} - 1\right)}{d_i d_j}$$

lies in the interval $\left(-\frac{\frac{d_i+d_j}{2}-1}{d_i d_j}, \frac{\tau}{d_i}\right)$ and that the elements $w_{ij} = w_{ji}$ define the symmetric $N \times N$ weight matrix W, which is specified in (5.10). Although symmetry is physically not required [1], the analysis below is greatly simplified, because eigenvalues and eigenvectors of a symmetric matrix are real.

We rewrite the law (D.10) as

$$\frac{dx_i(t^*)}{dt^*} = \sum_{j\in\mathcal{N}_i} w_{ij}x_j(t^*) - x_i(t^*)\sum_{j\in\mathcal{N}_i} w_{ij} \sum_{j=1}^{N} a_{ij}w_{ij}x_j(t^*) - x_i(t^*)v_i$$

where $v_i = \sum_{j\in\mathcal{N}_i} w_j = \sum_{j=1}^{N} a_{ij}w_{ij}$ is independent of time t^*. The cluster positioning law (D.9) for the vector $x(t^*)$ in continuous time is, in matrix form,

$$\frac{dx(t^*)}{dt^*} = (A\circ W - \text{diag}(v))x(t^*) \tag{D.11}$$

where the Hadamard product [94] is denoted by $\circ$ and the vector $v = (A\circ W)u$. The corresponding solution of (D.11) is [150, eq. (6)]

$$x(t^*) = e^{(A\circ W - \text{diag}((A\circ W)u))t^*}x(0) \tag{D.12}$$

which illustrates that a steady state is reached, provided that the real part of the largest eigenvalue of the matrix $H = (A\circ W - \text{diag}((A\circ W)u))$ is not positive.

D.4. PROOF OF THEOREMS

D.4.1. PROOF OF THEOREM 19

Similarly as in Section D.3, we rewrite the sum over all neighbours in the governing equation (5.7) in terms of the elements of the $N\times N$ adjacency matrix A:

$$x_i[k+1] - x_i[k] = \sum_{j=1}^{N} \frac{a_{ij}}{d_i d_j}\Big(x_j[k] - x_i[k]\Big)\Big(\alpha\left|\mathcal{N}_j\cap\mathcal{N}_i\right| - \frac{1}{2}\delta\left(\left|\mathcal{N}_j\setminus\mathcal{N}_i\right| + \left|\mathcal{N}_i\setminus\mathcal{N}_j\right|\right)\Big). \tag{D.13}$$

Firstly, we denote the $N\times 1$ vector $\tilde{d} = \Delta^{-1}\cdot u$ composed of the inverse nodal degrees:

$$\tilde{d} = \begin{bmatrix} \frac{1}{d_1} & \frac{1}{d_2} & \cdots & \frac{1}{d_N} \end{bmatrix}^T \tag{D.14}$$

In the sequel, we will deduce the corresponding matrix form of (D.13). With (5.1) and (5.2), the degree d_i of node i distracted by the number of common neighbours between nodes i and j (5.2), equals the number of node i neighbours, not adjacent to node j:

$$\left|\mathcal{N}_i\setminus\mathcal{N}_j\right| = \left(d\cdot u^T - A^2\right)_{ij}. \tag{D.15}$$

Similarly, the number of node j neighbours that do not share link with node i has following matrix form:

$$\left|\mathcal{N}_j\setminus\mathcal{N}_i\right| = \left(u\cdot d^T - A^2\right)_{ij}. \tag{D.16}$$

[1] The process described by

$$\frac{dx_i(t)}{dt} = \sum_{j\in\mathcal{N}_i} \frac{\alpha\cdot\left(\left|\mathcal{N}_j\cap\mathcal{N}_i\right|+1\right) - \delta\cdot\left(\left|\mathcal{N}_j\setminus\mathcal{N}_i\right|-1\right)}{d_j d_i}\cdot\left(x_j(t) - x_i(t)\right)$$

also works.

Finally, the position difference $(x_j[k]-x_i[k])$ between nodes i and j at time k equals the ij-th element of the matrix below:

$$\left(x_j[k]-x_i[k]\right)=\left(u\cdot x^T[k]-x[k]\cdot u^T\right)_{ij}, \tag{D.17}$$

while dividing by node i (j) degree d_i (d_j) is equivalent to product with the ij-th element of the $N\times N$ matrix $(\tilde{d}\cdot u^T)_{ij}$ and $(u\cdot\tilde{d}^T)_{ij}$, respectively. By implementing matrix notations (5.2), (D.15), (D.16) and (D.17) into the governing equation (D.13) and by applying the distributive property of the Hadamard product [94, p. 477] we obtain:

$$\begin{aligned} x[k+1]-x[k]=&\Big(\left(u\cdot x^T[k]-x[k]\cdot u^T\right)\circ A\circ \\ &\left(u\cdot\tilde{d}^T\right)\circ\left(\tilde{d}\cdot u^T\right)\circ\Big((\alpha+\delta)\cdot\left(A^2+A\right)- \\ &\frac{1}{2}\delta\cdot\left(u\cdot d^T+d\cdot u^T\right)\Big)\Big)\cdot u. \end{aligned} \tag{D.18}$$

We define the $N\times N$ topology-based matrix W as follows:

$$\begin{aligned} W=&A\circ\left(u\cdot\tilde{d}^T\right)\circ\left(\tilde{d}\cdot u^T\right)\circ \\ &\left((\alpha+\delta)\cdot\left(A^2+A\right)-\frac{1}{2}\delta\left(u\cdot d^T+d\cdot u^T\right)\right). \end{aligned} \tag{D.19}$$

Using the distributive property of a Hadamard product [94, p. 477], we develop the equation (D.19) further:

$$\begin{aligned} W=&(\alpha+\delta)\cdot\left(\left(u\cdot\tilde{d}^T\right)\circ\left(\tilde{d}\cdot u^T\right)\circ A\circ\left(A^2+A\right)\right)- \\ &\frac{1}{2}\delta\left(A\circ\left(u\cdot\tilde{d}^T\right)\circ\left(\tilde{d}\cdot u^T\right)\circ\left(u\cdot d^T\right)\right)- \\ &\frac{1}{2}\delta\left(A\circ\left(u\cdot\tilde{d}^T\right)\circ\left(\tilde{d}\cdot u^T\right)\circ\left(d\cdot u^T\right)\right). \end{aligned} \tag{D.20}$$

Since the Hadamard product is commutative [94, p. 477], we can reorder the products in previous equation. The Hadamard product $\left(u\cdot\tilde{d}^T\right)\circ\left(u\cdot d^T\right)$ equals all-one matrix J. Similarly, the product $\left(\tilde{d}\cdot u^T\right)\circ\left(d\cdot u^T\right)=J$. We further transform the Hadamard product of $\left(A\circ A^2+A\right)$ and the outer products $\left(u\cdot\tilde{d}^T\right)$ and $\left(\tilde{d}\cdot u^T\right)$ into product with diagonal matrices $\Delta^{-1}\cdot\left(A\circ A^2+A\right)\cdot\Delta^{-1}$. Thus, equation (D.20) transforms to (5.10). Substituting (D.19) into (D.18) yields

$$x[k+1]-x[k]=\left(\left(u\cdot x^T[k]-x[k]\cdot u^T\right)\circ W\right)\cdot u. \tag{D.21}$$

The Hadamard product of a matrix with an outer product of two vectors is equivalent to the product with diagonal matrices of vectors composing the outer product [94, p. 477]. Thus, we further transform the governing equation (D.21):

$$x[k+1]-x[k]=W\cdot\operatorname{diag}(x[k])\cdot u-\operatorname{diag}(x[k])\cdot(W\cdot u), \tag{D.22}$$

where the last term $\operatorname{diag}(x[k])\cdot(W\cdot u)$ represents the Hadamard product of two vectors, $x[k]\circ(W\cdot u)$ and can be presented as $\operatorname{diag}(W\cdot u)\cdot x[k]$. Thus, the equation transforms into (5.9) which completes the proof. □

D

D.4.2. PROOF OF PROPERTY 2

We observe that

$$W \cdot u - \operatorname{diag}(W \cdot u) \cdot u = 0$$

implying that the all-one vector u is an eigenvector of the matrix $W - \operatorname{diag}(W \cdot u)$ belonging to the zero eigenvalue. Therefore, the $N \times N$ matrix $I + W - \operatorname{diag}(W \cdot u)$ has an eigenvalue 1 corresponding to the all-one vector u.

By the Perron-Frobenius theorem [4] for a non-negative matrix, the principal eigenvector, belonging to the largest eigenvalue, has non-negative components. Since the eigenvector u has non-negative components and all eigenvectors of a symmetric matrix are orthogonal, it follows that the all-one vector u is the Perron or principal eigenvector belonging to the largest eigenvalue 1 of the matrix $I + W - \operatorname{diag}(W \cdot u)$ and, thus, all other real eigenvalues are, in absolute value, smaller than 1. □

D.4.3. PROOF OF PROPERTY 3

The non-negativity of the matrix $I + W - \operatorname{diag}(W \cdot u)$ implies that $w_{ij} \geq 0$ for $i \neq j$ and $1 + w_{ii} - \sum_{k=1}^{N} w_{ik} \geq 0$, hence,

$$1 \geq \sum_{k=1;k\neq i}^{N} w_{ik} \geq 0$$

Equivalently, the symmetric matrix $W - \operatorname{diag}(W \cdot u)$ has positive off-diagonal elements, but negative diagonal elements, similarly to the infinitesimal generator of a Markov chain (which is minus a weighted Laplacian [64]). Introducing the explicit expression (5.11) and requiring that each element w_{ij} is non-negative,

$$w_{ij} = a_{ij} \frac{\alpha \cdot \left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right) - \delta \cdot \left(\frac{\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right|}{2} - 1\right)}{d_i d_j} \geq 0$$

leads to

$$\frac{\alpha}{\delta} \geq \frac{1}{2} \cdot \frac{\left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)}$$

which holds for any $i, j \neq i \in \mathcal{N}$. With (5.1), (5.2) and $d_i = (Au)_i = (A^2)_{ii}$, the condition for the ratio $\frac{\alpha}{\delta}$ becomes

$$\begin{aligned}
\frac{\alpha}{\delta} &\geq \max_{i,j\neq i\in\mathcal{N}} \frac{1}{2} \cdot \frac{\left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{\left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)} \\
&= \max_{i,j\neq i\in\mathcal{N}} \frac{d_i + d_j}{2\left((A^2)_{ij} + 1\right)} - 1
\end{aligned}$$

which simplifies to

$$\frac{\alpha}{\delta} \geq d_{\max} - 1 \tag{D.23}$$

We write $\sum_{k=1;k\neq i}^{N} w_{ik}$ with (5.11) as

$$\sum_{k=1;k\neq i}^{N} w_{ik} = \frac{1}{d_i}\sum_{k=1}^{N} a_{ik}\Bigg(\frac{\alpha\cdot(|\mathcal{N}_k\cap\mathcal{N}_i|+1)}{d_k} - \frac{\frac{\delta}{2}\cdot(|\mathcal{N}_k\setminus\mathcal{N}_i|+|\mathcal{N}_i\setminus\mathcal{N}_k|-2)}{d_k}\Bigg)$$

Introducing (5.1) and (5.2),

$$\begin{aligned}\sum_{k=1;k\neq i}^{N} w_{ik} &= \frac{1}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\alpha\cdot\big((A^2)_{ik}+1\big)-\frac{\delta}{2}\cdot\big(d_i+d_k-2\big((A^2)_{ik}+1\big)\big)}{d_k}\\ &= \frac{\alpha}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\big((A^2)_{ik}+1\big)}{d_k} - \frac{\delta}{2d_i}\sum_{k=1}^{N} a_{ik}\left(\frac{d_i}{d_k}+1-\frac{2\big((A^2)_{ik}+1\big)}{d_k}\right)\end{aligned}$$

leads, with $d_i = \sum_{k=1}^{N} a_{ik}$, to

$$\sum_{k=1;k\neq i}^{N} w_{ik} = \frac{\alpha+\delta}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\big((A^2)_{ik}+1\big)}{d_k} - \frac{\delta}{2} - \frac{\delta}{2}\sum_{k=1}^{N}\frac{a_{ik}}{d_k}$$

The second condition $\sum_{k=1;k\neq i}^{N} w_{ik} \le 1$,

$$\frac{\alpha+\delta}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\big((A^2)_{ik}+1\big)}{d_k} - \frac{\delta}{2} - \frac{\delta}{2}\sum_{k=1}^{N}\frac{a_{ik}}{d_k} \le 1$$

must hold for all $i\in\mathcal{N}$, which translates to

$$\begin{aligned}1 &\ge \max_{i\in\mathcal{N}}\left(\frac{\alpha+\delta}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\big((A^2)_{ik}+1\big)}{d_k} - \frac{\delta}{2} - \frac{\delta}{2}\sum_{k=1}^{N}\frac{a_{ik}}{d_k}\right)\\ &\ge (\alpha+\delta)\max_{i\in\mathcal{N}}\frac{1}{d_i}\sum_{k=1}^{N} a_{ik}\frac{\big((A^2)_{ik}+1\big)}{d_k} - \frac{\delta}{2} - \frac{\delta}{2}\min_{i\in\mathcal{N}}\sum_{k=1}^{N}\frac{a_{ik}}{d_k}\end{aligned}$$

With $\big((A^2)_{ik}+1\big) \le d_k$ if $a_{ik}=1$, we have $\frac{1}{d_i}\sum_{k=1}^{N} a_{ik}\frac{((A^2)_{ik}+1)}{d_k} \le 1$, while $\frac{d_i}{d_{\min}} \ge \sum_{k=1}^{N}\frac{a_{ik}}{d_k} \ge \frac{d_i}{d_{\max}}$. Hence, the second condition becomes

$$1 \ge \alpha + \frac{\delta}{2}\left(1-\frac{d_{\min}}{d_{\max}}\right) \tag{D.24}$$

illustrating that $\alpha \le 1$. Combining the two conditions (D.23) and (D.24) into a linear set of inequalities

$$\begin{cases} 0 \ge -\alpha+\delta\,(d_{\max}-1)\\ 1 \ge \alpha+\frac{\delta}{2}\left(1-\frac{d_{\min}}{d_{\max}}\right)\end{cases}$$

or in matrix form, where $\succcurlyeq$ denotes componentwise inequalities [151, p. 32, 40]

$$\begin{bmatrix}0\\1\end{bmatrix} \succcurlyeq \begin{bmatrix}-1 & (d_{\max}-1)\\ 1 & \frac{1}{2}\left(1-\frac{d_{\min}}{d_{\max}}\right)\end{bmatrix}\begin{bmatrix}\alpha\\ \delta\end{bmatrix}$$

yields, after inversion, the bounds (5.15) and (5.16). □

D.5. INFLUENCE OF α AND δ ON THE EIGENVALUES β_k AND THE EIGENVECTOR y_2

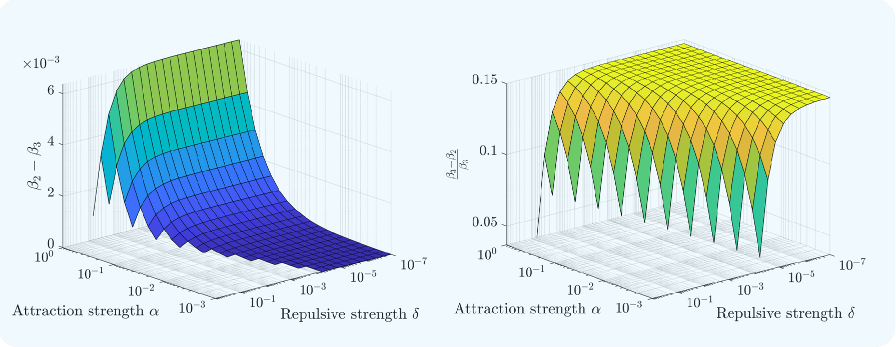


Figure D.1: Gap $\beta_2-\beta_3$ between the second and the third largest eigenvalue of the $N\times N$ matrix $W-\text{diag}(W\cdot u)$, for different values of the attractive α and repulsive δ strength (left-hand side figure). Relative difference $\frac{\beta_3-\beta_2}{\beta_3}$, for different values of the attractive α and repulsive δ strength (right-hand side figure). An SSBM network with $N = 1000$ nodes, $c = 5$ clusters is used for both plots, with $b_{in} = 25$ and $b_{out} = 2.5$.

Figure D.1 shows that influence of the attractive and repulsive strength α and δ on the eigenvalue gap $\beta_2 - \beta_3$ is relatively small if α and δ are not too small and obeying the bounds (5.15) and (5.16). While the difference increases when the attraction strength α is increasing, the repulsive strength δ has no visible influence on the eigenvalue gap.

The eigenvalue β_2 depends on the community structure of a graph. Figure D.2 reveals positive correlation between the eigenvalue β_2 and the modularity index m of a graph. As the modularity index increases, the eigenvalue β_2 approaches value 1. In the limit case, when there are only intra-community links in the network, $\beta_2 = 1$, indicating the eigenvector y_2 represents a steady state.

Figure D.3 reveals that the repulsive strength δ does not affect the eigenvector y_2 components significantly. Eigenvector y_2 components of nodes from the same cluster are better distinguished from the remaining components of y_2 for smaller values of repulsive strength δ.

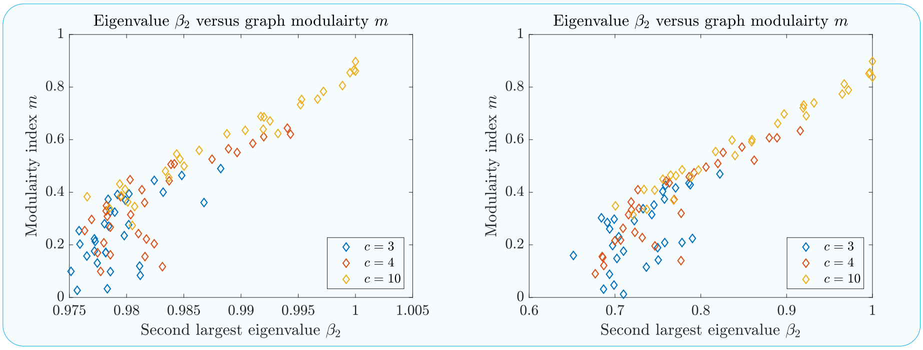


Figure D.2: The eigenvalue β_2 versus the modularity index m of an SSBM graph of $N = 999$ nodes and $c = 3$ clusters, and an SSBM graph of $N = 1000$ nodes, with $c = 4, 10$ clusters, respectively. The parameters b_{in} and b_{out} are varied, while keeping average degree $d_{av} = 7$ fixed. For each combination of b_{in} and b_{out}, the modularity index m and the eigenvalue β_2 are computed. The correlation is presented in case only interactions between direct neighbours are allowed (left-hand side figure) and in case interactions between each pair of nodes are allowed (right-hand side figure).

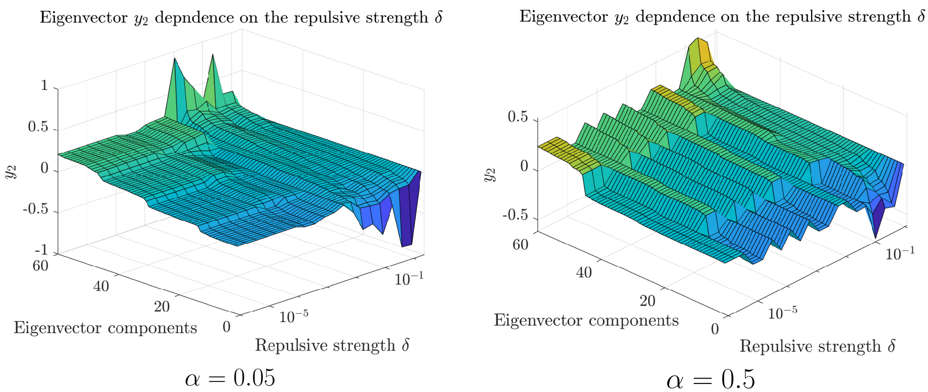


Figure D.3: Sorted eigenvector $\hat{y}_2$ components for different values of the repulsive strength δ, in case of a SSBM network of $N = 100$ nodes, $c = 4$ clusters and with parameters $b_{in} = 25$ and $b_{out} = 1$. The attraction rate equals $\alpha = 0.05$ (left-hand side figure) and $\alpha = 0.5$ (right-hand side figure), while the repulsive strength δ obeys bounds in (5.16).

D.6. COMPLEXITY OF LCP

The computational complexity of LCP consists of three parts: the computation of (i) the $N \times N$ matrix W in (5.10), (ii) the $N \times 1$ eigenvector y_2 of the matrix $W - \text{diag}(Wu)$ and (iii) the identification of the clusters based on the sorted eigenvector $\hat{y}_2$.

Algorithm 2 Computation of the $N \times N$ matrix $A \circ A^2$.

Require: A denotes the adjacency matrix, N denotes number of links, while the set of node i neighbours is denoted by $\mathcal{N}_i$.

$A_s \leftarrow O_{N\times N}$
for $i \leftarrow 1$ to N **do**
  for $j \leftarrow \mathcal{N}_i$ **do**
    for $m \leftarrow \left(\mathcal{N}_j \setminus \{1,2,\dots,i\}\right) \cap \mathcal{N}_i$ **do**
      $(A_s)_{i,m} \leftarrow (A_s)_{i,m} + 1$ ▷ Account for the 2-hop walk ($i \to j \to m$)
    end for
  end for
end for
$A_s \leftarrow A_s + A_s^T$
return A_s

D.6.1. COMPUTING THE $N \times N$ MATRIX W

The $N \times N$ matrix $A \circ A^2$ in (5.10) requires the highest computational effort. Generally, computing the square of a matrix involves $O(N^3)$ elementary operation, but the zero-one structure of the adjacency matrix significantly reduces the operations. We provide below an efficient algorithm for the computation of $A \circ A^2$, whose entries determine the number of 2-hop walks between any two direct neighbours in the network.

We initialize the $N \times N$ matrix $A \circ A^2$ with zeros and only compute elements above the main diagonal, because $A \circ A^2$ is symmetric. The algorithm identifies all 2-hop walks between any two direct neighbours and accordingly updates the matrix. Let us consider a node i with d_i neighbours, denoted as $\mathcal{N}_i$. For a neighbouring node $j \in \mathcal{N}_i$, we increment the elements $\left(A \circ A^2\right)_{im}$ by 1, where $m \in \left(\mathcal{N}_j \setminus \{1,2,\dots,i\}\right) \cap \mathcal{N}_i$, accounting for 2-hop walks $i \to j \to m$. By repeating the procedure for each node, we compute all the elements above the main diagonal. Finally, we sum the generated matrix with its transpose to obtain $A \circ A^2$. Since the algorithm 2 is based on incrementing the matrix entries per each 2-hop walk between direct neighbours, the number of operations equals the sum $s = u^T \cdot \left(A \circ A^2\right) \cdot u$ of all elements of $A \circ A^2$

$$s = \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \cdot \lambda_j^2 \cdot u^T \left(x_i \circ x_j\right) \cdot \left(x_i \circ x_j\right)^T u \tag{D.25}$$

The eigenvectors of the adjacency matrix A are orthogonal. Therefore $\left(x_i \circ x_j\right)^T \cdot u = x_i^T \cdot x_j = 0$ if $i \neq j$, otherwise it equals 1 and (D.25) further simplifies to

$$s = \sum_{i=1}^{N} \lambda_i^3, \tag{D.26}$$

which equals 6 times number of triangles in the network [51, p. 31], because a 2-hop walk between adjacent nodes i and j over a common neighbour m is equivalent to a triangle $i \to m \to j \to i$. The computational complexity of $A \circ A^2$ thus reduces to $O(d_{av} \cdot L)$, as presented in Figure D.4. For a given matrix $A \circ A^2$, the computational complexity of the $N \times N$ matrix W is $O(L)$, because (5.10) can be transformed into Hadamard product terms (i.e. element-based operations).

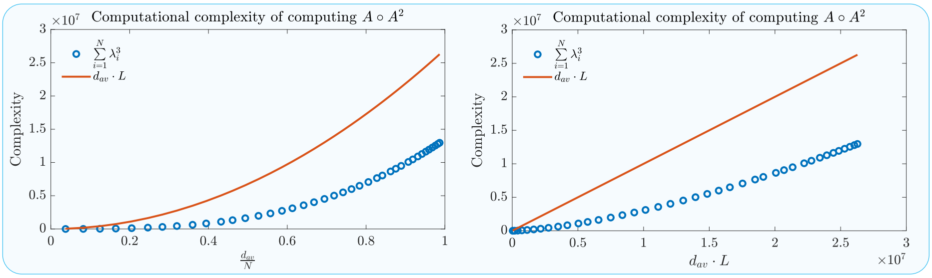


Figure D.4: Sum of the cubed eigenvalues λ of the adjacency matrix A (blue circles) and product of the average degree d_{av} and number of links L (red line), for an Erdős–Rényi random graph with $N = 300$ nodes, versus the relative mean degree $\frac{d_{av}}{N}$ (left-hand side figure) and $d_{av} \cdot L$ (right-hand side figure).

D

D.6.2. COMPUTING THE $N \times 1$ EIGENVECTOR y_2

The eigenvector y_2 corresponds to the second largest eigenvalue β_2 of the $N \times N$ matrix $W - \text{diag}(W \cdot u)$. The largest eigenvalue $\beta_1 = 1$ corresponds to the eigenvector $y_1 = \frac{1}{\sqrt{N}} u$. Computing the eigenvector y_2 is equivalent to computing the largest eigenvector of the matrix $W - \text{diag}(W \cdot u) - \frac{1}{N} \cdot u \cdot u^T$, which can be executed using the power method [51], for a given matrix W, with computational complexity $O(L)$.

D.6.3. COMPUTING THE CLUSTER MEMBERSHIP FUNCTION

We apply the recursive algorithm 1 to identify communities based on the $N \times 1$ eigenvector y_2. The number of iterations of the algorithm ideally equals $T = \log_2 c$, while in worst case scenario there are c iterations. Given a fixed number c of communities, the computational complexity within an iteration is $O(L)$, as shown in pseudocode 1. The number of clusters c may depend upon N and is in worst case equal to N. Thus, computational complexity increases in worst case to $O(N \cdot L)$.

D.6.4. SCALING THE INTER-COMMUNITY LINKS

Between two iterations of the linear clustering process, we identify inter-community links and scale their weights, as defined in (5.27). The computational complexity of this step is $O(L)$, as the ranking difference of neighbouring nodes is computer over each link.

Finally, computational complexity of the entire proposed clustering process equals $O(N \cdot L)$, because $d_{av} = O(N)$.

E

APPENDIX FOR CHAPTER 6

E.1. DATASETS DESCRIPTION

Since 1809, the nature and evolution of the Dutch society and economy were recorded in national statistics such as municipality-related population and surface measurements. From this data we selected three datasets which together describe the national year-on-year dynamics at the municipality level:

1 Population measurements of each municipality,

2 Digital geometries representing the area of each municipality,

3 Municipality merging.

Population measurements have been collected from two different sources. In period (1809 – 1960) the population data set is obtained from the Historical Database of Dutch Municipalities[1] (HDNG), collected by the International Institute of Social History, which is part of the Royal Netherlands Academy of Arts and Sciences. Further, the number of inhabitants per Dutch municipality in period (1960–2019) is obtained from the Statistics Netherlands [2] (CBS).

The other two datasets are collected from the online repositories of the CBS website. While digital geometries and the municipality merging datasets exist for each year in the period (1830 – 2019) consistent in time, population data sets cover in total more than two centuries, but with varying time resolution. A detailed overview of the availability of data sets over time is provided in Figure E.1.

E.2. CODING SCHEMES IDENTIFYING MUNICIPALITIES

In this research, two complementary coding schemes are used to identify municipalities and their geographic area, namely the four digit Central Bureau of Statistics code (CBS

[1] Historische Database Nederlandse Gemeenten
[2] Centraal Bureau voor de Statistiek

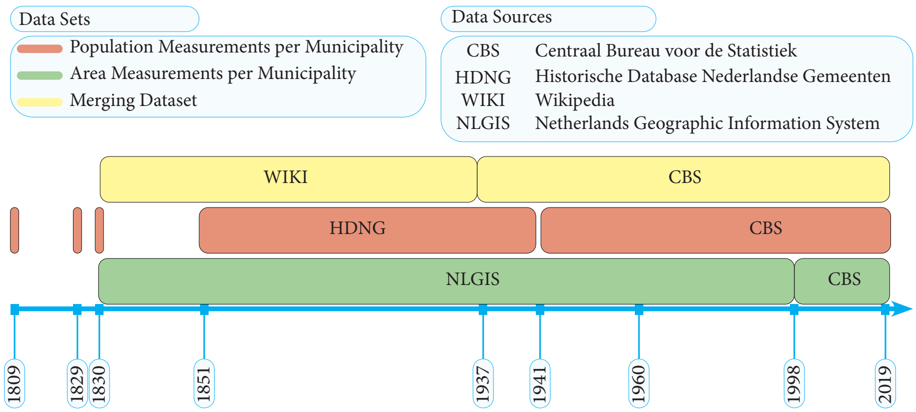


Figure E.1: Datasets Overview.

code) and the five digit Amsterdam code (AMS code). The CBS code identifies municipalities that existed since 1830, while the Amsterdam code can be traced back to Dutch municipalities that existed since 1812. The CBS code identifies specific administrative entities (municipality names), while the Amsterdam code identifies specific geographical areas on which municipalities are/were located.

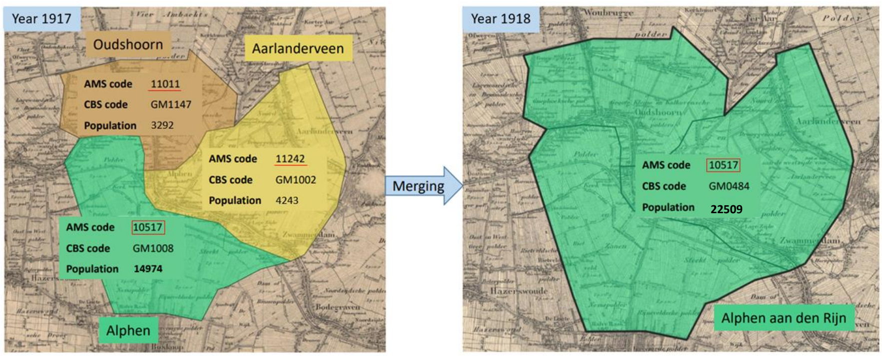


Figure E.2: Municipality merging example of Alphen aan den Rijn in 1918.

Whenever municipal restructuring leads to a municipality merger or a name change, a new CBS code is generated and assigned to the new municipality. However, the new municipality is assigned an existing Amsterdam code that belonged to one of the municipalities that were involved in the merging process, in order to ensure the historical continuity of the geographical area. As exemplified in Figure E.2, the municipality with the largest population passes on its Amsterdam code to the newly formed municipality. For example, when a municipality annexes an adjacent municipality, the Amsterdam code of the annexing municipality is preserved and the Amsterdam code of an annexed municipality is abolished, while all their CBS codes are abolished. The CBS code can be considered a unique identifier for a municipality, because it uniquely specifies a municipal entity that exists (or has existed) for a certain defined time period. The Amsterdam code, however, is not a unique municipality identifier; it has been designed in such a way that all Dutch municipalities possess an Amsterdam code that can be traced back to an Amsterdam code of a Dutch municipality which existed in 1812.

E.3. MUNICIPALITY MERGING

During the researched period (1830 – 2019) we analysed the process of municipality merging and municipality name changing. We distinguish five event types, of which each involves discontinuation of the CBS code of the municipalities involved. In case of a merging/renaming event, the CBS code is abolished (becomes inactive) at the end of year k, and the administrative change takes place at the beginning of the following year $k+1$. The five event types (A-E) are explained below:

- Type A (Annexation): the abolished municipality is annexed by an existing (usually adjacent) municipality at the end of year k. This process is officially called 'light merger' (in Dutch: lichte samenvoeging) and the CBS code of the abolished municipality becomes inactive in year k+1. This reclassification type has occurred 542 times in total during the studied time period (1830-2019).

- Type B (Border split): the area of the abolished municipality is split among an existing municipality and a newly formed municipality. This reclassification type is a combination of Type A and Type C, as both processes occur at the same time within the former municipality's boundaries. This reclassification type has occurred 10 times during the studied time period (1830-2019).

- Type C (Coalition): the abolished municipality, along with other neighboring municipalities which are abolished at the end of the same year k, form a coalition by creating a new municipality. The new municipality is assigned a new CBS code at the beginning of year k+1, and the CBS codes of the merger participants become inactive at the end of year k. This process is officially called 'regular merger' (in Dutch: Reguliere samenvoeging). This reclassification type has occurred 502 times during the studied time period (1830-2019).

- Type D (Dutch and/or Frisian Name-change): only the official name of a municipality is changed in Dutch or Frisian language, while its borders remain unchanged. The municipality is assigned a new CBS code at the beginning of year k+1, and the old CBS code of the municipality becomes inactive at the end of year k. A main difference between the Amsterdam and CBS coding schemes is that the municipality retains its Amsterdam code when undergoing a name-change. A municipality name-change has occurred 56 times during the studied time period (1830-2019).

- Type E (Exchanged internationally): the area of a municipality is exchanged between a neighboring country and the Netherlands. In case a municipality is allocated to a neighboring country, it is recorded in statistics to be no longer part of the Netherlands in year k+1. This reclassification type has occurred 2 times during the studied time period (1830-2019). The municipalities Tudderen (Drostambt) and Elten, both annexed after the Second World War by Germany in 1963.

E.4. MUNICIPALITY MERGING PROCESS DEMONSTRATED ON A PLANAR GRAPH

Although the municipality merging process introduced in Section 6.3.1 changed the topology significantly, the average degree of the DMN remained almost unchanged, $d_{av}[k] \approx 5$. Figure E.3 shows a planar graph with two examples of centrally positioned merging municipality nodes, each having a degree 5. On the right-hand side of the Figure E.3 the different of the two merger examples are given.

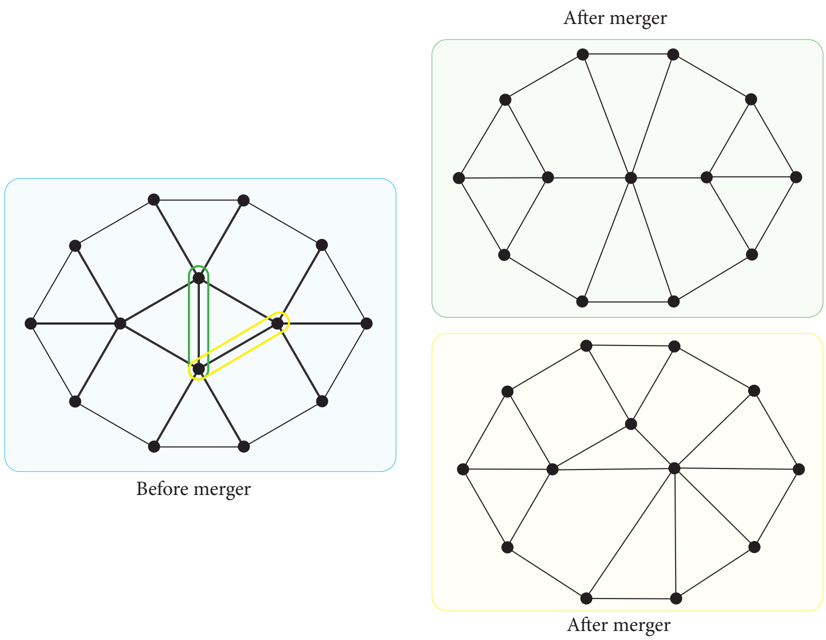


Figure E.3: Two merging process examples demonstrated on a planar graph.

Two adjacent nodes i and j can have either $|\mathcal{N}_i[k] \cap \mathcal{N}_j[k]| = 2$ or $|\mathcal{N}_i[k] \cap \mathcal{N}_j[k]| = 3$ common neighbours. When nodes i and j are merged into one node, the number of nodes $N[k+1]$ and number of links $L[k+1]$ in the next year $k+1$ change as follows

$$\begin{aligned} L[k+1] &= L[k] - |\mathcal{N}_i[k] \cap \mathcal{N}_j[k]| - 1 \\ N[k+1] &= N[k] - 1. \end{aligned}$$

We perform the merging of two adjacent municipalities and provide the updated topology on the right-hand side of Figure E.3. The conservation law on the average degree $d_{av}[k+1]$, when nodes i and j are merged at the end of the year k, can be obtained by importing $d_{av}[k] = 2\frac{L[k]}{N[k]}$ into the above equation

$$d_{av}[k+1] = \left(1 + \frac{1}{N[k]-1}\right) \cdot d_{av}[k] - 2\frac{|\mathcal{N}_i[k] \cap \mathcal{N}_j[k]| - 1}{N[k]-1}. \tag{E.1}$$

We further consider a case when three municipalities i, j and m are merged into one, at the end of year k. The newly formed municipality in the following year $k+1$ is connected to another municipality if either municipality i, j or m was connected to that municipality in year k. Therefore, the degree of the newly formed municipality equals $|\mathcal{N}_i \cup \mathcal{N}_j \cup \mathcal{N}_m|$. To determine the number of removed links $L[k+1] - L[k]$ in the DMN due to the merger, we apply the inclusion-exclusion formula (see for example [64, p.10]) and obtain

$$|\mathcal{N}_i \cup \mathcal{N}_j \cup \mathcal{N}_m| = |\mathcal{N}_i| + |\mathcal{N}_j| + |\mathcal{N}_m| - |\mathcal{N}_i \cap \mathcal{N}_j| - |\mathcal{N}_i \cap \mathcal{N}_m| - |\mathcal{N}_j \cap \mathcal{N}_m| + |\mathcal{N}_i \cap \mathcal{N}_j \cap \mathcal{N}_m|,$$

from where we derive the number of nodes $N[k+1]$ and the number of links $L[k+1]$ in the year $k+1$

$$\begin{aligned} L[k+1] &= L[k] - |\mathcal{N}_i \cap \mathcal{N}_j| - |\mathcal{N}_i \cap \mathcal{N}_m| - |\mathcal{N}_j \cap \mathcal{N}_m| - a_{ij}[k] - a_{im}[k] - a_{jm}[k] + |\mathcal{N}_i \cap \mathcal{N}_j \cap \mathcal{N}_m| \\ N[k+1] &= N[k] - 2, \end{aligned}$$

leading to the following conservation law of the average degree $d_{av}[k]$

$$\begin{aligned} d_{av}[k+1] = \left(1 + \frac{1}{N[k]-2}\right) \cdot d_{av}[k] - \\ 2\frac{|\mathcal{N}_i \cap \mathcal{N}_j| + |\mathcal{N}_i \cap \mathcal{N}_m| + |\mathcal{N}_j \cap \mathcal{N}_m| + a_{ij}[k] + a_{im}[k] + a_{jm}[k] - |\mathcal{N}_i \cap \mathcal{N}_j \cap \mathcal{N}_m|}{N[k]-2}. \end{aligned} \tag{E.2}$$

In case three municipalities merge into one municipality, the average degree $d_{av}[k]$ slightly decreases in time, which is the opposite effect of when two municipalities merge. During the period 1960 – 2000, mergers involving more than two municipalities were common, causing a decreasing trend in the average degree $d_{av}[k]$, as visible in the lower part of Figure 6.3.

E.5. DIFFERENT DISTRIBUTIONS

E.5.1. NORMAL DISTRIBUTION

A Gaussian random variable $X = N(\mu, \sigma^2)$ is a continuous random variable with an extent over the entire real axis and is defined [64] by the distribution function $F_X(x) = \Pr[X \le x]$ as

$$F_X(x) = \frac{1}{\sigma \cdot \sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{(t-\mu)^2}{2\sigma^2}\right) dt, \tag{E.3}$$

with the mean $E[X] = \mu$ and with the variance $\text{Var}[X] = \sigma^2$. The corresponding probability density function $f_X(x) = \frac{d}{dx}\Pr[X \le x]$ is

$$f_X(x) = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sigma \cdot \sqrt{2\pi}}. \tag{E.4}$$

E.5.2. LOGNORMAL DISTRIBUTION

A lognormal random variable is defined as $Y = e^X$, where $X = N(\mu, \sigma^2)$ is a Gaussian or normal random variable [64]. The distribution function $F_Y(y) = \Pr[Y \le y] = \Pr[X \le \log y]$ follows from (E.3), for non-negative real values of y, as

$$F_Y(y) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\log y} \exp\left[-\frac{(t-\mu)^2}{2\sigma^2}\right] dt, \tag{E.5}$$

The corresponding probability density function $f_Y(y) = \frac{dF_Y(y)}{dy}$ of a lognormal random variable Y follows by differentiation of (E.5) as

$$f_Y(y) = \frac{\exp\left[-\frac{(\log y-\mu)^2}{2\sigma^2}\right]}{\sigma \cdot \sqrt{2\pi} \cdot y}, \tag{E.6}$$

The mean $E[Y]$ and the variance $\text{Var}(Y)$ can be computed [64, Sec. 3.5.5] as

$$\begin{aligned} E[Y] &= e^{\left(\mu + \frac{\sigma^2}{2}\right)} \\ \text{Var}[Y] &= \left(e^{\sigma^2} - 1\right) \cdot e^{(2\mu+\sigma^2)}. \end{aligned} \tag{E.7}$$

E.5.3. LOGISTIC DISTRIBUTION

A logistic random variable X, also known as a Fermi-Dirac random variable [64, Sec. 19.6.2], has the distribution function

$$F_X(x) = \frac{1}{1+e^{-\frac{x-\mu}{s}}} = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{x-\mu}{2s}\right), \tag{E.8}$$

The probability density function (pdf) of a logistic random variable X again follows by differentiation of (E.8) as

$$f_X(x) = \frac{1}{4s}\text{sech}^2\left(\frac{x-\mu}{2s}\right). \tag{E.9}$$

It is more convenient to consider the normalized Fermi-Dirac random variable $Z = \frac{X-\mu}{s}$ that obeys[3] $F_Z(z) = \Pr[Z \le z] = \frac{1}{1+e^{-z}} = 1 - \frac{1}{1+e^{z}}$. The probability generating function (pgf) $\phi(w) = E\left[e^{-wZ}\right] = \int_{-\infty}^{\infty} e^{-wt} f_Z(t)dt$ is the double-sided Laplace transform [64, p. 20] of $f_Z(t)$ and equals

$$\varphi(w) = \int_{-\infty}^{\infty} e^{-wt} \frac{d}{dt}\frac{1}{1+e^{-t}} dt$$

[3]Indeed, after letting $z = -\frac{x-\mu}{s}$ in (E.8), we have $\frac{1}{1+e^{-z}} = \Pr[X \le \mu + sz] = \Pr[\frac{X-\mu}{s} \le z]$.

Partial integration leads to

$$\varphi(w) = \frac{e^{-wt}}{1+e^{-t}}\Big|_{-\infty}^{\infty} + w\int_{-\infty}^{\infty} \frac{e^{-wt}}{1+e^{-t}}dt$$

and the first term vanishes, provided that $0 < \text{Re}(w) < 1$ holds, with $\text{Re}(w)$ denoting the real part of w. Let $u = e^{-t}$ and $t = -\log u$, then

$$\frac{\varphi(w)}{w} = \int_0^{\infty} \frac{u^{w-1}}{1+u}du$$

One of the Beta function integrals, $B(x,y) = \int_0^{\infty} \frac{t^{x-1}}{(1+t)^{x+y}}dt = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$, valid for $\text{Re}(x) > 0$ and $\text{Re}(y) > 0$, shows that, for $0 < \text{Re}(w) < 1$,

$$\varphi(w) = \Gamma(w)\Gamma(1-w) = \frac{\pi w}{\sin \pi w}$$

where the last equality is the reflection formula of the Gamma function $\Gamma(w)$, valid for all complex numbers w. The pgf $\varphi(w) = E\left[e^{-wZ}\right] = \sum_{n=0}^{\infty} \frac{(-1)^n}{n!} E[Z^n]\, w^n$ contains all moments, whereas the Taylor series of $\frac{\pi w}{\sin \pi w}$ around $w = 0$ equals

$$\frac{\pi w}{\sin \pi w} = 1 + \sum_{n=1}^{\infty} \left(2^{1-2n} - 1\right)(2\pi)^{2n} B_{2n} \frac{(-1)^n w^{2n}}{(2n)!}$$

where B_n is the n-th Bernoulli number. By equating the corresponding powers of w in $\frac{\pi w}{\sin \pi w} = E\left[e^{-wZ}\right]$, we find all even moments, for $n > 0$

$$E\left[Z^{2n}\right] = \left(2^{1-2n} - 1\right)(2\pi)^{2n}(-1)^n B_{2n}$$

and while all odd $E\left[Z^{2n+1}\right] = 0$ for $n \geq 0$. Since $Z = \frac{X-\mu}{s}$ is a normalized random variable, the mean $E[Z] = 0$ and thus the mean $E[X] = \mu$. The variance $\text{Var}[Z] = E\left[(Z - E[Z])^2\right] = E\left[Z^2\right] = \frac{\pi^2}{3}$, because $B_2 = \frac{1}{6}$. Hence,

$$E\left[\left(\frac{X-\mu}{s}\right)^2\right] = \frac{1}{s^2}\text{Var}[X] = \frac{\pi^2}{3}$$

resulting in $\text{Var}[X] = \sigma^2 = \frac{\pi^2 s^2}{3}$.

E.5.4. LOG-LOGISTIC DISTRIBUTION

A log-logistic random variable, defined by $Y = e^X$ where X is a logistic random variable with mean μ and variance σ, has the probability function

$$F_Y(y) = \frac{1}{1+e^{-\frac{\log(y)-\mu}{s}}} = \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{\log(y)-\mu}{2s}\right), \tag{E.10}$$

with the corresponding probability density function

$$f_Y(y) = \frac{1}{4s\cdot y}\text{sech}^2\left(\frac{\log y - \mu}{2s}\right). \tag{E.11}$$

We concentrate first on the normalized log-logistic random variable. The probability distribution of a log-logistic random variable $Y = e^X$, which is always positive, is

$$\Pr[Y \le y] = \Pr[e^X \le y] = \Pr[X \le \log y] = \Pr\left[\frac{X-\mu}{s} \le \frac{\log y - \mu}{s}\right]$$

which, in terms of the normalized logistic random variable Z with $\log z = \frac{\log y - \mu}{s}$ (and thus $z = \left(ye^{-\mu}\right)^{\frac{1}{s}}$ and $z > 0$) is

$$\Pr[Z \le \log z] = \frac{1}{1+e^{-\log z}} = \frac{z}{z+1} = 1 - \frac{1}{z+1}$$

and the pdf is

$$f_Z(z) = \frac{1}{(z+1)^2}$$

The moments

$$E\left[Z^k\right] = \int_0^\infty \frac{t^k}{(t+1)^2} dt = \Gamma(k+1)\Gamma(1-k) = \frac{\pi k}{\sin \pi k}$$

do not exist for integers $k \ge 1$.

With a little more effort, we compute the moments directly for a log-logistic random variable Y,

$$E\left[Y^k\right] = \frac{1}{4s}\int_0^\infty \frac{t^{k-1}}{\cosh^2\left(\frac{\log t - \mu}{s}\right)} dt$$

We modify this integral by a series of substitutions. First, let $u = \log t$, then

$$E\left[Y^k\right] = \frac{1}{4s}\int_{-\infty}^\infty \frac{e^{ku}}{\cosh^2\left(\frac{u-\mu}{s}\right)} du$$

followed by the substitution $w = \frac{u-\mu}{s}$ yields

$$E\left[Y^k\right] = \frac{e^{k\mu}}{4}\int_{-\infty}^\infty \frac{e^{ksw}}{\cosh^2(w)} dw$$

illustrating that the integral exists provided $-1 < \frac{ks}{2} < 1$, thus, the integer $k < \frac{2}{s}$. Next, let $p = e^w$, then

$$E\left[Y^k\right] = e^{k\mu}\int_0^\infty \frac{p^{ks+1}}{\left(p^2+1\right)^2} dp$$

A last substitution $t = p^2$ reveals again the above Beta function integral

$$\begin{aligned} E\left[Y^k\right] &= \frac{1}{2}e^{k\mu}\int_0^\infty \frac{t^{\frac{ks}{2}}}{(t+1)^2} dp \\ &= \frac{1}{2}e^{k\mu}\Gamma\left(\frac{ks}{2}+1\right)\Gamma\left(1-\frac{ks}{2}\right) = \frac{1}{2}e^{k\mu}\frac{\pi\frac{ks}{2}}{\sin\pi\frac{ks}{2}} \end{aligned}$$

Hence, provided that $-\frac{2}{s} < \text{Re}\,\alpha < \frac{2}{s}$ where α is now a complex number, the $\alpha-$moments of the log-logistic random variable exists

$$E\left[Y^{\alpha}\right] = \frac{1}{2} e^{\alpha\mu} \frac{\frac{\pi\alpha s}{2}}{\sin\frac{\pi\alpha s}{2}}.$$

The mean E[Y] and the variance E[Y^2] $-$ (E[Y])^2 are

$$\begin{cases} E[Y] = \frac{1}{4} \cdot e^{\mu} \cdot \frac{\pi s}{\sin(\frac{\pi s}{2})} & \text{If } s < 2 \\ \text{Var}[Y] = \frac{1}{4} \cdot e^{2\mu} \cdot \left(\frac{2\pi s}{\sin(\pi s)} - \frac{1}{4} \cdot \frac{\pi^2 s^2}{\sin^2(\frac{\pi s}{2})} \right) & \text{If } s < 1. \end{cases} \tag{E.12}$$

E.5.5. TAIL DISTRIBUTIONS

The probability density function $f_{Z[k]}(z)$ of the logistic distribution model in (E.9) can be transformed as follows

$$f_{Z[k]}(z) = \frac{1}{4s} \cdot \frac{e^{\frac{z-\mu}{s}}}{\left(e^{\frac{z-\mu}{s}} + 1\right)^2}. \tag{E.13}$$

We introduce the logarithm of the probability density function as $L_l(z) = \log(f_{Z[k]}(z))$ and obtain from (E.13)

$$L_l(z) = -\log(4s) + \frac{z-\mu}{s} - 2\log\left(e^{\frac{z-\mu}{s}} + 1\right).$$

Since we are interested in tail distribution, it holds $e^{\frac{z-\mu}{s}} >> 1$, allowing us to introduce the approximation $e^{\frac{z-\mu}{s}} + 1 \approx e^{\frac{z-\mu}{s}}$, simplifying the above equation as follows

$$L_l(z) = -\log(4s) - \frac{z-\mu}{s}.$$

Finally, by importing $z = \log p$, we obtain

$$\log(f_{Z[k]}(z)) = \frac{\mu}{s} - \log(4s) - \frac{1}{s}\log(p), \tag{E.14}$$

informing us that the probability density function $f_{Z[k]}(z)$ of the logistic distribution model in (E.9), decays linearly on a double logarithmic scale, for $z >> \mu$. Therefore, the subset of the largest municipalities in population follows a power-law distribution, as discussed in Section 6.2.3.

Further, we define the logarithm of the probability density function $L_n(z) = \log(f_{Z[k]}(z))$ of a normal distribution in (E.4) and obtain

$$L_n(z) = -\log\left(\sigma\sqrt{2\pi}\right) - \frac{(z-\mu)^2}{2\pi^2}.$$

The above equation further transforms after importing $z = \log p$

$$\log(f_{Z[k]}(z)) = -\log\left(\sigma\sqrt{2\pi}\right) - \frac{(\log p - \mu)^2}{2\pi^2}, \tag{E.15}$$

teaching us that the probability density function $f_{Z[k]}(z)$ of the normal distribution in (E.4) decreases on a double logarithmic scale as a square function of the population p. Tail distribution in (E.15) better fits the area per Dutch municipality, given the constraint that the sum of area per municipality $\sum_{i=1}^{N[k]} s_i[k]$ remains relatively constant in time.

E

E.6. GOODNESS-OF-FIT TESTS

The goodness-of-fit tests Anderson-Darling (AD) and Kolmogorov-Smirnov (KS) [152, Ch. 14] are used, as shown in Figure E.4, to determine the plausibility of the hypothesis that the logarithm of the area per Dutch municipality follows either a normal or a logistic distribution. Both tests provide a p value that answers how likely the hypothesis holds. These tests are based on measuring the difference (distance) between the hypothesized and measured distributions. Artificial datasets are created using the same model, and the distance is computed. Finally, the p value represents the ratio of the artificial distance larger than the measured distance from the empirical data. If the computed p value is close to 0, the measured data does not agree with the hypothesized distribution, whereas p close to 1 confirms the hypothesis.

E.6.1. AREA DISTRIBUTION

Both the AD and KS tests indicate that, from 1830 until 1918, the logarithm of the area of a typical Dutch municipality follows a logistic distribution rather than a normal distribution, while from 1918 until 1990, the opposite holds. The AD and the KS test do not favour any distribution consistently during the last three decades.

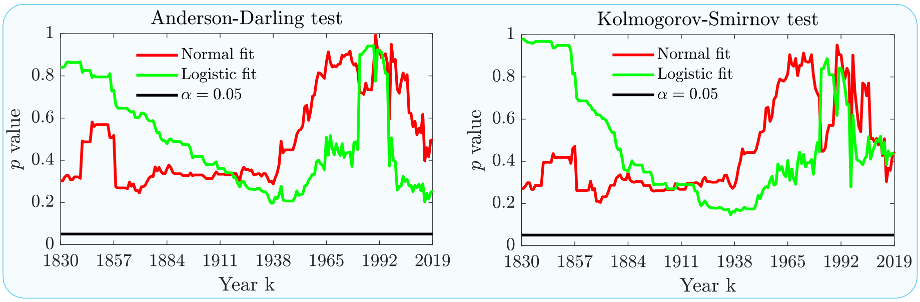


Figure E.4: Anderson-Darling test (left-hand side) and Kolmogorov-Smirnov test (right-hand side) results of the normal distribution fit (red colour) and a logistic distribution fit (green colour) of the logarithm of the area Y distribution in the period 1830 – 2019.

E.6.2. POPULATION DISTRIBUTION

The p value of the AD and KS goodness-of-fit tests is shown for both the normal and logistic distribution of the logarithmic of the population in Figure E.5. Over the entire period (1809 – 2019), the p value of the AD and the KS goodness-of-fit tests indicates that the logarithm of population $Z[k]$ per municipality follows the logistic distribution more closely than the normal distribution.

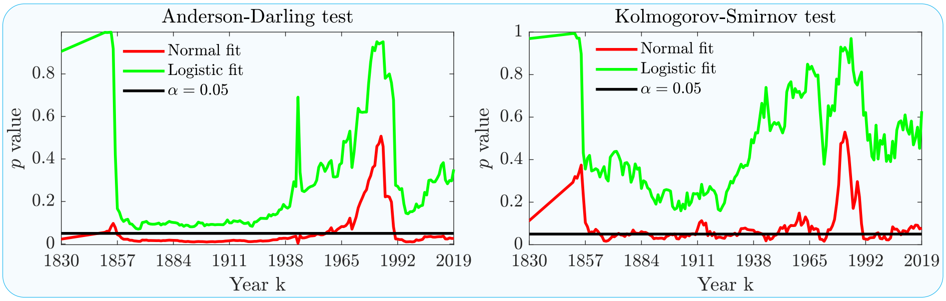


Figure E.5: The p value of the Anderson-Darling test (left-hand side) and Kolmogorov-Smirnov test (right-hand side) for the hypothesis that the logarithm of population vector $Z[k]$ follows the normal (red colour) and logistic (green colour) distribution model in the period 1809 – 2019.

E.7. CONSERVATION LAWS OF POPULATION AND AREA

E.7.1. AREA EVOLUTION

In this section, we analyse a merger case when $N_a[k] = |\mathscr{A}[k]|$ municipalities are abolished at the end of year k and annexed by an existing municipality $\eta \in \mathscr{N}[k]$. The mean $y_{av}[k]$ of the $N[k] \times 1$ logarithm of area vector $y[k]$ evolves after the merger as follows

$$y_{av}[k] = \frac{1}{N[k]} \sum_{i \in \mathscr{N}[k]} y_i[k] = \frac{1}{N[k]} \log\left(\prod_{i \in \mathscr{N}[k]} s_i[k] \right), \tag{E.16}$$

while in the next year $k+1$, after abolishing $N_a[k]$ municipalities into a single new municipality η with area $s_\eta[k+1] = s_\eta[k] + \sum_{j \in \mathscr{A}[k]} s_j[k]$ in year $k+1$, we obtain

$$y_{av}[k+1] = \frac{1}{N[k]-N_a[k]} \log\left(\prod_{j \in \mathscr{N}[k] \setminus (\eta \cup \mathscr{A}[k])} s_j[k] \right) + \frac{1}{N[k]-N_a[k]} \log\left(\sum_{i \in \eta \cup \mathscr{A}[k]} s_i[k] \right).$$

By adding and subtracting the term $\frac{1}{N[k]-N_a[k]} \sum_{j \in \eta \cup \mathscr{A}[k]} \log(s_j[k])$ from the above relation, we obtain

$$y_{av}[k+1] = \frac{N[k]}{N[k]-N_a[k]} y_{av}[k] - \frac{1}{N[k]-N_a[k]} \sum_{j \in \eta \cup \mathscr{A}[k]} \log(s_j[k]) + \frac{1}{N[k]-N_a[k]} \log\left(\sum_{i \in \eta \cup \mathscr{A}[k]} s_i[k] \right),$$

from where the governing equation for the average of logarithm of the area vector $y_{av}[k]$ is as follows

$$y_{av}[k+1] = y_{av}[k] + \frac{N_a[k]}{N[k]-N_a[k]} \cdot y_{av}[k] + \frac{1}{N[k]-N_a[k]} \log\left(\frac{\sum_{i \in \eta \cup \mathscr{A}[k]} s_i[k]}{\prod_{j \in \eta \cup \mathscr{A}[k]} s_j[k]} \right). \tag{E.17}$$

E.7.2. POPULATION EVOLUTION

Based on the revealed rank size distribution of population per municipality, presented in Figure 6.14, municipality i population in year k can be approximated as

$$z_i[k] \approx -\beta[k] \cdot \log r_i[k] + z_A[k], \tag{E.18}$$

where $z_A[k] = \log(x_A[k])$ denotes the logarithm of Amsterdam population in year k. By assuming that the ranking of a municipality i does not change $r_i[k+1] = r_i[k]$ between years k and $k+1$, we obtain

$$z_i[k+1] = -\beta[k+1] \cdot \log r_i[k] + \log\left(p_A[k+1]\right). \tag{E.19}$$

By combining (6.14) and (E.19), the difference $z_i[k+1] - z_i[k]$ in logarithm of the population of the i-th municipality in two consecutive years k and $k+1$ follows

$$z_i[k+1] - z_i[k] = -\left(\beta[k+1] - \beta[k]\right) \cdot \log r_i[k] + (z_A[k+1] - z_A[k]).$$

E

After importing (6.12), the relation above translates into

$$\log\left(\frac{p_i[k+1]}{p_i[k]}\right) = -b_1 \cdot \log r_i[k] + \left(\log\left(p_A[k+1]\right) - \log\left(p_A[k]\right)\right).$$

Finally, we obtain the population increase of municipality i in year k

$$\frac{p_i[k+1]}{p_i[k]} = (r_i[k])^{-b_1} \cdot \frac{p_A[k+1]}{p_A[k]}.$$

By assuming that each municipality follows the rank-size distribution in the equation above, we derive the mean $z_{av}[k]$ as follows

$$z_{av}[k] = \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} \left(-\beta[k] \cdot \log r_i[k] + z_A[k]\right) = -\beta[k] \cdot \log\left(N[k]!^{\frac{1}{N[k]}}\right) + z_A[k]. \tag{E.20}$$

After importing (E.18) and (E.20) into the definition of the variance $\mathrm{Var}(z[k])$, we obtain

$$\mathrm{Var}(z[k]) = \frac{1}{N[k]} \cdot \sum_{i=1}^{N[k]} \left(z_A[k] + \beta[k] \cdot \log\left(N[k]!^{\frac{1}{N[k]}}\right) - \beta[k] \cdot \log r_i[k] - z_A[k]\right)^2.$$

that further simplifies as follows

$$\mathrm{Var}(z[k]) = \beta^2[k] \cdot g(N[k]), \tag{E.21}$$

where

$$g(N[k]) = \frac{1}{N[k]} \sum_{i=1}^{N[k]} \left(\log \frac{N[k]!^{\frac{1}{N[k]}}}{r_i}\right)^2 = \frac{1}{N[k]} \sum_{i=1}^{N[k]} \left(\log \frac{N[k]!^{\frac{1}{N[k]}}}{i}\right)^2,$$

when the sum terms are ordered in descending order. Equation (E.21) teaches us that, for a given $N[k]$, the variance $\mathrm{Var}(z[k])$ is a square function of the rank-size distribution slope $\beta[k]$.

E.7.3. RANK-SIZE DISTRIBUTION VERSUS POWER-LAW DISTRIBUTION

Under the assumptions introduced in Section 6.3.3, we derive the following probability distribution function

$$\Pr(X[k] > x_i) = \frac{r_i}{N[k]}. \tag{E.22}$$

From (6.14), we obtain

$$r_i[k] = \left(\frac{x_i[k]}{x_A[k]}\right)^{-\frac{1}{\beta[k]}},$$

transforming further relation (E.22)

$$\Pr(X[k] > x_i) = \frac{1}{N[k]} \cdot \left(\frac{x_i[k]}{x_A[k]}\right)^{-\frac{1}{\beta[k]}}.$$

On the other side, probability distribution function of the power-law distribution is

$$\Pr(X[k] > x_i) = x_i^{-(\tau[k]-1)}.$$

By comparing the last relation with (E.22), we obtain

$$\frac{1}{\beta[k]} \approx \tau[k] - 1,$$

leading to the dependece between the rank-size distribution slope $\beta[k]$ and the exponent of the power-law distribution $\tau[k]$

$$\beta[k] = \frac{1}{\tau[k] - 1}.$$

E.8. PROPERTIES OF THE MIGRATION MODEL

Theorem 23 *The proposed migration model (6.24) does not change the total population over time, but internally redistributes the population among neighbouring municipalities:*

$$T[k] = T[0], k > 0 \tag{E.23}$$

Proof Total population in year k is denoted as $T[k] = u^T \cdot p[k]$. The total population $T[k+1]$ in the following $k+1$ is as follows:

$$T[k+1] = u^T \cdot p[k+1].$$

By implementing (6.24) we obtain:

$$T[k+1] = u^T \cdot \left(I + \alpha \cdot M^T[k] + \delta \cdot M[k] - \delta \cdot \operatorname{diag}\left(M^T[k] \cdot u\right) - \alpha \cdot \operatorname{diag}(M[k] \cdot u)\right) \cdot p[k].$$

We further group terms of the previous equation

$$\begin{aligned} T[k+1] = T[k] \\ &+ \left(\delta \cdot \left(M^T \cdot u\right)^T - \delta \cdot \left(M^T \cdot u\right)^T\right) \cdot p[k] \\ &+ \left(\alpha \cdot (M \cdot u)^T - \alpha \cdot (M \cdot u)^T\right) \cdot p[k], \end{aligned} \tag{E.24}$$

from where we conclude $T[k+1] = T[k]$ or $T[k] = T[0]$, which completes the proof. □

E

Theorem 24 *The proposed migration model, defined in (6.24), is in a steady state if the following condition holds for each node $i \in \mathcal{N}[k]$:*

$$p_i[k] = \frac{\sum_{j\in\mathcal{N}_i^+[k]} \delta \cdot p_j[k] + \sum_{m\in\mathcal{N}_i^-[k]} \alpha \cdot p_m[k]}{\alpha \cdot d_i^+ + \delta \cdot d_i^-} \tag{E.25}$$

where $\mathcal{N}_i^+[k] = \{j \mid j \in \mathcal{N}_i[k]\, p_i[k] > p_j[k]\}$ defines a set of neighbours of node i, that have a larger population, while the set of smaller neighbours is given by $\mathcal{N}_i^-[k] = \{j \mid j \in \mathcal{N}_i[k],\, p_i[k] < p_j[k]\}$.

Proof We transform the governing equation (6.24) of the migration model into a node-level governing equation:

$$\begin{aligned} p_i[k+1] = p_i[k] & \\ &+ \sum_{j\in\mathcal{N}_i^+[k]} \delta \cdot p_j[k] + \sum_{m\in\mathcal{N}_i^-[k]} \alpha \cdot p_m[k] \\ &- \sum_{j\in\mathcal{N}_i^+[k]} \alpha \cdot p_i[k] - \sum_{m\in\mathcal{N}_i^-[k]} \delta \cdot p_i[k] \end{aligned} \tag{E.26}$$

We implement the steady state equality $p_i[k+1] = p_i[k]$ into (E.26) and obtain:

$$\left(\sum_{j\in\mathcal{N}_i^+[k]} \alpha + \sum_{m\in\mathcal{N}_i^-[k]} \delta \right) \cdot p_i[k] = \sum_{j\in\mathcal{N}_i^+[k]} \delta \cdot p_j[k] + \sum_{m\in\mathcal{N}_i^-[k]} \alpha \cdot p_m[k], \tag{E.27}$$

from where we conclude

$$p_i[k] = \frac{\sum_{j\in\mathcal{N}_i^+[k]} \delta \cdot p_j[k] + \sum_{m\in\mathcal{N}_i^-[k]} \alpha \cdot p_m[k]}{\alpha \cdot d_i^+ + \delta \cdot d_i^-}, \tag{E.28}$$

which completes the proof. □

E.9. LIST OF NOTATIONS

Four tables with the list of used notations in the paper are provided below.

Table E.1: Notations used in the paper for the Dutch Municipality Network

Notation	Explanation
k	Year k
$N[k]$	Number of active municipalities in year k
$N_a[k]$	Number of abolished municipalities in year k
$N_n[k]$	Number of newly established municipalities in year k
$\mathcal{N}[k]$	Set of active municipalities in year k
$\mathcal{A}[k]$	Set of abolished municipalities at the end of year k
$\mathcal{N}_i[k]$	Set of municipality i neighbouring municipalities in year k
$\mathcal{N}_i^+[k]$	Set of municipality i neighbours with larger population in year k
$\mathcal{N}_i^-[k]$	Set of municipality i neighbours with smaller population in year k
$d_i[k]$	Degree of node i in year k
$d_i^+[k]$	Number of municipality i neighbours with larger population in year k
$d_i^-[k]$	Number of municipality i neighbours with smaller population in year k
$d_{av}[k]$	Average degree of the DMN in year k
$L[k]$	Number of links in the DMN in year k
$\mathcal{L}[k]$	Set of links in the DMN in year k
$A[k]$	Adjacency matrix of the DMN in year k
$a_{ij}[k]$	ij-th element of the adjacency matrix $A[k]$ in year k

Table E.2: Notations used for the analysis of population dynamics

Notation	Explanation
$p_i[k]$	Population of municipality i in year k
$p_A[k]$	Population of Amsterdam in year k
$z_i[k]$	Logarithm of the population size of municipality i in year k
$z_A[k]$	Logarithm of the population size of municipality Amsterdam in year k
$p[k]$	The $N[k] \times 1$ vector of the population per municipality in year k
$z[k]$	The $N[k] \times 1$ vector of logarithm of the population size per municipality in year k
$p_{av}[k]$	Average population per municipality in year k
$z_{av}[k]$	Average logarithm of the population per municipality in year k
$P[k]$	Population random variable in year k
$Z[k]$	Logarithm of the population random variable in year k
$T[k]$	Total population of The Netherlands in year k
$M[k]$	The $N[k] \times N[k]$ migration matrix of the DMN in year k
$m_{ij}[k]$	ij-th element of the migration matrix $M[k]$ in year k
$\alpha[k]$	Forward migration rate in year k
$\delta[k]$	Backward migration rate in year k
$c_1[k]$	Estimated slope of the population increase in year k
$c_2[k]$	Estimated additive constant of the population increase in year k

E

Table E.3: Notations used for the area and the merging process

Notation	Explanation
$s_i[k]$	Area size of municipality i in year k
$y_i[k]$	Logarithm of the area size of municipality i in year k
$s[k]$	The $N[k] \times 1$ vector of area size per municipality in year k
$y[k]$	The $N[k] \times 1$ vector of logarithm of the area size per municipality in year k
$s_{av}[k]$	Average area size per municipality in year k
$y_{av}[k]$	Average logarithm of the area size per municipality in year k
$S[k]$	Area random variable in year k
$Y[k]$	Logarithm of the area random variable in year k
$x_i[k]$	Abolishment Likelihood index of municipality i in year k
$x[k]$	The $N[k] \times 1$ vector with an Abolishment Likelihood index per municipality in year k

E

Table E.4: Notations used for distribution functions

Notation	Explanation
$\mu_n[k]$	Shape parameter of the normal distribution in year k
$\sigma_n[k]$	Scale parameter of the normal distribution in year k
$\mu_l[k]$	Shape parameter of the logistic distribution in year k
$\sigma_l[k]$	Scale parameter of the logistic distribution in year k
$\beta[k]$	Population rank-size distribution slope in year k
$b_1[k]$	First parameter of a linear fit of $\beta[k]$
$b_2[k]$	Second parameter of a linear fit of $\beta[k]$
$\tau[k]$	Exponent of the power-law distribution in year k
$C[k]$	Normalisation constant of the power-law distribution in year k
$E[P]$	Expectation of the random variable P
$\mathrm{Var}(P)$	Variance of the random variable P
$\mathrm{Cov}(P)$	Covariance of the random variable P
p	p value of a goodness-of-fit test

F

Appendix for Chapter 7

F.1. List of Notations

Table F.1: Notations for the graph G and DLSS models

Notation	Explanation
G	Graph
$\mathcal{N}$	Set of N nodes of graph G
$\mathcal{L}$	Set of L links of graph G
N	Number of nodes in graph G
L	Number of links in graph G
W	Adjacency matrix of graph G
A_i	State matrix of a DLSS model of node/system i
B_i	Input matrix of a DLSS model of node/system i
C_i	Output matrix of a DLSS model of node/system i
D_i	Feedforward matrix of a DLSS model of node/system i
A_d	Diagonal block matrix composed of A_i matrices, $i \in \mathcal{N}$
B_d	Diagonal block matrix composed of B_i matrices, $i \in \mathcal{N}$
C_d	Diagonal block matrix composed of C_i matrices, $i \in \mathcal{N}$
D_d	Diagonal block matrix composed of D_i matrices, $i \in \mathcal{N}$
A_e	State matrix of a DLSS model of the network
B_e	Input matrix of a DLSS model of the network
C_e	Output matrix of a DLSS model of the network
D_e	Feedforward matrix of a DLSS model of the network

Table F.2: Notations for the links in G_e

Notation	Explanation
l_w	Vector with number of internal links connected to each internal node in G_e
l_ϕ	Vector with number of input links connected to each internal node in G_e
l_ψ	Vector with number of output links connected to each output node in G_e
l_z	Vector with number of external links connected to each output node in G_e
L_w	Total number of internal links in G_e
L_ϕ	Total number of input links in G_e
L_ψ	Total number of output links in G_e
L_z	Total number of external links in G_e
s_w	Vector with number of components of each internal link in G_e
s_ϕ	Vector with number of components of each input link in G_e
s_ψ	Vector with number of components of each output link in G_e
s_z	Vector with number of components of each external link in G_e
S_w	Total number of components of all internal links in G_e
S_ϕ	Total number of components of all input links in G_e
S_ψ	Total number of components of all output links in G_e
S_z	Total number of components of all external links in G_e

F

F.2. ELABORATION OF DEFINITION 20

We recall the definition of the matrix Γ:

$$\Gamma = \begin{bmatrix} \left(\Gamma_\phi\right)_{(L_w+L_\phi)\times r} & \left(\Gamma_w\right)_{(L_w+L_\phi)\times N} & O_{(L_w+L_\phi)\times q} \\ \left(\Gamma_z\right)_{(L_\psi+L_z)\times r} & \left(\Gamma_\psi\right)_{(L_\psi+L_z)\times N} & O_{(L_\psi+L_z)\times q} \end{bmatrix}$$

Matrix Γ preserves information of the source node of each link in G_e. Each row of the matrix Γ contains exactly one non-zero element and this element is equal to 1.

When $\left(\Gamma_w\right)_{ij} = 1$, it means that j-th internal node provides the i-th link of G_e. In case $\left(\Gamma_\phi\right)_{ij} = 1$, we conclude that the i-th link of G_e originates from the j-th input node. The links connected to the internal nodes are defined with the matrices Γ_w and Γ_ϕ. There are $L_w + L_\phi$ such links (*i.e. internal and input links*).

Remaining $L_\psi + L_z$ links of G_e are connected to the output nodes and they are defined by the matrices Γ_ψ and Γ_z (*i.e. output and external links*). For $\left(\Gamma_\psi\right)_{ij} = 1$, we conclude that the $(L_w + L_\phi + i)$-th link of G_e originates from the j-th internal node. Analogously, $\left(\Gamma_z\right)_{ij} = 1$ indicates that the j-th input node provides the $(L_w + L_\phi + i)$-th link of G_e.

In case all the links in G_e are one-dimensional, i.e. $p_i = 1$ and $\mu_j = 1$, where $i \in \mathcal{N}$, $j \in \mathcal{M}$, the following relations hold:

$$\begin{cases} u_d[k] & = \Gamma_w \cdot y_d[k] + \Gamma_\phi \cdot \eta[k] \\ \xi[k] & = \Gamma_\psi \cdot y_d[k] + \Gamma_z \cdot \eta[k] \end{cases}$$

The definitions of the matrices F_w, F_ϕ, F_ψ and F_z represent the generalization of the matrices Γ_w, Γ_ϕ, Γ_ψ and Γ_z, respectively, in case when not all the links in G_e are one-dimensional.

Table F.3: Notations for the processes in G_e

Notation	Explanation
k	Discrete time variable
t	Continuous time variable
s	Complex variable
n_i	Number of states of i-th node/system in G
n	Vector with number of states of each node/system in G
m_i	Dimension of the input vector u_i of the i-th node/system in G
m	Vector with dimension of the input vector u_i of each node/system in G ($i \in \mathcal{N}$)
p_i	Dimension of the output vector y_i of the i-th node/system in G
p	Vector with dimension of the output vector y_i of each node/system in G ($i \in \mathcal{N}$)
x_i	State vector of the i-th node/system in G
x_e	State vector of entire network
$X_e(s)$	Laplace transform of the state vector $x_e(t)$
u_i	Input vector of the i-th node/system in G
u_d	Aggregated input vectors u_i of each node/system in G ($i \in \mathcal{N}$)
$U_d(s)$	Laplace transform of the aggregated input vector $u_d(t)$
y_i	Output vector of the i-th node/system in G
y_d	Aggregated output vectors y_i of each node/system in G ($i \in \mathcal{N}$)
$Y_d(s)$	Laplace transform of the aggregated output vector $y_d(t)$
$\mathcal{M}$	Set of input nodes in G_e
r	Number of input nodes in G_e
μ_i	Dimension of the i-th external input vector η_i
μ	Vector with dimension of the external input vector η_i of each input node in G_e ($i \in \mathcal{M}$)
M	Sum of elements of the vector μ
η_i	The i-th external input vector in G_e
$H_i(s)$	Laplace transform of the i-th external input vector $\eta_i(t)$
η	Aggregated external input vector
$H(s)$	Laplace transform of the aggregated external input vector $\eta(t)$
$\mathcal{P}$	Set of output nodes in G_e
q	Number of output nodes in G_e
ρ_i	Dimension of the i-th external output vector ξ_i in G_e
ρ	Vector with dimension of the external output vector ξ_i of each input node in G_e ($i \in \mathcal{P}$)
P	Sum of elements of the vector ρ
ξ_i	The i-th external output vector in G_e
$\Xi_i(s)$	Laplace transform of the i-th external output vector $\xi_i(t)$
ξ	Aggregated external output vector
$\Xi(s)$	Laplace transform of the aggregated external output vector $\xi(t)$
$H_i(s)$	Matrix of transfer functions of the i-th node/system in G
$G_d(s)$	Diagonal matrix composed of matrices $H_i(s)$ of each node/system in G ($i \in \mathcal{N}$)
$G_e(s)$	Matrix of transfer functions of the entire network

F.3. PROOF OF THEOREM 21

After substituting the first relation from (7.31) into the second relation from (7.29) we obtain:

$$y_d[k] = C_d \cdot x_e[k] + D_d \cdot F_w \cdot y_d[k] + D_d \cdot F_\phi \cdot \eta[k]$$

Table F.4: Notations for the extended graph G_e

Notation	Explanation
G_e	Extended graph
$\mathscr{N}_e$	Set of N_e nodes of extended graph G_e
$\mathscr{L}_e$	Set of L_e links of extended graph G_e
N_e	Number of nodes in extended graph G_e
L_e	Number of links in extended graph G_e
W_e	Adjacency matrix of extended graph G_e
Λ	Incidence matrix of extended graph G_e
Γ	Transposed incidence matrix Λ with all negative entries set to 0
Γ_w	Internal sub-matrix of Γ
Γ_ϕ	Input sub-matrix of Γ
Γ_ψ	Output sub-matrix of Γ
Γ_z	External sub-matrix of Γ
Φ	Matrix that defines the input links existence
Ψ	Matrix that defines the output links existence
Z	Matrix that defines the external links existence
F	Extension of the matrix Γ for higher-dimensional vectors in G_e
F_w	Internal topology matrix, defined upon Γ_w
F_ϕ	Input topology matrix, defined upon Γ_ϕ
F_ψ	Output topology matrix, defined upon Γ_ψ
F_z	External topology matrix, defined upon Γ_z

Under the assumption $\det(I - D_d \cdot F_w)^{-1} \neq 0$, we further obtain:

$$y_d[k] = \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot x_e[k] + \left(I - D_d \cdot F_w\right)^{-1} \cdot (D_d \cdot F_\phi) \cdot \eta[k] \tag{F.1}$$

After substituting relation (F.1) into first relation from (7.31), we obtain the expression for the aggregated input vector u_d:

$$u_d[k] = F_w \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot x_e[k] + \left(F_w \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot (D_d \cdot F_\phi) + F_\phi\right) \cdot \eta[k] \tag{F.2}$$

Further, after substituting relation (F.2) into first relation from (7.29), we obtain:

$$x_e[k+1] = \left(A_d + B_d \cdot F_w \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d\right) \cdot x_e[k] + \left(B_d \cdot F_w \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot D_d \cdot F_\phi + B_d \cdot F_\phi\right) \cdot \eta[k]$$

from where we recognize the matrices A_e and B_e:

$$\begin{cases} A_e & = A_d + (B_d \cdot F_w) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \\ B_e & = (B_d \cdot F_w) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot (D_d \cdot F_\phi) + B_d \cdot F_\phi \end{cases}$$

Finally, after substituting expression for the aggregated output vector y_d from (F.1) into second relation from (7.31), we obtain:

$$\xi[k] = F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot x_e[k] + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot D_d \cdot F_\phi \cdot \eta[k] + F_z \cdot \eta[k]$$

Hence, we find:

$$\begin{cases} C_e & = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot C_d \\ D_e & = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot (D_d \cdot F_\phi) + F_z \end{cases}$$

which completes the proof. □

F.4. HOMOGENEOUS NETWORK WITH IDENTICAL INTERACTIONS BETWEEN THE NODES

In the following subsection, we examine the simplest network with linear processes, i.e. a homogeneous network (a network with nodes that perform identical internal dynamics) with identical dynamic interactions between the nodes/systems. Consequently, dimensions of the external input (7.8) and external output (7.10) vectors, as well of the input (7.6) and output vectors (7.7) are the same:

$$m_i = p_j = \mu_l = \rho_v = p_1 \quad i, j \in \mathcal{N} \quad l \in \mathcal{M} \quad v \in \mathcal{P} \tag{F.3}$$

Node/system i performs internal dynamics defined by (7.4), where the $n_i \times n_i$ state matrix A, the $n_i \times m_i$ input matrix B, the $p_i \times n_i$ output matrix C and the $p_i \times m_i$ feed-forward matrix D are identical for each node/system in the network. For identical interactions, instead of stacking incoming vectors of a certain node into an input/external output vector as in (7.16), they are summed:

$$\begin{aligned} u_i[k] &= \sum_{j \in \mathcal{N}, w_{ji}=1} y_j[k] + \sum_{l \in \mathcal{M}, \phi_{li}=1} \eta_l[k] \\ \xi_i[k] &= \sum_{j \in \mathcal{N}, \psi_{ji}=1} y_j[k] + \sum_{l \in \mathcal{M}, z_{li}=1} \eta_l[k] \end{aligned} \tag{F.4}$$

Therefore, Definition 20 for a homogeneous network with identical interactions reduces to:

$$\begin{cases} u_d[k] & = \left(W^T \otimes I_{p_1 \times p_1}\right) \cdot y_d[k] + \left(\Phi^T \otimes I_{p_1 \times p_1}\right) \cdot \eta[k] \\ \xi[k] & = \left(\Psi^T \otimes I_{p_1 \times p_1}\right) \cdot y_d[k] + \left(Z^T \otimes I_{p_1 \times p_1}\right) \cdot \eta[k] \end{cases} \tag{F.5}$$

Analogously to the Theorem 21, we provide the parameters of the DLSS model for the time dynamics of the entire network:

$$\begin{cases} A_e = \left(W^T \otimes B\right) \cdot \left(I_{Np_1 \times Np_1} - W^T \otimes D\right)^{-1} \cdot (I_N \otimes C) + (I_{N \times N} \otimes A) \\ B_e = \left(W^T \otimes B\right) \cdot \left(I_{Np_1 \times Np_1} - W^T \otimes D\right)^{-1} \cdot \left(\Phi^T \otimes D\right) + \left(\Phi^T \otimes B\right) \\ C_e = \left(\Psi^T \otimes I_{p_1 \times p_1}\right) \cdot \left(I_{Np_1 \times Np_1} - W^T \otimes D\right)^{-1} \cdot (I_{N \times N} \otimes C) \\ D_e = \left(\Psi^T \otimes I_{p_1 \times p_1}\right) \cdot \left(I_{Np_1 \times Np_1} - W^T \otimes D\right)^{-1} \cdot \left(\Phi^T \otimes D\right) + \left(Z^T \otimes I_{p_1 \times p_1}\right) \end{cases} \tag{F.6}$$

while, in the case, the $p_1 \times p_1$ feed-forward matrix $D = O_{p_1 \times p_1}$, the solution for parameters of the governing model (7.34) becomes considerably simpler:

$$\begin{cases} A_e = \left(W^T \otimes B \cdot C\right) + (I_{N \times N} \otimes A) \\ B_e = \left(\Phi^T \otimes B\right) \\ C_e = \left(\Psi^T \otimes C\right) \\ D_e = \left(Z^T \otimes I_{p_1 \times p_1}\right) \end{cases} \tag{F.7}$$

F

F.5. CONTINUOUS-TIME LINEAR PROCESSES ON COMPLEX NETWORKS

F.5.1. TIME-DOMAIN ANALYSIS

The continuous-time linear dynamics of the i-th node/system of the network obey a similar governing equation as (7.4):

$$\begin{cases} \frac{dx_i(t)}{dt} &= A_i \cdot x_i(t) + B_i \cdot u_i(t) \\ y_i(t) &= C_i \cdot x_i(t) + D_i \cdot u_i(t) \end{cases} \tag{F.8}$$

where t denotes continuous time. We revise the definition of the $\sum_{i=1}^{N} m_i \times 1$ aggregated input vector u_d from (7.30), the $\sum_{i=1}^{N} p_i \times 1$ aggregated output vector y_d from (7.30), the $M \times 1$ aggregated external input vector η from (7.9) and the $P \times 1$ aggregated external output vector ξ from (7.11) as follows:

$$\begin{aligned} u_d(t) &= \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_N(t) \end{bmatrix} & y_d(t) &= \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_N(t) \end{bmatrix} \\ \eta(t) &= \begin{bmatrix} \eta_1(t) \\ \eta_2(t) \\ \vdots \\ \eta_r(t) \end{bmatrix} & \xi(t) &= \begin{bmatrix} \xi_1(t) \\ \xi_2(t) \\ \vdots \\ \xi_q(t) \end{bmatrix} \end{aligned} \tag{F.9}$$

The aim is to determine the dynamics between the aggregated external output vector $\xi(t)$ and the aggregated external input vector $\eta(t)$, by following governing equations:

$$\begin{aligned} \frac{dx_e(t)}{dt} &= A_e \cdot x_e(t) + B_e \cdot \eta(t) \\ \xi(t) &= C_e \cdot x_e(t) + D_e \cdot \eta(t) \end{aligned} \tag{F.10}$$

where the $\sum_{i=1}^{N} n_i$ state vector $x_e(t)$ is defined as follows:

$$x_e(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_N(t) \end{bmatrix} \tag{F.11}$$

The direct continuous-time analogy of Theorem 21 in discrete-time domain is as follows:

Theorem 25 *The matrices A_e, B_e, C_e and D_e from the DLSS equations in (F.10),*

$$\begin{aligned} \frac{dx_e(t)}{dt} &= A_e \cdot x_e(t) + B_e \cdot \eta(t) \\ \xi(t) &= C_e \cdot x_e(t) + D_e \cdot \eta(t) \end{aligned}$$

F

provided the matrix $(I - D_d \cdot F_w)$ is non-singular or $(D_d \cdot F_w)$ has not an eigenvalue 1, are explicitly determined as follows:

$$\begin{cases} A_e = (B_d \cdot F_w) \cdot (I - D_d \cdot F_w)^{-1} \cdot C_d + A_d \\ B_e = (B_d \cdot F_w) \cdot (I - D_d \cdot F_w)^{-1} \cdot (D_d \cdot F_\phi) + B_d \cdot F_\phi \\ C_e = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot C_d \\ D_e = F_\psi \cdot (I - D_d \cdot F_w)^{-1} \cdot (D_d \cdot F_\phi) + F_z \end{cases} \tag{F.12}$$

while Corollary 1 remains the same.

F.5.2. LAPLACE-DOMAIN ANALYSIS

The unilateral (one-sided) Laplace transform, denoted as $\mathscr{L}\{f(t)\}$, of a continuous-time function $f(t)$ that is defined for all real numbers $t \geq 0$ is the complex function $F(s)$ defined as follows:

$$F(s) = \mathscr{L}\{f(t)\} = \int_0^\infty e^{-st} f(t) dt \tag{F.13}$$

where s is a complex variable. In case the function $f(t)$ is defined also for negative real numbers, the bilateral (two-sided) Laplace transform is defined as en extension of (F.13), where the limits of the integral become entire real axis:

$$F(s) = \mathscr{L}\{f(t)\} = \int_{-\infty}^\infty e^{-st} f(t) dt \tag{F.14}$$

The inverse Laplace transform, denoted as $\mathscr{L}^{-1}\{F(s)\}$ is defined by:

$$f(t) = \mathscr{L}^{-1}\{F(s)\} = \frac{1}{2\pi i} \lim_{T\to\infty} \oint_{\gamma - iT}^{\gamma + iT} e^{st} F(s)\, ds \tag{F.15}$$

where the real number γ defines the contour path of integration, that belongs to the region of convergence of $F(s)$.

The governing equations of the i-th node/system in continuous-time domain from (F.8) can be transformed into transfer functions using Laplace transform:

$$Y_i(s) = G_i(s) \cdot U_i(s) = \left(C_i \cdot (sI - A_i)^{-1} \cdot B_i + D_i\right) \cdot U_i(s) \tag{F.16}$$

where the $p_i \times 1$ complex output vector $Y_i(s)$ and the $m_i \times 1$ complex input vector $U_i(s)$ are the Laplace transforms of the output vector $y_i(t)$ and the input vector $u_i(t)$, respectively. The $p_i \times m_i$ complex matrix $G_i(s)$ is a matrix of transfer functions between the complex vectors $Y_d(s)$ and $U_d(s)$, where the $\left(G_i(s)\right)_{jk}$ transfer function defines the dynamics between the j-th component of the complex output vector $\left(Y_i(s)\right)_j$ and the k-th component of the complex input vector $\left(U_i(s)\right)_k$.

The Laplace transforms of the aggregated input vector $u_d(t)$, aggregated output vector $y_d(t)$, aggregated external input vector $\eta(t)$ and aggregated external output vector

F

$\xi(t)$ from (F.9) are defined as follows, respectively:

$$U_d(s) = \begin{bmatrix} U_1(s) \\ U_2(s) \\ \vdots \\ U_N(s) \end{bmatrix} \quad Y_d(s) = \begin{bmatrix} Y_1(s) \\ Y_2(s) \\ \vdots \\ Y_N(s) \end{bmatrix}$$
$$H(s) = \begin{bmatrix} H_1(s) \\ H_2(s) \\ \vdots \\ H_r(s) \end{bmatrix} \quad \Xi(s) = \begin{bmatrix} \Xi_1(s) \\ \Xi_2(s) \\ \vdots \\ \Xi_q(s) \end{bmatrix} \tag{F.17}$$

By defining the $\sum_{i=1}^{N} p_i \times \sum_{i=1}^{N} m_i$ complex matrix $G_d(s)$ as a block diagonal matrix, composed of the transfer functions $G_i(s)$ of each individual node/system (i.e. $i \in \mathcal{N}$):

$$G_d(s) = \text{diagonal}\begin{bmatrix} G_1(s) & G_2(s) & \dots & G_N(s) \end{bmatrix} = C_d \cdot (sI - A_d)^{-1} \cdot B_d + D_d \tag{F.18}$$

we are able to define the dynamics between the complex aggregated output vector $Y_d(s)$ and complex aggregated input vector $U_d(s)$ in a compact form:

$$Y_d(s) = G_d(s) \cdot U_d(s) \tag{F.19}$$

The aim of this subsection is to determine the $P \times M$ complex matrix $G_e(s)$ of transfer functions between the complex aggregated external output vector $\Xi(s)$ and the complex aggregated external input vector $H(s)$:

$$\Xi(s) = G_e(s) \cdot H(s) \tag{F.20}$$

where the $P \times 1$ complex aggregated external output vector $\Xi(s)$ and the $M \times 1$ complex aggregated external input vector $H(s)$ are Laplace transforms of the aggregated external input vector $\xi(t)$ and the aggregated external input vector $\eta(t)$, respectively.

The Laplace transform of the direct continuous-time analogy of Definition 1 in discrete-time domain is as follows:

$$\begin{cases} U_d(s) & = F_w \cdot Y_d(s) + F_\phi \cdot H(s) \\ \Xi(s) & = F_\psi \cdot Y_d(s) + F_z \cdot H(s) \end{cases} \tag{F.21}$$

Theorem 26 *The complex matrix $G_e(s)$ of transfer functions from (F.20) is explicitly determined as follows:*

$$G_e(s) = F_\psi \cdot \left(I - G_d(s) \cdot F_w\right)^{-1} \cdot G_d(s) \cdot F_\phi + F_z \tag{F.22}$$

Proof. We provide two different proofs of the Theorem 26. The first proof is based upon (F.21).

1) After substituting first relation from (F.21) into (F.19), we obtain:

$$Y_d(s) = G_d(s) \cdot F_w \cdot Y_d(s) + G_d(s) \cdot F_\phi \cdot H(s)$$

from where, under the assumption $\det(I - F_w \cdot G_d(s)) \neq 0$ we express the complex aggregated output vector $Y_d(s)$:

$$Y_d(s) = \left(I - G_d(s) \cdot F_w\right)^{-1} \cdot G_d(s) \cdot F_\phi \cdot H(s) \tag{F.23}$$

Next, we substitute (F.23) into second relation from (F.21) and obtain:

$$\Xi(s) = \left(F_\psi \cdot (I - G_d(s) \cdot F_w)^{-1} \cdot G_d(s) \cdot F_\phi + F_z\right) \cdot H(s)$$

which completes the proof.

2) In Theorem 25, the dynamics between the aggregated external output vector $\xi(t)$ and the aggregated external input vector $\eta(t)$ are determined by the governing equations in (F.10). Hence, the Laplace transform of the governing equations from (F.10) is actually the complex matrix $G_e(s)$ of transfer functions between the complex aggregated external output vector $\Xi(s)$ and the complex aggregated external input vector $H(s)$:

$$G_e(s) = C_e \cdot \left(sI - A_e\right)^{-1} \cdot B_e + D_e \tag{F.24}$$

After substituting (F.12) into (F.24), we obtain:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot \left(sI - \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d - A_d\right)^{-1} \\ & \cdot \left((B_d \cdot F_w) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + B_d \cdot F_\phi\right) \\ & + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.25}$$

We right multiply the inverse term $(sI - A_e)^{-1}$ from (F.25) with $(sI - A_d) \cdot (sI - A_d)^{-1}$ (i.e. with identity matrix) and regroup the terms inside the same term:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot \left(\left(sI - A_d\right) - \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d\right)^{-1} \\ & \cdot \left(sI - A_d\right) \cdot \left(sI - A_d\right)^{-1} \cdot \left(\left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + B_d \cdot F_\phi\right) \\ & + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.26}$$

After applying the property of a matrix inverse onto the product $(sI - A_e)^{-1} \cdot \left((sI - A_d)^{-1}\right)^{-1}$ from (F.26) we obtain:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d \cdot \left(I - \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot C_d\right)^{-1} \\ & \cdot \left(\left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right)\right) \\ & + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.27}$$

We define the $\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} p_i$ complex matrix $K(s)$ as follows:

$$K(s) = \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \tag{F.28}$$

F

and observe the following matrix product from (F.27):

$$C_d \cdot \left(I - K(s) \cdot C_d\right)^{-1} \tag{F.29}$$

We claim the next identity holds:

$$C_d \cdot \left(I - K(s) \cdot C_d\right)^{-1} = \left(I - C_d \cdot K(s)\right)^{-1} \cdot C_d \tag{F.30}$$

where the identity matrix I from the left-hand side is of dimensions $\sum_{i=1}^{N} n_i \times \sum_{i=1}^{N} n_i$ and the identity matrix from the right-hand side of (F.30) has dimensions $\sum_{i=1}^{N} p_i \times \sum_{i=1}^{N} p_i$. We prove (F.30) by contradiction.

We denote the difference between the left-hand and the right-hand side of (F.30) as a complex matrix $E(s)$ of dimensions $\sum_{i=1}^{N} p_i \times \sum_{i=1}^{N} n_i$:

$$E(s) = C_d \cdot \left(I - K(s) \cdot C_d\right)^{-1} - \left(I - C_d \cdot K(s)\right)^{-1} \cdot C_d \tag{F.31}$$

After left multiplying with $\left(I - C_d \cdot K(s)\right)$ and right multiplying with $\left(I - K(s) \cdot C_d\right)$ both sides of (F.31) we obtain:

$$\left(I - C_d \cdot K(s)\right) \cdot E(s) \cdot \left(I - K(s) \cdot C_d\right) = O$$

Left side of last equation is always zero, thus we conclude $E(s) = O$. We import the proven identity (F.30) into (F.27) and obtain:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(I - C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1}\right)^{-1} \\ & \cdot \Big(C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) \\ & + C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right)\Big) + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.32}$$

We regroup the terms inside the fourth product term of (F.32) in such a way to build a matrix, whose inverse appears as the third product term in (F.32):

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(I - C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1}\right)^{-1} \\ & \cdot \Big(-\left(I - C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1}\right) \cdot \left(D_d \cdot F_\phi\right) \\ & + \left(D_d \cdot F_\phi\right) + C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right)\Big) \\ & + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.33}$$

After multiplying the third and the fourth product terms from (F.33), we obtain:

$$\begin{aligned} G_e(s) =& - F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \\ & \left(I - C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1}\right)^{-1} \\ & \cdot \left(C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right) + \left(D_d \cdot F_\phi\right)\right) \\ & + F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.34}$$

The first and the third sum terms from (F.34) are the same, but with opposite signs. Hence, we obtain:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - D_d \cdot F_w\right)^{-1} \cdot \left(I - C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) \cdot \left(I - D_d \cdot F_w\right)^{-1}\right)^{-1} \\ & \cdot \left(C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right) + \left(D_d \cdot F_\phi\right)\right) + F_z \end{aligned} \tag{F.35}$$

Finally, after applying the property of a matrix inverse onto the product of the second and third product terms in (F.35), we obtain the final form for $G_e(s)$:

$$\begin{aligned} G_e(s) =& F_\psi \cdot \left(I - \left(C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_w\right) + D_d \cdot F_w\right)\right)^{-1} \\ & \cdot \left(C_d \cdot \left(sI - A_d\right)^{-1} \cdot \left(B_d \cdot F_\phi\right) + D_d \cdot F_\phi\right) + F_z \end{aligned} \tag{F.36}$$

which equals (F.22) and completes the proof. □

F

Bibliography

[1] Albert-László Barabási. "Network science". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1987 (2013), p. 20120375.

[2] Mark Newman. *Networks*. Oxford university press, 2018.

[3] Leonhard Euler. "Solutio problematis ad geometriam situs pertinentis". In: *Commentarii academiae scientiarum Petropolitanae* (1741), pp. 128–140.

[4] Piet Van Mieghem et al. "Spectral graph analysis of modularity and assortativity". In: *Physical Review E* 82.5 (2010), p. 056113.

[5] Alex Arenas et al. "Synchronization in complex networks". In: *Physics reports* 469.3 (2008), pp. 93–153.

[6] Romualdo Pastor-Satorras et al. "Epidemic processes in complex networks". In: *Reviews of modern physics* 87.3 (2015), p. 925.

[7] Karel Devriendt and Piet Van Mieghem. "The simplex geometry of graphs". In: *Journal of Complex Networks* 7.4 (2019), pp. 469–490.

[8] Santo Fortunato. "Community detection in graphs". In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[9] Gabriel Budel and Piet Van Mieghem. "Detecting the number of clusters in a network". In: *Journal of Complex Networks* 8.6 (2020), cnaa047.

[10] Mark EJ Newman. "Communities, modules and large-scale structure in networks". In: *Nature physics* 8.1 (2012), pp. 25–31.

[11] Mark EJ Newman and Michelle Girvan. "Finding and evaluating community structure in networks". In: *Physical review E* 69.2 (2004), p. 026113.

[12] Mark EJ Newman. "Fast algorithm for detecting community structure in networks". In: *Physical review E* 69.6 (2004), p. 066133.

[13] Sune Lehmann and Lars Kai Hansen. "Deterministic modularity optimization". In: *The European Physical Journal B* 60.1 (2007), pp. 83–88.

[14] Vincent D Blondel et al. "Fast unfolding of communities in large networks". In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[15] Luca Donetti and Miguel A Munoz. "Detecting network communities: a new systematic and efficient algorithm". In: *Journal of Statistical Mechanics: Theory and Experiment* 2004.10 (2004), P10012.

[16] Ingve Simonsen. "Diffusion and networks: A powerful combination!" In: *Physica A: Statistical Mechanics and its Applications* 357.2 (2005), pp. 317–330.

[17] Nelson A Alves. "Unveiling community structures in weighted networks". In: *Physical Review E* 76.3 (2007), p. 036101.

[18] Andrea Capocci et al. "Detecting communities in large networks". In: *Physica A: Statistical Mechanics and its Applications* 352.2-4 (2005), pp. 669–676.

[19] Florent Krzakala et al. "Spectral redemption in clustering sparse networks". In: *Proceedings of the National Academy of Sciences* 110.52 (2013), pp. 20935–20940.

[20] Fa-Yueh Wu. "The potts model". In: *Reviews of modern physics* 54.1 (1982), p. 235.

[21] Marcelo Blatt, Shai Wiseman, and Eytan Domany. "Superparamagnetic clustering of data". In: *Physical review letters* 76.18 (1996), p. 3251.

[22] I Ispolatov, I Mazo, and A Yuryev. "Finding mesoscopic communities in sparse networks". In: *Journal of Statistical Mechanics: Theory and Experiment* 2006.09 (2006), P09014.

[23] Piet Van Mieghem, Karel Devriendt, and H Cetinay. "Pseudoinverse of the Laplacian and best spreader node in a network". In: *Physical Review E* 96.3 (2017), p. 032311.

[24] Daniel A Spielman and Nikhil Srivastava. "Graph sparsification by effective resistances". In: *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 2008, pp. 563–568.

[25] Prasad Tetali. "Random walks and the effective resistance of networks". In: *Journal of Theoretical Probability* 4.1 (1991), pp. 101–109.

[26] Vedat Levi Alev et al. "Graph clustering using effective resistance". In: *arXiv preprint arXiv:1711.06530* (2017).

[27] Piet Van Mieghem. "A tree realization of a distance matrix: the inverse shortest path problem with a demand matrix generated by a tree". In: *TU Delft report 20211012* (Nov. 2022), pp. 1–16.

[28] Daniel A Spielman and Shang-Hua Teng. "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems". In: *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. 2004, pp. 81–90.

[29] Kook Jin Ahn and Sudipto Guha. "Graph sparsification in the semi-streaming model". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2009, pp. 328–338.

[30] Joshua Batson et al. "Spectral sparsification of graphs: theory and algorithms". In: *Communications of the ACM* 56.8 (2013), pp. 87–94.

[31] Mark EJ Newman. "The structure and function of complex networks". In: *SIAM review* 45.2 (2003), pp. 167–256.

[32] Baruch Barzel and Albert-László Barabási. "Universality in network dynamics". In: *Nature physics* 9.10 (2013), p. 673.

[33] Uzi Harush and Baruch Barzel. "Dynamic patterns of information flow in complex networks". In: *Nature communications* 8.1 (2017), pp. 1–11.

[34] Baruch Barzel, Yang-Yu Liu, and Albert-László Barabási. "Constructing minimal models for complex system dynamics". In: *Nature communications* 6 (2015), p. 7186.

[35] Alejandro D Sánchez, Juan M López, and Miguel A Rodriguez. "Nonequilibrium phase transitions in directed small-world networks". In: *Physical review letters* 88.4 (2002), p. 048701.

[36] Remco Van Der Hofstad, Gerard Hooghiemstra, and Piet Van Mieghem. "First-passage percolation on the random graph". In: *Probability in the Engineering and Informational Sciences* 15.2 (2001), pp. 225–237.

[37] Steven H Strogatz. "From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators". In: *Physica D: Nonlinear Phenomena* 143.1-4 (2000), pp. 1–20.

[38] Naoki Masuda, Mason A Porter, and Renaud Lambiotte. "Random walks and diffusion on networks". In: *Physics reports* 716 (2017), pp. 1–58.

[39] Bastian Prasse and Piet Van Mieghem. "The viral state dynamics of the discrete-time NIMFA epidemic model". In: *IEEE Transactions on Network Science and Engineering* 7.3 (2019), pp. 1667–1674.

[40] Zhidong He and Piet Van Mieghem. "Optimal Induced Spreading of SIS Epidemics in Networks". In: *IEEE Transactions on Control of Network Systems* (2018).

[41] Cheng-yi Xia et al. "Effects of delayed recovery and nonuniform transmission on the spreading of diseases in complex networks". In: *Physica A: Statistical Mechanics and its Applications* 392.7 (2013), pp. 1577–1585.

[42] Chengyi Xia et al. "A new coupled disease-awareness spreading model with mass media on multiplex networks". In: *Information Sciences* 471 (2019), pp. 185–200.

[43] Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. "Random graphs with arbitrary degree distributions and their applications". In: *Physical review E* 64.2 (2001), p. 026118.

[44] M Van den Berg et al. "A macroscopic traffic flow model for integrated control of freeway and urban traffic networks". In: *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*. Vol. 3. IEEE. 2003, pp. 2774–2779.

[45] Alain Barrat, Marc Barthelemy, and Alessandro Vespignani. *Dynamical processes on complex networks*. Cambridge university press, 2008.

[46] Steven H Strogatz. "Exploring complex networks". In: *nature* 410.6825 (2001), p. 268.

[47] Linying Xiang et al. "Advances in Network Controllability". In: *IEEE Circuits and Systems Magazine* 19 (2019), pp. 8–32.

[48] Yang-Yu Liu and Albert-László Barabási. "Control principles of complex systems". In: *Reviews of Modern Physics* 88.3 (2016), p. 035006.

[49] Ivan Jokić and Piet Van Mieghem. "Linear processes on complex networks". In: *Journal of Complex Networks* 8.4 (2020), cnaa030.

[50] Piet Van Mieghem and Ivan Jokić. “Co-eigenvector graphs”. Manuscript submitted for publication. 2022.

[51] Piet Van Mieghem. *Graph spectra for complex networks.* Cambridge University Press, 2010.

[52] Dragan Stevanović. *Spectral radius of graphs.* Academic Press, 2014.

[53] D Cvetkovic, P Rowlinson, and SK Simic. “Eigenspaces of Graphs, Cambridge Uni”. In: *Press, Cambridge* (1997).

[54] Dragoš M Cvetković, Peter Rowlinson, and Slobodan Simić. *An introduction to the theory of graph spectra.* Vol. 75. Cambridge University Press Cambridge, 2010.

[55] Edwin R Van Dam and Willem H Haemers. “Which graphs are determined by their spectrum?” In: *Linear Algebra and its applications* 373 (2003), pp. 241–272.

[56] Gustaaf Borghs et al. “Band-gap narrowing in highly doped n-and p-type GaAs studied by photoluminescence spectroscopy”. In: *Journal of applied physics* 66.9 (1989), pp. 4381–4386.

[57] Dennis Iligan Merino. *Topics in matrix analysis.* The Johns Hopkins University, 1992.

[58] S Barik, S Fallat, and S Kirkland. “On Hadamard diagonalizable graphs”. In: *Linear algebra and its applications* 435.8 (2011), pp. 1885–1902.

[59] Edwin R van Dam, Jack H Koolen, and Hajime Tanaka. “Distance-regular graphs”. In: *arXiv preprint arXiv:1410.6294* (2014).

[60] Dajie Liu, Huijuan Wang, and Piet Van Mieghem. “Spectral perturbation and reconstructability of complex networks”. In: *Physical Review E* 81.1 (2010), p. 016101.

[61] P Van Mieghem. “Can the topology of two graphs be compared by one number?” In: *Delft University of Technology* 72 (2013).

[62] Santiago Segarra et al. “Network topology inference from spectral templates”. In: *IEEE Transactions on Signal and Information Processing over Networks* 3.3 (2017), pp. 467–483.

[63] Ivan Jokić and Piet Van Mieghem. “Number of paths in a graph”. In: *arXiv preprint arXiv:2209.08840* (2022).

[64] Piet Van Mieghem. *Performance analysis of complex networks and systems.* Cambridge University Press, 2014.

[65] Piet Van Mieghem. “Paths in the simple random graph and the Waxman graph”. In: *Probability in the Engineering and Informational Sciences* 15.4 (2001), pp. 535–555.

[66] Andrzej Grzesik et al. “The maximum number of paths of length three in a planar graph”. In: *Journal of Graph Theory* (2022).

[67] Charles Eric Leiserson et al. *Introduction to algorithms.* Vol. 3. MIT press, 1994.

[68] Ryan Williams. “Finding paths of length k in $O(2^k)$ time”. In: *Information Processing Letters* 109.6 (2009), pp. 315–318.

[69] Andreas Schmid and Jens M Schmidt. “Computing Tutte paths”. In: *arXiv preprint arXiv:1707.05994* (2017).

[70] Andreas Bjorklund. “Determinant sums for undirected Hamiltonicity”. In: *SIAM Journal on Computing* 43.1 (2014), pp. 280–299.

[71] Eric T Bax. “Inclusion and exclusion algorithm for the Hamiltonian path problem”. In: *Information Processing Letters* 47.4 (1993), pp. 203–207.

[72] Zhihao Qiu et al. “The Inverse ALL Shortest Path Problem”. Unpublished Manuscript. 2023.

[73] Douglas J Klein and Milan Randić. “Resistance distance”. In: *Journal of mathematical chemistry* 12 (1993), pp. 81–95.

[74] Gustav Kirchhoff. “Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird”. In: *Annalen der Physik* 148.12 (1847), pp. 497–508.

[75] Ashok K Chandra et al. “The electrical resistance of a graph captures its commute and cover times”. In: *Proceedings of the twenty-first annual ACM symposium on Theory of computing*. 1989, pp. 574–586.

[76] Mark Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[77] Peter G Doyle and J Laurie Snell. *Random walks and electric networks*. Vol. 22. American Mathematical Soc., 1984.

[78] David G Wagner. “Combinatorics of electrical networks”. In: *Lecture Notes, Dept. of C&O, University of Waterloo* (2009).

[79] Alexander Mercier, Samuel Scarpino, and Cristopher Moore. “Effective resistance against pandemics: Mobility network sparsification for high-fidelity epidemic simulations”. In: *PLOS Computational Biology* 18.11 (2022), e1010650.

[80] Piet Van Mieghem. *Graph Spectra for Complex Networks*. Cambridge University Press, Dec. 2010, pp. 1–346.

[81] Ivan Jokić and Piet Van Mieghem. “Linear Clustering Process on Networks”. In: *IEEE Transactions on Network Science and Engineering* (2023), pp. 1–10.

[82] Brian Karrer, Mark EJ Newman, and Lenka Zdeborová. “Percolation on sparse networks”. In: *Physical review letters* 113.20 (2014), p. 208702.

[83] H Eugene Stanley. *Phase transitions and critical phenomena*. Vol. 7. Clarendon Press, Oxford, 1971.

[84] Mark EJ Newman. “Modularity and community structure in networks”. In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[85] Xiaowei Xu et al. “Scan: a structural clustering algorithm for networks”. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007, pp. 824–833.

[86] Andrea Lancichinetti and Santo Fortunato. “Community detection algorithms: a comparative analysis”. In: *Physical review E* 80.5 (2009), p. 056117.

[87] Tiago P Peixoto. “Hierarchical block structures and high-resolution model selection in large networks”. In: *Physical Review X* 4.1 (2014), p. 011047.

[88] Leon Danon et al. “Comparing community structure identification”. In: *Journal of statistical mechanics: Theory and experiment* 2005.09 (2005), P09008.

[89] Pascal Pons and Matthieu Latapy. “Computing communities in large networks using random walks”. In: *Computer and Information Sciences-ISCIS 2005: 20th International Symposium, Istanbul, Turkey, October 26-28, 2005. Proceedings 20*. Springer. 2005, pp. 284–293.

[90] Renaud Lambiotte et al. “Flow graphs: Interweaving dynamics and structure”. In: *Physical Review E* 84.1 (2011), p. 017102.

[91] Alex Arenas, Albert Diaz-Guilera, and Conrad J Pérez-Vicente. “Synchronization reveals topological scales in complex networks”. In: *Physical review letters* 96.11 (2006), p. 114102.

[92] Di Jin et al. “A survey of community detection approaches: From statistical modeling to deep learning”. In: *IEEE Transactions on Knowledge and Data Engineering* (2021).

[93] Ulrik Brandes et al. “On modularity clustering”. In: *IEEE transactions on knowledge and data engineering* 20.2 (2007), pp. 172–188.

[94] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[95] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. “Stochastic blockmodels: First steps”. In: *Social networks* 5.2 (1983), pp. 109–137.

[96] Michelle Girvan and Mark EJ Newman. “Community structure in social and biological networks”. In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[97] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. “Benchmark graphs for testing community detection algorithms”. In: *Physical review E* 78.4 (2008), p. 046110.

[98] Omer Angel, Joel Friedman, and Shlomo Hoory. “The non-backtracking spectrum of the universal cover of a graph”. In: *Transactions of the American Mathematical Society* 367.6 (2015), pp. 4287–4318.

[99] Ivan Jokić et al. “Time Dynamics of the Dutch Municipality Network”. Manuscript submitted for publication. 2022.

[100] C. Marchetti. “Anthropological invariants in travel behavior”. In: *Technological Forecasting and Social Change* 47.1 (Sept. 1994), pp. 75–88.

[101] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. “Understanding individual human mobility patterns”. In: *nature* 453.7196 (2008), pp. 779–782.

[102] Rosario N. Mantegna and H. Eugene Stanley. “Stochastic Process with Ultraslow Convergence to a Gaussian: The Truncated Lévy Flight”. In: *Physical Review Letters* 73.22 (Nov. 1994), pp. 2946–2949.

[103] D. Brockmann, L. Hufnagel, and T. Geisel. “The scaling laws of human travel”. In: *Nature 2006 439:7075* 439.7075 (Jan. 2006), pp. 462–465.

[104] Stefan Rayer and David L. Brown. “Geographic diversity of inter-county migration in the united states, 1980–1995”. In: *Population Res. Policy Rev.* 20.3 (2001), pp. 229–252.

[105] Kenneth M. Johnson and Glenn V. Fuguitt. “Continuity and change in rural migration patterns, 1950–1995”. In: *Rural Sociol.* 65.1 (2000), pp. 27–49.

[106] Hernán A. Makse, Shlomo Havlin, and H. Eugene Stanley. “Modelling urban growth patterns”. In: *Nature 1995 377:6550* 377.6550 (Oct. 1995), pp. 608–612.

[107] Markus Schläpfer et al. “The scaling of human interactions with city size”. In: *Journal of The Royal Society Interface* 11.98 (Sept. 2014).

[108] Vincent Verbavatz and Marc Barthelemy. “The growth equation of cities”. In: *Nature* 587.7834 (Nov. 2020), pp. 397–401.

[109] Luís M.A. Bettencourt et al. “Growth, innovation, scaling, and the pace of life in cities”. In: *Proc. Natl Acad. Sci.* 104.17 (Apr. 2007), pp. 7301–7306.

[110] Luís M.A. Bettencourt. “The origins of scaling in cities”. In: *Science* 340.6139 (2013), pp. 1438–1441.

[111] A. Schneider and C. M. Mertes. “Expansion and growth in Chinese cities, 1978–2010”. In: *Environ. Res. Lett.s* 9.2 (2014), p. 024008.

[112] Rolf Bergs. “Spatial dependence in the rank-size distribution of cities – weak but not negligible”. In: *PLOS ONE* 16.2 (Feb. 2021), e0246796.

[113] M. E. J. Newman. “The Structure and Function of Complex Networks”. In: *SIAM Review* 45.2 (Jan. 2003), pp. 167–256.

[114] Albert-László Barabási. “Network science introduction”. In: *Network science* (2016), pp. 1–27.

[115] Hermine Molnár-in ‘t Veld. *De groei van het Nederlandse personenautopark.* 2019.

[116] Josje J. Hoekveld. “Urban decline within the region: Understanding the intra-regional differentiation in urban population development in the declining regions Saarland and Southern-Limburg”. PhD thesis. University of Amsterdam, 2014, p. 181.

[117] Onno Boonstra and Rick Mourits. *Historical Database of Dutch Municipalities (Historische Database Nederlandse Gemeenten (HDNG)) - dataLegend - Druid.*

[118] Pandu R. Tadikamalla and Norman L. Johnson. “Systems of Frequency Curves Generated by Transformations of Logistic Variables”. In: *Biometrika* 69.2 (Aug. 1982), p. 461.

[119] Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. “Power-Law Distributions in Empirical Data”. In: *SIAM Review* 51.4 (Nov. 2009), pp. 661–703.

[120] Christian Doerr, Norbert Blenn, and Piet Van Mieghem. “Lognormal Infection Times of Online Information Spread”. In: *PLoS ONE* 8.5 (May 2013). Ed. by Alain Barrat.

[121] P. Van Mieghem, N. Blenn, and C. Doerr. "Lognormal distribution in the digg online social network". In: *The European Physical Journal B* 83.2 (Sept. 2011), pp. 251–261.

[122] Piet. Van Mieghem. "Data communications networking". In: (2010), p. 414.

[123] Matthieu Cristelli, Michael Batty, and Luciano Pietronero. "There is More than a Power Law in Zipf". In: *Scientific Reports 2012 2:1* 2.1 (Nov. 2012), pp. 1–7.

[124] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions.* Revised 9. Dover Publications, June 1968, pp. 1–1046.

[125] Florian Dorfler and Francesco Bullo. "Kron reduction of graphs with applications to electrical networks". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 60.1 (2012), pp. 150–163.

[126] Florian Dörfler, John W Simpson-Porco, and Francesco Bullo. "Electrical networks and algebraic graph theory: Models, properties, and applications". In: *Proceedings of the IEEE* 106.5 (2018), pp. 977–1005.

[127] Hale Cetinay, Karel Devriendt, and Piet Van Mieghem. "Nodal vulnerability to targeted attacks in power grids". In: *Applied network science* 3.1 (2018), p. 34.

[128] Roger Guimera et al. "The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles". In: *Proceedings of the National Academy of Sciences* 102.22 (2005), pp. 7794–7799.

[129] Jennifer A Dunne, Richard J Williams, and Neo D Martinez. "Food-web structure and network theory: the role of connectance and size". In: *Proceedings of the National Academy of Sciences* 99.20 (2002), pp. 12917–12922.

[130] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. "On power-law relationships of the internet topology". In: *ACM SIGCOMM computer communication review.* Vol. 29. 4. ACM. 1999, pp. 251–262.

[131] Stefano Boccaletti et al. "Complex networks: Structure and dynamics". In: *Physics reports* 424.4-5 (2006), pp. 175–308.

[132] Gang Yan et al. "Network control principles predict neuron function in the Caenorhabditis elegans connectome". In: *Nature* 550.7677 (2017), p. 519.

[133] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. "A survey of multi-agent formation control". In: *Automatica* 53 (2015), pp. 424–440.

[134] Johan Schoukens and Lennart Ljung. "Nonlinear System Identification: A User-Oriented Roadmap". In: *arXiv preprint arXiv:1902.00683* (2019).

[135] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. "Consensus and cooperation in networked multi-agent systems". In: *Proceedings of the IEEE* 95.1 (2007), pp. 215–233.

[136] Linying Xiang et al. "Controllability of Directed Heterogeneous Networked MIMO Systems". In: *arXiv preprint arXiv:1812.03302* (2018).

[137] Bastian Prasse and Piet Van Mieghem. "Time-dependent solution of the NIMFA equations around the epidemic threshold". In: *Journal of mathematical biology* 81.6-7 (2020), pp. 1299–1355.

[138] Bart De Moor et al. "A geometrical strategy for the identification of state space models of linear multivariable systems with singular value decomposition". In: *IFAC Proceedings Volumes* 21.9 (1988), pp. 493–497.

[139] P Van Mieghem et al. "A framework for computing topological network robustness". In: *Delft University of Technology, Report20101218* (2010).

[140] Michel Verhaegen and Vincent Verdult. *Filtering and system identification: a least squares approach*. Cambridge university press, 2007.

[141] S Akbari, PJ Cameron, and GB Khosrovshahi. "Ranks and signatures of adjacency matrices". In: *preprint* (2004).

[142] Carl D Meyer. *Matrix analysis and applied linear algebra*. Vol. 71. Siam, 2000.

[143] Albert W Marshall, Ingram Olkin, and Barry C Arnold. *Inequalities: theory of majorization and its applications*. Vol. 143. Springer, 1979.

[144] Piet Van Mieghem. "Graph eigenvectors, fundamental weights and centrality metrics for nodes in networks". In: *arXiv preprint arXiv:1401.4580* (2014).

[145] Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. "From Louvain to Leiden: guaranteeing well-connected communities". In: *Scientific reports* 9.1 (2019), p. 5233.

[146] Aurelien Decelle et al. "Inference and phase transitions in the detection of modules in sparse networks". In: *Physical Review Letters* 107.6 (2011), p. 065701.

[147] Aurelien Decelle et al. "Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications". In: *Physical Review E* 84.6 (2011), p. 066106.

[148] Lev Muchnik et al. "Origins of power-law degree distribution in the heterogeneity of human activity in social networks". In: *Scientific reports* 3.1 (2013), p. 1783.

[149] Gergely Palla et al. "Uncovering the overlapping community structure of complex networks in nature and society". In: *nature* 435.7043 (2005), pp. 814–818.

[150] P Van Mieghem. "Approximate formula and bounds for the time-varying susceptible-infected-susceptible prevalence in networks". In: *Physical Review E* 93.5 (2016), p. 052312.

[151] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[152] Erich Leo Lehmann, Joseph P Romano, and George Casella. *Testing statistical hypotheses*. Vol. 3. Springer, 2005.

Curriculum Vitæ

Ivan Jokić

Ivan Jokić obtained a bachelor's degree in Energetics and Control Theory and a master's degree in Power Systems and Automatic Control in Control Theory, from the University of Montenegro, in 2015 and 2018, respectively. Since February 2019, he has been pursuing his PhD degree in the Network Architecture and Services (NAS) group, within the Faculty of Electrical Engineering, Mathematics and Computer Science, at the Delft University of Technology, under the supervision of Prof.dr.ir. Piet Van Mieghem.

His main research interests include graph theory, network dynamics, systems theory, and the identification of networked systems.

List of Publications

6. Z. Qiu, **I. Jokić**, Siyu Tang, Rogier Noldus and P. Van Mieghem, *The Inverse ALL Shortest Path Problem*, in preparation (2023).

5. **I. Jokić**, E. van Boven, I. Manolopoulos, T. Verma, G. Buiten, F. Pijpers, H. van Hooff and P. Van Mieghem, *Time Dynamics of the Dutch Municipality Network*, under review (2022).

4. P. Van Mieghem and **I. Jokić**, *Co-eigenvector graphs*, under review (2022).

3. **I. Jokić** and P. Van Mieghem, *Linear Clustering Process on Networks*, IEEE Transactions on Network Science and Engineering, **doi: 10.1109/TNSE.2023.3271360**, (2023).

2. **I. Jokić** and P. Van Mieghem, *Number of paths in a graph*, arXiv preprint **arXiv:2209.08840** (2022).

1. **I. Jokić** and P. Van Mieghem, *Linear Processes on Complex Networks*, Journal of Complex Networks **cna030, 8(4),** (2020).