Author:
Kempton, Louis

Title:
Distributed Control and Optimisation of Complex Networks via their Laplacian Spectra

Distributed Control and Optimisation of Complex Networks via their Laplacian Spectra

By

LOUIS KEMPTON

Department of Engineering Mathematics
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering.

OCTOBER 2017

Word count: 52,091

Abstract

In complex networked systems, the structure of the underlying communication between individual agents can have profound impacts on the performance and dynamics of the system as a whole. For example, processes such as consensus can occur at faster or slower rates depending on the structure of the communication graph, and the synchronisation of coupled chaotic oscillators may be unachievable in one configuration, when it is achievable in another. As such, it is vital for agents within a complex networked system to be able to make estimates of the properties of the network as a whole, and be able to direct their own actions to modify these properties in a desirable way, even when they are only able to communicate with their direct neighbours, and have no global knowledge of the structure of the network. In this thesis we explore decentralised strategies by which individual agents in a network can make estimates of several functions of the graph Laplacian matrix eigenvalues, and control or optimise these functions in a desired manner, subject to constraints. We focus on the following spectral functions of the graph Laplacian matrix, which determine or bound many interesting properties of graphs and dynamics on networks: the algebraic connectivity (the smallest non-zero eigenvalue), the spectral radius (the largest eigenvalue), the ratio between these extremal eigenvalues (also known as the synchronisability ratio), the total effective graph resistance (proportional to the sum of the reciprocals of non-zero eigenvalues), and the reduced determinant (the product of the non-zero eigenvalues).

DEDICATION AND ACKNOWLEDGEMENTS

In dedication to my parents, for their endless encouragement, and to my darling Rebecca, without whose patience and support, I may have been writing this from a cave in the Avon Gorge.

I would also like to acknowledge and give my deepest thanks to my supervisors, Mario di Bernardo and Guido Herrmann, who have provided invaluable guidance and discussion. And I would also like to thank all those students in the BCCS whose friendship and conversations have made Bristol wonderful.

Finally, this thesis would not have been possible without the generous funding from the EPSRC, through the Bristol Centre for Complexity Sciences [EP/I013717/1].

Author's declaration

Table of Contents

Page

List of Tables

List of Figures

Chapter 1

Introduction

Networked systems are ubiquitous within both natural and engineering systems. We see networks of interconnected, individual, and dynamic agents present in the connected sources and sinks in power grids [1, 2, 3], hyperlinked pages in the world wide web [4, 5], friendship groups in social networks [6, 7], food webs in ecology [8, 9], and even in the neurons forming the connectome of the brain [10, 11]. It is important to realise that in networked systems, the performance and properties of the whole system are not only resultant from the performance of the individual parts that constitute the networked system, but also from the interactions between the parts, and the properties of the underlying structure of the network. It may, for example, prove impossible for a network of coupled oscillators to synchronise for specific network structures [12, 13]. Furthermore, with relatively little edge rewiring, global properties of networks such as the diameter and mean path length can rapidly change resulting in radically different collective behaviours [14, 15].

It is the structure of networks which is the primary topic of interest in this thesis, and in particular, how individual agents within a network, following simple local rules, can adapt their structure over time so the network itself can better perform its function. The human brain (and the network of interconnected neurons that comprise it) is often seen as the epitome of natural networked systems that exhibit this *complex adaptive behaviour*; reorganising itself and learning from past experience. This self-organisation of the network of neurons is seen both in the changing strengths of synapses [16], and in the rewiring of collections of neurons [17]. Adaptive behaviours such as these could afford engineering networks increased robustness, allowing networks to manage broken nodes or other faults in the network with minimal cost to performance. Indeed, devising strategies for adapting networks to improve and control desired attributes has become a fruitful direction for research [18, 19, 20, 21]. Self-organising behaviours remove the need for a dedicated centralised designer, giving increased flexibility to autonomous robotic networks which may be required to operate in a dynamic environment. Moreover, self-organisation could be a great advantage for the scalability of large networks, where distributed computing power will grow linearly with the size of the network [22]. It is these complex adaptive and self-organising behaviours which we aim to replicate in the following chapters of this thesis.

1.1 Complex Systems

There have been many attempts to define and categorise the ideas of complexity and complex systems since their inception [23, 24, 25]. Central to most definitions of complexity are the ideas of *emergent* properties (properties of the whole which are not self-evident from the properties of the parts), spontaneous order (which may be viewed as self-organisation), and hierarchical structure. In Mitchell [26], page 12, a definition of a complex system is given as “A system in which large networks of components with no central control and simple rules of operation give rise to complex collective behaviour, sophisticated information processing, and adaptation via learning or evolution”. By observing systems we think of as complex, and the previous attempts at defining complex systems, Ladyman, Lambert, and Wiesner [27] devised a core list of properties that are often features of complex systems. This list includes: *Nonlinearity* - though it is argued that both nonlinearity and chaos are neither sufficient nor necessary conditions for complexity to arise (but is a frequent source of the emergent properies that arise); *Feedback* - which is fundamentally important and necessary for any system that adapts to its environment; *Spontaneous Order* - which relates to pattern formation, symmetry, and self-organisation; *Robustness and Lack of Central Control* - this will be a central theme of the control strategies presented later in this thesis; *Emergence* - which is often characterised by limitations of reductionist methods, and often hazily described as “the whole being greater than the sum of its parts”, and hence is often related to non-linear dynamics; *Hierarchical Organisation* - which again relates to self-organisation, but also to clustering of subsystems into systems; and finally *Numerosity* - the large number of individual agents and interactions between them within the collective.

These features of a complex system could be broken down into those which are inherrent properties of the system and the agents within: nonlinearity, feedback, lack of central control, and numerosity, and those which are resultant from the behaviour of the system: spontaneous order through self-organisation, emergence, and robustness to changing environments. It is these resultant behaviours of the whole, which emerge through a multitude of local interactions, which we want to capture and replicate by formulating decentralised control laws which use feedback and the inherrent numerosity present in networks.

In Liu, Slotine, and Barabási [28], it was stated that: “The ultimate proof of our understanding of natural or technological systems is reflected in our ability to control them”. In a similar vein, we seek to design these simple local interactions within a network so that a desired global behaviour emerges. Namely, we desire that the network of autonomous individual agents is able to adapt and self-organise, without any central control, into a globally optimal configuration that maximises some property of the network as a whole. We will see that to achieve this feat, the local rules we devise will be nonlinear in nature, comprising a hierarchical multi-layered control structure of decentralised estimation and decentralised optimisation, with feedback present in all scales across the network, both within each layer of control and also between each of the layers, and hence is clearly a complex and adaptive networked system.

Decentralised control is well suited to the control of such complex networked systems where there are many autonomous interacting parts, which through their interactions determine the behaviour of the collective. And through designing methods for the decentralised control of networks, we may perhaps achieve a deeper understanding of these dichotomies of part and whole, local and global, simple and complex, present in the study of complex systems.

1.2 Outline

In the next chapter, Chapter 2, we will introduce some useful notation and definitions, comprising the mathematical preliminaries from graph theory and spectral graph theory, which will be relied upon in subsequent chapters. Further to this, we will provide several interesting theorems and examples that demonstrate the relationship between properties of graphs and the spectra of their related matrices.

Chapter 3 reviews a number of recent papers on the control and optimisation of the eigenvalues of graph Laplacian matrices, looking both at centralised and decentralised strategies. Special attention is paid to two decentralised methods for agents in a network to estimate the algebraic connectivity and its associated eigenvector [29, 30], which are then compared.

Using one of these methods [29], we proceed in Chapter 4 to suggest two edge weight adaptation strategies which, when used in conjuction with the decentralised algebraic connectivity estimator, solve a constrained algebraic connectivity maximisation problem. The first strategy (presented in Section 4.2) is a primal interior point logarithmic barrier method first presented in [31]. The second method (Section 4.3) uses a sytem of coupled nonlinear differential equations, for which the stable equilibrium point solves the Karush-Kuhn-Tucker (KKT) first order necessary conditions, and this weight adaptation method is presented in [32] and utilised in [33]. Each of these adaptive edge weight control schemes can be used flexibly to account for different objective functions and sets of constraints, provided that these functions and their partial derivatives can themselves be estimated in a decentralised manner. In subsequent chapters, we focus on modifying and extending the decentralised algebraic connectivity estimator for the estimation of other interesting spectral functions of the graph Laplacian, so that we may solve a wide range of decentralised network optimisation problems.

In Chapter 5, we modify the algebraic connectivity estimator so that, instead, the largest eigenvalue of the graph Laplacian, its spectral radius and its associated eigenvector may be estimated. Utilising this spectral radius estimator in conjuction with the algebraic connectivity estimator, and a suitable weight adaptation strategy, allows the solution of a number of network optimisation problems. One such problem relevant to a number of engineering applications is the synchonisability maximisation problem, in which the ratio between the algebraic connectivity and spectral radius is maximised, facilitating synchronisation in networks of coupled oscillators. This modification to the algebraic connectivity estimator and the application to maximise the synchronisability of complex networks was presented in [34], using the logarithmic barrier weight adaptation strategy presented in Section 4.2. We also strive in this chapter to prove that the suggested multi-layer control scheme using the KKT satisfaction weight adaptation strategy (Section 4.3) is stable given sufficient separation in time-scale between the estimation and control layers, using methods from singular perturbation theory, and this proof of stability of the multi-layer system is published in [32].

Chapter 6 then introduces two further modifications to the algebraic connectivity estimator presented in Chapter 3. This time, changes are made so that the effective graph resistance and its partial derivatives, and the partial derivatives of the reduced graph Laplacian determinant may be estimated, and hence controlled in a decentralised manner, using the schemes presented in Chapter 4. These two spectral functions are closely related to random walks on networks: the effective graph resistance related to the expected commute time of a random walker, and the determinant of the reduced Laplacian being related to the number of spanning trees of a network. The ideas behind the estimator for the effective graph resistance, and its relation to the algebraic connectivity and spectral radius estimators were presented in [33]. The proof of convergence of the effective resistance estimator and the reduced Laplacian determinant estimator, and the connection between these two estimators presented in this Chapter is planned to be the subject of a future paper.

Finally, conclusions and suggestions for further work are made in Chapter 7. Figure 1.1, illustrates a schematic representation of the dependencies between chapters, with pointers to the publications which each of the main results chapters are based upon.

1.3 List of Publications

Part of the work presented in this thesis has been described in a number of publications, which we list below and provide a brief synopsis for. We also refer to which chapter of the thesis the publication relates.

- [31] Louis Kempton, Guido Herrmann, and Mario di Bernardo. “Adaptive weight selection for optimal consensus performance”. In: *53rd Annual Conference on Decision and Control (CDC)*. IEEE. 2014, pp. 2234–2239. DOI: 10.1109/CDC.2014.7039730

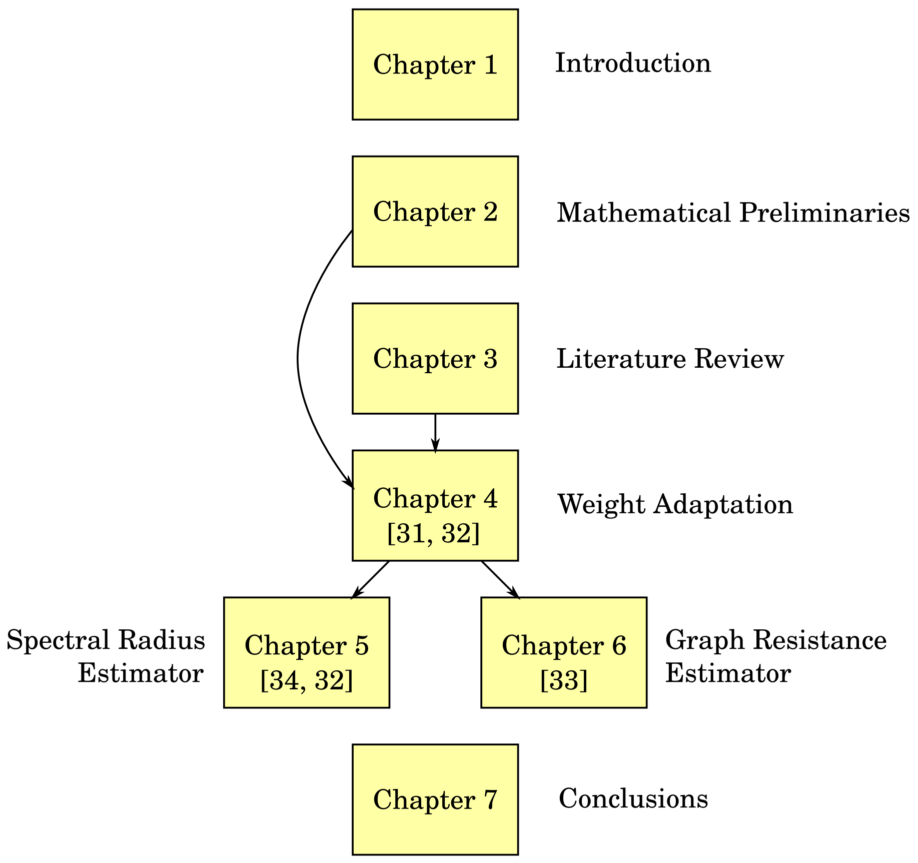


Figure 1.1: Notation and definitions are introduced in Chapter 2 and hence this chapter serves as a basis for subsequent chapters. Chapter 4 utilises the decentralised algebraic connectivity estimator from [29] and presented in Section 3.3.1 of the literature review. Both Chapters 5 and 6 use the weight adaptation strategies described in Chapter 4, but can be read independently of each other. Each of the main results chapters (4, 5, and 6) indicate which publications the results appear in.

This conference paper utilises the decentralised algebraic connectivity estimator of [29] in conjuction with a novel decentralised edge weight adaptation strategy to solve a constrained algebraic connectivity maximisation problem. This weight adaptation strategy utilises adaptive logarithmic barriers to enforce the constraints, and is described in Section 4.2 of Chapter 4. As such, Section 4.2 of Chapter 4 is based upon the paper [31] and draws material from it.

- [34] Louis Kempton, Guido Herrmann, and Mario di Bernardo. “Self-organization of weighted networks for optimal synchronizability”. In: *Transactions on Control of Network Systems* (2017). (Accepted). DOI: `10.1109/TCNS.2017.2732161`

  This paper presents the modification to the algebraic connectivity estimator [29] so that the spectral radius may also be estimated in a decentralised manner (this modification is presented in this thesis in Section 5.1). Utilising the adaptive logarithmic edge adaptation strategy (Section 4.2, [31]), the problem of maximising the synchronisability of complex networks is solved using a wholly decentralised approach.

- [33] Louis Kempton, Guido Herrmann, and Mario di Bernardo. "Distributed adaptive optimization and control of network structures". In: *55th Annual Conference on Decision and Control (CDC)*. IEEE. 2016, pp. 5839–5844. DOI: 10.1109/CDC.2016.7799167

  In this conference paper, we present the flexibility of the multi-layer decentralised estimation and adaptation strategies, by illustrating the modularity of the estimator and weight adaptation systems. This flexibility is demonstrated through solving a number of frequently studied network optimisation problems: maximising algebraic connectivity in a network of mobile robots, maximising the synchronisability of a network, and minimising the effective resistance of a network. We also introduce a decentralised estimator for the total effective graph resistance, which is presented more fully in Chapter 6 of this thesis.

- [32] Louis Kempton, Guido Herrmann, and Mario di Bernardo. "Distributed optimisation and control of graph Laplacian eigenvalues for robust consensus via an adaptive multi-layer strategy". In: *International Journal of Robust and Nonlinear Control* 27.9 (2017), pp. 1499–1525. DOI: 10.1002/rnc.3808

  In this paper, a decentralised strategy for minimising the spectral radius of the graph Laplacian, subject to the global connectivity constraint of maintaining a minimal value for the algebraic connectivity is presented. This improves robustness to time-delays in networks following a linear consensus protocol. The KKT satisfaction weight adaption strategy is utilised in this pursuit, and hence, Section 4.3 of Chapter 4 is based upon the weight adaptation section of this paper [32]. In the paper, the proof of stability of the multi-layer approach of decentralised estimation and adaptation is then given, provided a sufficient time-scale separation exists, and this proof is reiterated in Chapter 5, Sections 5.4 and 5.5 of this thesis.

Chapter 2

Mathematical Preliminaries

2.1 Graph Theory

The primary object of study in this thesis is the weighted, undirected graph. A graph describes a collection of objects, which are referred to as nodes[1], and maps pairwise interactions between them by ascribing edges[2] between two nodes if such an interaction exists. As such, graphs are frequently used to describe and model many multi-agent systems; systems where the entirety is composed of many interacting sub-units. Classic examples are social networks, where nodes represent individual people, and edges represent a relationship between people. Relationships may be, for example, friendship, sexual, or familial relationship.

Edges on graphs may be undirected, where the direction of interactions between agents does not matter or is symmetric. Take friendship in the previous example, if Alice is friends with Bob, we might expect Bob to be friends with Alice. Contrarily, direction may be important, as in the familial tree: Bob may sire Charlie, but certainly Charlie then does not father Bob! This is where we make the first distinction in categories of graph: if pairwise interactions are symmetric or direction is unimportant then we may model a multi-agent system using an undirected graph; if pairwise interactions are directional then it is more suitable to model the multi-agent system with a directed graph.

Definition 2.1 (Simple, undirected, unweighted graph)**.** A simple, undirected, unweighted graph $\mathcal{G}$, of n nodes and m edges is an ordered pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the vertex set $\mathcal{V} = \{1, \ldots, n\}$, and the edge set is a subset of all two element subsets (pairs, $\mathcal{P}$) of the vertex set: $\mathcal{E} \subseteq \{\mathcal{P} | \mathcal{P} \subseteq \mathcal{V}, |\mathcal{P}| = 2\}$.

[1]Also referred in literature as vertices.
[2]Also known as arcs or links.

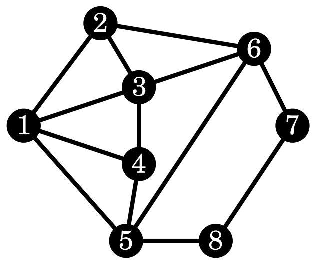


Figure 2.1: A simple, undirected, unweighted graph, $\mathcal{G}_1 = (\mathcal{V},\mathcal{E})$, on $n = 8$ nodes, and with $m = 13$ edges. This graph is defined by the vertex set $\mathcal{V} = \{1,2,3,4,5,6,7,8\}$, and edge set $\mathcal{E} = \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,6\},\{3,4\},\{3,6\},\{4,5\},\{5,6\},\{5,8\},\{6,7\},\{7,8\}\}$.

The number of nodes in the graph is denoted by $n(\mathcal{G}) \triangleq |\mathcal{V}|$, and the number of edges by $m(\mathcal{G}) \triangleq |\mathcal{E}|$. △

Note 2.1. *This definition of the edge set does not allow for self loops, or for multiple edges, so the graph is known as simple.*

The sole difference between a directed and an undirected graph is that edges in a directed graph have direction. That is, each edge has one node from which it originates, and one node to which it is destined. The origin and destination nodes may also be called, respectively, the parent and the child.

Definition 2.2 (Simple, directed, unweighted graph)**.** Keeping the same definition of a graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ of n nodes and m edges, with the vertex set defined as before: $\mathcal{V} = \{1,\dots,n\}$, we now define the edge set to be a subset of all ordered pairs of vertices: $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Note that so far this definition allows for self-loops in the edge set, so to restrict the definition to simple graphs, we need to add the following requirement: $(i,i) \notin \mathcal{E}, \forall i \in \mathcal{V}$. Convention is that the edge (i,j) indicates that there is an edge originating at node i and terminating at node j; j is then the child of i. △

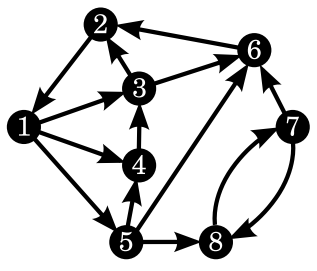


Figure 2.2: A simple, directed, and unweighted graph, $\mathcal{G}_2 = (\mathcal{V},\mathcal{E})$, on $n = 8$ nodes, and with $m = 14$ edges. The graph is defined by the vertex set $\mathcal{V} = \{1,2,3,4,5,6,7,8\}$, and edge set $\mathcal{E} = \{(1,3),(1,4),(1,5),(2,1),(3,2),(3,6),(4,3),(5,4),(5,6),(5,8),(6,2),(7,6),(7,8),(8,7)\}$.

2.1.1 Edge weighted graphs

Taking the definition of an unweighted graph given above, we can model the strength of interactions by labelling each edge with a real number. This formulation offers more flexibility in modelling. In the friendship example mentioned previously, we could only describe who was friends with whom, but using a weighted network we can also describe the strength of particular friendships, perhaps reflecting the amount of time that two people spend together. Such a refinement would clearly be beneficial for modelling the spread of a communicable disease on the network.

In this way, we define a simple, undirected, weighted graph in the following manner:

Definition 2.3 (Simple, undirected, weighted graph)**.** A simple, undirected, weighted graph $\mathscr{G}$ is defined by the ordered triple $\mathscr{G} = (\mathscr{V}, \mathscr{E}, w)$, with the vertex set $\mathscr{V}$ and edge set $\mathscr{E}$ defined in Definition 2.1. Now however, a function $w : \mathscr{E} \to \mathbb{R}$ is defined so that each edge is associated with a real number. For conciseness of notation, we shall denote the weight of the edge between nodes i and j, $w_{\{i,j\}} \triangleq w(\{i,j\})$. △

Likewise a weighted and directed graph can be defined by augmenting the pair $(\mathscr{V}, \mathscr{E})$, which defines the topology of the weighted graph, with an edge weight function $w : \mathscr{E} \to \mathbb{R}$. Furthermore, it is entirely reasonable to ascribe a weight function to the nodes as well, for both the directed and undirected case, and such a vertex weight is commonly called a mass. However, as we focus on edge weighted graphs in this thesis, a rigorous definition of vertex weighted graphs is not required.

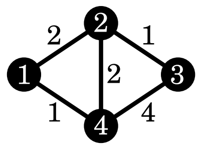


Figure 2.3: An example edge weighted, undirected graph. The graph $\mathscr{G}_3 = (\mathscr{V}, \mathscr{E}, w)$ is defined through the vertex set, $\mathscr{V} = \{1,2,3,4\}$, edge set, $\mathscr{E} = \{\{1,2\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}$, and the weight function: $w : \mathscr{E} \to \mathbb{R}, \quad \{1,2\} \mapsto 2,\ \{1,4\} \mapsto 1,\ \{2,3\} \mapsto 1,\ \{2,4\} \mapsto 2,\ \{3,4\} \mapsto 4.$

2.2 Matrix Representations of Graphs

Before introducing the spectra of graphs (of which there are several), it will be necessary to define the matrix representations of graphs to which these spectra belong. Matrices can succinctly describe the structure of the graphs which they represent, and can reveal many interesting properties of the graphs structure through linear algebra [35, 36, 37].

2.2.1 The Adjacency Matrix

The unweighted adjacency matrix is perhaps the simplest and most fundamental matrix representation of a graph, and has been used extensively since its introduction[3]. For undirected graphs, two nodes i and j are said to be adjacent if there exists an edge between them, and the adjacency matrix $\mathbf{A}_{n\times n}$ fully lists all possible pairs of nodes, and whether or not an edge exists between them. In the case that an edge exists between the nodes i and j a '1' is to be found in the (i,j) element of the adjacency matrix, else a '0' indicates that no such edge exists. As such, the unweighted adjacency matrix is both a square and Boolean matrix.

Definition 2.4 (Unweighted, undirected adjacency matrix)**.** The adjacency matrix is defined $\mathbf{A} = [a_{i,j}]_{n\times n}$ where,

$$a_{i,j} = \begin{cases} 1 & \text{if } \{i,j\} \in \mathscr{E} \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

△

As $\{i,j\} = \{j,i\}$ it is clear that the adjacency matrix is symmetric for undirected graphs. For simple graphs, i.e. $\{i,i\} \notin \mathscr{E}$, it is also evident that the main diagonal of the matrix consists entirely of zeros.

Example 2.1. *For the graph $\mathscr{G}_1$ illustrated in Figure 2.1, with vertex set $\mathscr{V} = \{1,2,3,4,5,6,7,8\}$, and edge set $\mathscr{E} = \{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,6\},\{3,4\},\{3,6\},\{4,5\},\{5,6\},\{5,8\},\{6,7\},\{7,8\}\}$, the adjacency matrix is defined according to Definition 2.4:*

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{2.2}$$

For directed graphs, the adjacency matrix need not be symmetric, and an entry in the (i,j) element indicates that there is an edge oriented from node i to node j. If indeed the adjacency matrix is symmetric then $(i,j) \in \mathscr{E} \iff (j,i) \in \mathscr{E}$, and the definition of a directed graph collapses onto the definition of an undirected graph.

[3]The earliest recorded use of the adjacency matrix I have found is in Shannon's 1956 paper "The zero error capacity of a noisy channel" [38], though the definition used here differs from the modern definition by the addition of the identity matrix.

Definition 2.5 (Unweighted, directed adjacency matrix)**.** The adjacency matrix of an unweighted and directed graph is defined $\mathbf{A} = [a_{i,j}]_{n\times n}$ where,

$$a_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in \mathscr{E} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

△

Weighted graphs can also be succinctly described by the adjacency matrix, simply by allocating the edge weight $w_{i,j}$ to the $(i,j)^{\text{th}}$ element:

Definition 2.6 (Weighted, directed adjacency matrix)**.** For a weighted and directed graph, the weighted adjacency matrix is defined $\mathbf{A} = [a_{i,j}]_{n\times n}$ where,

$$a_{i,j} = \begin{cases} w_{i,j} & \text{if } (i,j) \in \mathscr{E} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

△

From this definition, we see that an unweighted network may be treated as a special case of weighted network where all edges have unit weight, $w_{i,j} = 1$, $\forall (i,j) \in \mathscr{E}$. Moreover, if any edge weight in a weighted network is 0, this can be seen as being equivalent to removing that edge from the network.

Example 2.2. *For the graph $\mathscr{G}_3$ illustrated in Figure 2.3, the weighted adjacency matrix, according to Definition 2.6, is simply:*

$$\mathbf{A} = \begin{bmatrix} 0 & 2 & 0 & 1 \\ 2 & 0 & 1 & 2 \\ 0 & 1 & 0 & 4 \\ 1 & 2 & 4 & 0 \end{bmatrix} \tag{2.5}$$

As the graph $\mathscr{G}_3$, is undirected, its weighted adjacency matrix is symmetric.

2.2.2 The Incidence Matrix

The incidence matrix first appeared in the fundamental book on graph theory, König's 1936 text "Theorie der endlichen und unendlichen Graphen" [39]. Whereas the adjacency matrix indicates which nodes are adjacent to each other, the incidence matrix indicates which nodes are incident to each edge[4]. Unlike the adjacency matrix, the incidence matrix is not necessarily square[5], as it relates edges to nodes, and is instead an $m \times n$ matrix. In this thesis, we will be interested in two related incidence matrices, namely the unoriented and oriented incidence matrices, defined as follows:

[4]Somewhat unsurprisingly...

[5]Unless $n = m$.

Definition 2.7 (Unoriented incidence matrix)**.** Let $q : \{1,\dots,m\} \to \mathcal{E}$ be a bijective function from the set of integers from 1 to m, to the edge set, so that we may define an ordering on the edges. We may then define the unoriented incidence matrix $\mathbf{M} = [m_{i,j}]_{m\times n}$

$$m_{i,j} = \begin{cases} 1 \text{ if } j \in q(i) \\ 0 \text{ if } j \notin q(i) \end{cases} \tag{2.6}$$

△

Example 2.3. *For the graph $\mathcal{G}_1$ illustrated in Figure 2.1, with ordering on the edges defined by the function q,*

$$\begin{aligned} &q : \{1,\dots,m\} \to \mathcal{E} \\ &1 \mapsto \{1,2\},\quad 2 \mapsto \{1,3\},\quad 3 \mapsto \{1,4\},\quad 4 \mapsto \{1,5\},\quad 5 \mapsto \{2,3\}, \\ &6 \mapsto \{2,6\},\quad 7 \mapsto \{3,4\},\quad 8 \mapsto \{3,6\},\quad 9 \mapsto \{4,5\},\quad 10 \mapsto \{5,6\}, \\ &11 \mapsto \{5,8\},\quad 12 \mapsto \{6,7\},\quad 13 \mapsto \{7,8\}. \end{aligned} \tag{2.7}$$

the unoriented incidence matrix is defined according to Definition 2.7 as,

$$\mathbf{M} = \begin{bmatrix} 1&1&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0 \\ 1&0&0&1&0&0&0&0 \\ 1&0&0&0&1&0&0&0 \\ 0&1&1&0&0&0&0&0 \\ 0&1&0&0&0&1&0&0 \\ 0&0&1&1&0&0&0&0 \\ 0&0&1&0&0&1&0&0 \\ 0&0&0&1&1&0&0&0 \\ 0&0&0&0&1&1&0&0 \\ 0&0&0&0&1&0&0&1 \\ 0&0&0&0&0&1&1&0 \\ 0&0&0&0&0&0&1&1 \end{bmatrix} \tag{2.8}$$

Note that the choice of ordering on the edges is arbitrary, and corresponds to a permutation of rows in the incidence matrix. Likewise, permuting columns corresponds to a relabelling of nodes in a graph. Alternatively, it can be seen that an incidence matrix induces an ordering on an otherwise unlabelled graph.

We can see that the unoriented incidence matrix is a Boolean matrix and each row has exactly two non-zero elements, corresponding to the incident nodes of each edge. Row sums are equal to 2, and column sums are equal to the degree of each node. No distinction is made between the two incident nodes, so the unoriented incidence matrix is more suited to describing undirected graphs. Contrariwise, the oriented incidence matrix does make this distinction by denoting the origin of an edge with a -1 and the destination with the positive 1.

Definition 2.8 (Oriented incidence matrix (for directed graphs))**.** Again, let $q : \{1,\dots,m\} \to \mathscr{E}$ be a bijective function from the set of integers from 1 to m, to the edge set, defining an ordering on the edges, and let $q(i) = (a_i, b_i)$. We may then define the oriented incidence matrix $\mathbf{P} = [p_{i,j}]_{m\times n}$

$$p_{i,j} = \begin{cases} -1 & \text{if } j = a_i \\ +1 & \text{if } j = b_i \\ 0 & \text{if } j \notin q(i) \end{cases} \tag{2.9}$$

△

This matrix again has exactly two non-zero elements in each row, but now row sums are equal to zero; column sums are equal to the in-degree minus the out-degree.

Although the oriented incidence matrix is the natural choice for directed graphs, it is also useful in the analysis of undirected graphs. In this case, as both incident nodes are treated equally, it does not matter which is labelled with the -1 and which with the $+1$. As such, we arbitrarily choose to label the node with lower index with the negative value.

Definition 2.9 (Oriented incidence matrix (for undirected graphs))**.** Again letting $q : \{1,\dots,m\} \to \mathscr{E}$ be a bijective function from the set of integers from 1 to m, to the edge set, defining an ordering on the edges, we may then define the oriented incidence matrix $\mathbf{P} = [p_{i,j}]_{m\times n}$

$$p_{i,j} = \begin{cases} -1 \text{ if } j = \min(q(i)) \\ +1 \text{ if } j = \max(q(i)) \\ 0 \text{ if } j \notin q(i) \end{cases} \tag{2.10}$$

△

Example 2.4. *Again, using the undirected graph $\mathscr{G}_1$ and the same ordering on the edges as specified through the function q in Example 2.3, the oriented incidence matrix $\mathbf{P}$ is specified* *according to Definition 2.9,*

$$\mathbf{P} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix} \tag{2.11}$$

2.2.3 The Graph Laplacian

The weighted graph Laplacian, $\mathbf{L}$, is the primary matrix of interest in this thesis, and it connects the adjacency matrix and the incidence matrix through two equivalent definitions. Firstly, the graph Laplacian can be seen simply as the diagonal matrix of weighted degrees minus the adjacency matrix:

Definition 2.10. The out-degree graph Laplacian matrix for weighted and directed graphs $\mathbf{L} = [l_{i,j}]_{n\times n}$ is defined as:

$$l_{i,j} = \begin{cases} \sum_{k\in\mathcal{N}_i} w_{i,k} & \text{if } i = j \\ -w_{i,j} & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

Alternatively if we define the weighted out-degree matrix $\mathbf{D} = [d_{i,j}]_{n\times n}$ as the diagonal matrix of weighted out-degrees,

$$d_{i,j} = \begin{cases} \sum_{k\in\mathcal{N}_i} w_{i,k} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

then the out-degree graph Laplacian is simply given by:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2.14}$$

As such, it is clear that the out-degree graph Laplacian matrix always has zero row sum. If the graph is undirected, then the graph Laplacian is symmetric and also has zero column sum, as the out-degree is equal to the in-degree at each node. △

The graph Laplacian also admits another definition when the graph is symmetric, showing a connection to the oriented incidence matrix $\mathbf{P}$, defined previously in Definition 2.7:

Definition 2.11. Defining the vector of edge weights $\mathbf{w} = [w_i]_{m\times 1}$ using the ordering on the edges defined by the function q, as in Definition 2.7:

$$w_i \triangleq w_{q(i)} \tag{2.15}$$

and using the diag function which maps a vector to a diagonal matrix with elements of the vector on the main diagonal, in the same order,

$$\mathbf{L} = \mathbf{P}^\top \mathrm{diag}(\mathbf{w})\mathbf{P} \tag{2.16}$$

Furthermore, if the the network is unweighted (i.e. $\mathrm{diag}(\mathbf{w}) = \mathbf{I}$) then the above equation simplifies to:

$$\mathbf{L} = \mathbf{P}^\top \mathbf{P} \tag{2.17}$$

This view of the the graph Laplacian reveals a number of interesting properties. Specifically, it is immediately clear that the Laplacian matrix is positive semi-definite, $\mathbf{L} \succeq 0$, if the edge weights are non-negative. △

The graph Laplacian has the spectrum $\sigma_\mathbf{L} = \{\lambda_1, \lambda_2 \dots, \lambda_n\}$, and as stated previously, the graph Laplacian has zero row sum, i.e. $\mathbf{L1} = \mathbf{0}$, that is $\lambda_1 = 0$ is always an eigenvalue of the graph Laplacian. Moreover, in the symmetric case (for undirected graphs), the graph Laplacian has real eigenvalues and is positive semi-definite if edge weights are non-negative[6]. It can also be seen that the graph Laplacian is diagonally dominant (but not strictly diagonally dominant), so that we can bound the eigenvalues between zero and twice the maximum diagonal element (the maximum weighted degree, Δ) using Geršgorin's circle theorem. Thus we can order the eigenvalues of the graph Laplacian $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n \leq 2\Delta$. Many known results on the Laplacian spectrum, connecting the spectrum to properties such as diameter, degree distribution, isoperimetric number, can be found in the reviews [40, 41] and the textbooks [35, 36].

2.3 Spectral Graph Theory

Having defined several matrix representations of weighted and unweighted graphs, and their respective spectra, it will be useful to discuss a few select applications of these spectra, to provide some motivation for why these spectra are interesting, and why controlling or optimising the graph spectra may prove useful.

[6]Note that this a sufficient but not necessary condition; the graph Laplacian may still be positive semi-definite with some (but not all) negative edge weights.

2.3.1 Connectivity and Partitioning of Graphs

Firstly, we would like to present some results that connect the graph Laplacian and its spectrum to connectivity properties of the graph: specifically on the number of connected components, and measures of how easy it is to partition a graph into two disconnected subgraphs. In 1973, it was Fiedler [42] who first recognised the importance of the second smallest eigenvalue, λ_2, on connectivity properties of the graph, and named this eigenvalue *the algebraic connectivity* of the graph[7], thanks to its connection to the edge and vertex connectivities: respectively the number of edges or nodes that need to be removed to disconnect a graph.

Lemma 2.1. *In a connected component $\mathscr{G}$ of an undirected weighted graph, with Laplacian matrix $\mathbf{L}(\mathscr{G})$, if all edge weights are positive, the algebraic connectivity $\lambda_2(\mathbf{L}(\mathscr{G})) > 0$.*

Proof. [36] Let $\mathbf{x}$ be the eigenvector associated to the null eigenvalue. Thus,

$$\begin{aligned} \mathbf{Lx} &= \mathbf{0} \\ \mathbf{x}^\top \mathbf{Lx} &= \sum_{\{i,j\}\in\mathscr{E}} 2w_{i,j}(x_i - x_j)^2 = 0 \end{aligned}$$

If all $w_{i,j} > 0$, then $x_i = x_j$ if $\{i,j\} \in \mathscr{E}$. As $\mathscr{G}$ is connected then there exists a path between all pairs of vertices, and we can conclude that $\mathbf{x}$ is a vector where all components are equal, i.e. $\mathbf{x}$ is a scalar multiple of the vector of all ones, $\mathbf{x} = \alpha\mathbf{1}$, $\alpha \neq 0$. Hence the eigenspace associated with the zero eigenvalue has dimension 1, and as $\mathbf{L} \succeq 0$, we can conclude that all other eigenvalues are strictly positive, and specifically $\lambda_2 > 0$. □

The related Lemma for unweighted graphs was first proved in [42] using the Perron-Frobenius theorem, however, we choose to present a proof based upon Proposition 1.3.7 in [36], as it is simple to extend to positively weighted graphs.

Lemma 2.2. *The multiplicity of the zero eigenvalue is at least k in a graph $\mathscr{G}$ with k connected components.*

Proof. There exists a permutation matrix $\mathbf{P}$ so that,

$$\mathbf{P}^\top \mathbf{L}(\mathscr{G})\mathbf{P} = \begin{bmatrix} \mathbf{L}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{L}_k \end{bmatrix}$$

This transformation essentially corresponds to a relabelling of vertices so that if node i is in the p^{th} component and node j is in the q^{th} component, then $q > p \implies j > i$ for all i,j.

[7] A number of authors, however, call this eigenvalue *the Fiedler value*, and its associated eigenvector, *the Fiedler vector*.

Following from the zero row-sum property of Laplacian matrices we can immediately find that:

$$\begin{bmatrix} \mathbf{L}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} = \mathbf{0}, \quad \begin{bmatrix} \mathbf{L}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} = \mathbf{0}, \ldots, \quad \begin{bmatrix} \mathbf{L}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{L}_k \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} = \mathbf{0}$$

Hence, there are k linearly independent eigenvectors associated to the null eigenvalue. □

Lemma 2.3. *A weighted, undirected graph with positive edge weights and k connected components has precisely k zero eigenvalues, with all other eigenvalues being strictly positive.*

Proof. Proof follows directly from Lemmas 2.1 and 2.2. □

From these lemmas, it is clear that the algebraic connectivity λ_2 is central to determining the connectivity of a graph, and moreover, if we can guarantee that $\lambda_2 > 0$, then we guarantee that a network is connected; therefore information can spread across the entire network, which is for example essential for consensus and synchronisation problems, [43, 44, 45, 46].

Furthermore, not only does the algebraic connectivity determine connectivity, it is also a metric for measuring how strongly connected a graph is. In [42], Fiedler showed that the algebraic connectivity λ_2 was a lower bound on the more common notions of connectivity, the edge connectivity e (the minimum number of edges required to be removed to disconnect the graph), and the vertex connectivity v (the minimum number of nodes, with incident edges, required to be removed to disconnect the graph). For unweighted graphs it is shown in [42] that,

$$\lambda_2 \leq v \leq e \tag{2.18}$$

However, as shown in the following sections, the spectrum of the graph Laplacian reveals much more about partitioning of graphs, including those of weighted graphs.

2.3.2 Min & Max Cut

A partition of a graph's vertex set $\mathscr{P}(\mathcal{V})$ is a division of the vertex set into multiple sub-sets, so that the union of subsets covers the original vertex set, but the intersection of any two subsets is zero; i.e. all nodes in the vertex set belong to exactly one partition:

$$\begin{aligned} \mathscr{P}(\mathcal{V}) = \mathcal{V}_1, \ldots, \mathcal{V}_p : \quad & \mathcal{V}_1 \cup \cdots \cup \mathcal{V}_p = \mathcal{V} \\ & \mathcal{V}_a \cap \mathcal{V}_b = \emptyset \ \forall a \neq b \end{aligned} \tag{2.19}$$

The cost of a cut is typically defined as the sum of all edge weights in the cut set $\mathscr{C}$, where the cut set is defined:

$$\mathscr{C}(\mathscr{P}(\mathcal{V})) = \left\{ \{i,j\} \in \mathscr{E} : i \in \mathcal{V}_a, j \notin \mathcal{V}_a, \forall a \right\} \tag{2.20}$$

The cost of the cut is then:

$$\mathrm{cut}(\mathscr{P}(\mathscr{V})) = \sum_{\{i,j\}\in\mathscr{C}} w_{i,j} \tag{2.21}$$

A particularly useful inequality relating the eigenvalues of the Laplacian matrix to the cost of cuts on a graph is shown in the set of lecture notes [47][8], illustrating that the cost of any bipartition is bounded by the extremal non-trivial eigenvalues of the graph Laplacian, λ_2 and λ_n:

$$\frac{\lambda_2}{n} \le \frac{\mathrm{cut}(\mathscr{V}_1,\mathscr{V}_2)}{|\mathscr{V}_1||\mathscr{V}_2|} \le \frac{\lambda_n}{n} \tag{2.22}$$

For the minimum cut (when $|\mathscr{V}_1|$ may be as low as 1), the weighted edge connectivity (equivalent to the min-cut),

$$e(\mathscr{G}) \ge \frac{n-1}{n}\lambda_2 \tag{2.23}$$

which is a slightly weaker bound than Fiedler provided for unweighted graphs: $e(\mathscr{G}) \ge \lambda_2$ [42]. Contrarily, the maximum cut can only be achieved when $|\mathscr{V}_1| = |\mathscr{V}_2| = \frac{n}{2}$, a balanced bipartition:

$$\max\ \mathrm{cut}(\mathscr{G}) \le \frac{n\lambda_n}{4} \tag{2.24}$$

which was previously proven in [49]. We can see that by maximising the algebraic connectivity we can provide a larger guarantee on the amount of edge weight required to be removed to disconnect the graph. Finding the minimum or maximum cut is an NP-hard problem in general, so using optimisation of the eigenvalues of the graph Laplacian as a proxy for making the graph more resilient to disconnection is a sensible method.

Example 2.5. *Taking the weighted, undirected graph $\mathscr{G}_3$, depicted in Figure 2.3, the graph Laplacian matrix is:*

$$\mathbf{L} = \begin{bmatrix} 3 & -2 & 0 & -1 \\ -2 & 5 & -1 & -2 \\ 0 & -1 & 5 & -4 \\ -1 & -2 & -4 & 7 \end{bmatrix}$$

with the graph Laplacian spectrum found to be:

$$\sigma_{\mathbf{L}} = \{0, 3.0376\ldots, 6.6222\ldots, 10.0342\ldots\}$$

Specifically, the algebraic connectivity is found to be $\lambda_2 = 3.0376\ldots$, and the spectral radius of the graph Laplacian, $\lambda_n = 10.0342\ldots$. According to inequality (2.23)*, the minimum cut of this graph is at least:*

$$\begin{aligned} e(\mathscr{G}_3) &\ge \frac{n-1}{n}\lambda_2 \\ &\ge \frac{3}{4} \times 3.0376\cdots = 2.2782\ldots \end{aligned}$$

[8] This inequality stems from Theorem 2 in [48], which equates the Rayleigh coefficient of a partition vector to the cost of the associated cut, but is then relaxed for vectors which may contain elements other than +1 or −1.

Using the further information that all edge weights are integer, and so the cost of any cut is also integer, we can be sure that the minimum cut in this graph is at least 3. *In fact, it can be quickly seen that the degree of node* 1 *is* 3*, and so the cut that isolates this node is a minimum cut.*

Furthermore, a lower bound on a balanced cut, when the two partitions are each of two nodes ($|\mathcal{V}_1| = |\mathcal{V}_2| = 2$*), can be inferred from the algebraic connectivity, using the inequality 2.22:*

$$\begin{aligned}\frac{\lambda_2}{4} &\leq \frac{cut(\mathcal{V}_1,\mathcal{V}_2)}{2\times 2}\\ cut(\mathcal{V}_1,\mathcal{V}_2) &\geq \lambda_2\end{aligned}$$

Again, using the fact that all edges have integer weight, then we can be sure that the minimum balnced cut has a cost of at least 4*. Such a bipartition is the cut that separates* $\mathcal{V}_1 = \{1,2\}$ *from* $\mathcal{V}_2 = \{3,4\}$*. The edges in the cut set have a combined weight of* $w_{\{1,4\}} + w_{\{2,4\}} + w_{\{2,3\}} = 1+2+1 = 4$.

Finally, an upper bound on the maximum cut can be deduced using 2.24

$$\begin{aligned}max\text{-}cut(\mathcal{G}_3) &\leq \frac{n\lambda_n}{4}\\ &\leq \lambda_n = 10.0342\ldots\end{aligned}$$

Again, the range can be further narrowed to max-cut($\mathcal{G}_3$*)* ≤ 10*, but for this particular graph, the bound is not tight: the maximum cut is found to be that which divides* $\mathcal{V}_1 = \{1,4\}$ *from* $\mathcal{V}_2 = \{2,3\}$*, with a cost of* 8.

2.3.3 Consensus

Of particular interest to this thesis is the problem of achieving consensus in a network of dynamical agents [50, 45, 51, 52, 53]. Given a network of interconnected agents, it is often required that all agents are able to reach a collective decision, and that every agent will agree with every other in the network. Such a task is important, for instance, in the applications of distributed averaging in wireless sensor networks [54], clock synchronisation in wireless networks [55, 56], resource allocation for parallel computing [57], energy management in the smart grid [2, 58], and formation control in swarms of mobile robots [59, 60, 61].

In general, consensus is achieved in a network of n nodes when each agent's local state variable x_i is equal to every other in the network: $x_1 = x_2 = \cdots = x_n$. However, as a specific case, we can consider the scenario when state variables take a real number, $x_i \in \mathbb{R}$. Then, one of the simplest consensus protocols[9][45] is given by the set of ordinary differential equations:

$$\dot{x}_i = \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(x_j - x_i) \tag{2.25}$$

That is, each agent only updates its local state in response to the (possibly weighted) sum of differences between its state and those of its neighbours. The dynamics of the collective can be

[9]Often referred to as *simple linear consensus*, or *linear diffusive coupling*.

more concisely described using the (weighted) graph Laplacian matrix:

$$\dot{\mathbf{x}} = -\mathbf{L}\mathbf{x} \tag{2.26}$$

where the vector $\mathbf{x}$ is the vector of state variables $[x_1, x_2, \ldots, x_n]^\top$.

Lemma 2.4. *In the case when the communication graph is undirected (*$\mathbf{L}$ *is symmetric), consensus is achieved if and only if the graph is connected,* $\lambda_2 > 0$. *Moreover, consensus is reached exponentially fast with the rate of convergence governed by the algebraic connectivity,* λ_2, *and the consensus value reached is the arithmetic mean of the initial condition.*

Proof. As $\mathbf{L}$ is symmetric, there exists a unitary matrix $\mathbf{V}$ which diagonalises $\mathbf{L}$. Let $\mathbf{\Lambda} = \mathbf{V}^\top\mathbf{L}\mathbf{V}$, where $\mathbf{\Lambda} = \text{diag}(0, \lambda_2, \ldots, \lambda_n)$, and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$. Then, applying the transformation $\mathbf{x} = \mathbf{V}\mathbf{z}$:

$$\begin{aligned} \dot{\mathbf{x}} &= -\mathbf{L}\mathbf{x} \\ \mathbf{V}\dot{\mathbf{z}} &= -\mathbf{L}\mathbf{V}\mathbf{z} \\ \mathbf{V}^\top\mathbf{V}\dot{\mathbf{z}} &= -\mathbf{V}^\top\mathbf{L}\mathbf{V}\mathbf{z} \\ \dot{\mathbf{z}} &= -\mathbf{\Lambda}\mathbf{z} \end{aligned} \tag{2.27}$$

It can be seen that, through diagonalisation, the modes have been decoupled. The consensus mode is constant, $\dot{z}_1 = 0$, where all other modes decay exponentially fast, $\dot{z}_i = -\lambda_i z_i$ for $i = 2, \ldots, n$. The slowest decaying of these modes is the smallest non-zero eigenvalue of the graph Laplacian, λ_2, and is the limiting rate of convergence to the consensus mode. Furthermore, the consensus mode, which is constant $z_1 = \mathbf{v_1}^\top\mathbf{x}$, where $\mathbf{v_1}$ is the right column eigenvector associated with $\lambda_1 = 0$, $\mathbf{v_1} = \frac{1}{\sqrt{n}}\mathbf{1}$, shows that the sum of all initial states is an invariant quantity. Asymptotically, when all other modes except the consensus mode have decayed, $\mathbf{z}^* = [\frac{1}{\sqrt{n}}\mathbf{1}\mathbf{x}(t=0), 0, \ldots, 0]^\top$. Reversing the transformation to the diagonal basis we find that the equilibrium point $\mathbf{x}^* = \mathbf{V}\mathbf{z} = \frac{\mathbf{1}\mathbf{1}^\top\mathbf{x}(t=0)}{n}$; each node's state has converged on the arithmetic mean of the networks intial states. □

Example 2.6. *The simple linear consensus protocol, as described by Equation* (2.25)*, is run on the network* $\mathscr{G}_1$*, illustrated in Figure 2.1. Initial states are drawn uniformly at random from the interval* $[-5, 5)$*, the mean of the initial states being 0.2284.... This mean is the consensus value and indicated by the red line in Figure 2.4. The continuous time equations have been discretised using the forward Euler method, with a time step of* 1×10^{-3}. *It can be seen that consensus is achieved rapidly in the network. Further, by examining the error from the consensus value,* $\mathbf{y} = \mathbf{x} - \frac{\mathbf{1}\mathbf{1}^\top\mathbf{x}(0)}{n}$, *and plotting the natural log of the Euclidean norm of this error against time (Figure 2.5), the rate of convergence can be graphically illustrated. For the unweighted network* $\mathbf{G}_1$ *, the algebraic connectivity is exactly equal to* 1*, and this corresponds to the gradient of the log error norm in Figure 2.5.*

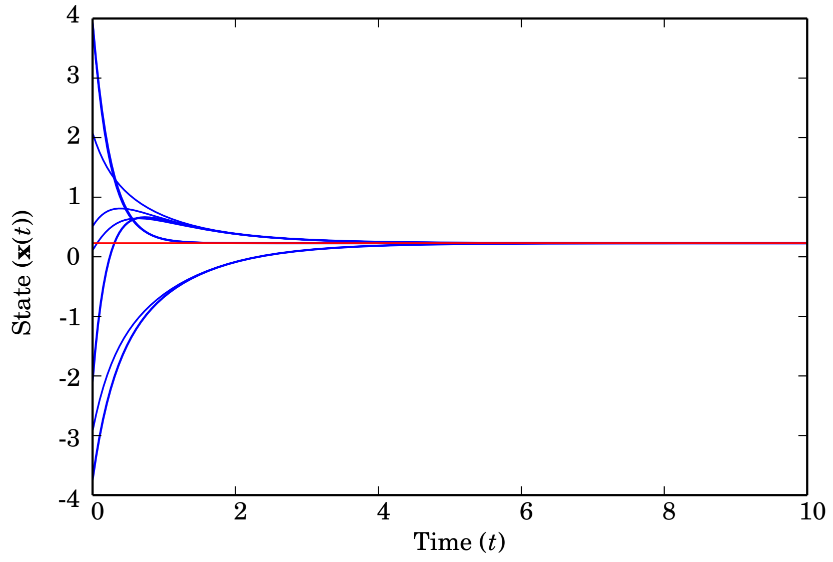


Figure 2.4: An illustration of the simple linear consensus protocol, Equation (2.25), instantiated on the example graph $\mathscr{G}_1$, with initial values drawn uniformly at random from the interval $[-5,5)$. In this figure, the arithmetic mean of the initial states is designated by the red line.

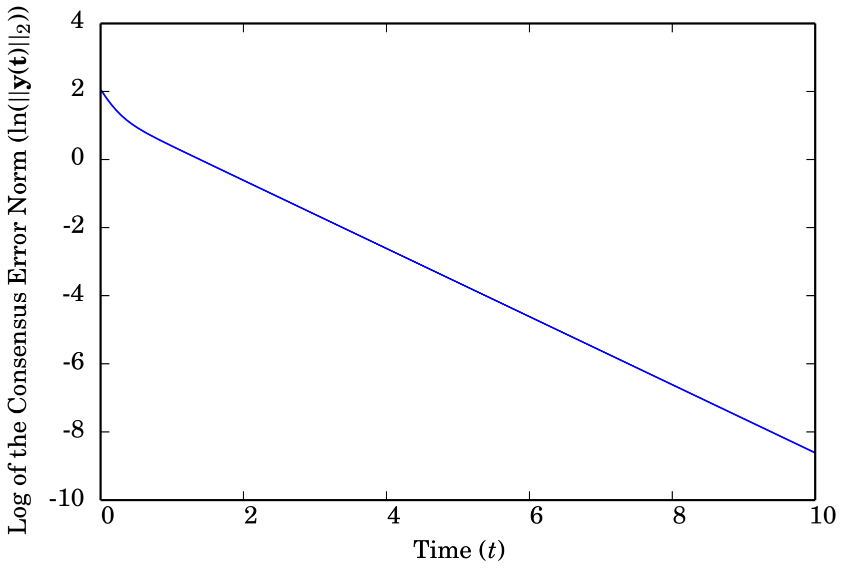


Figure 2.5: The algebraic connectivity of the graph $\mathscr{G}_1$, on which the simple linear consensus protocol is run, is $\lambda_2 = 1$. It can be seen graphically that this is the quantity determining the rate at which consensus is reached, through observing the gradient of the natural logarithm of the norm of the discrepancy from the consensus value, $\ln(||\mathbf{y}||_2)$, where $\mathbf{y}$ is the discrepancy $\mathbf{y} = \mathbf{x} - \frac{\mathbf{1}\mathbf{1}^\top \mathbf{x}(0)}{n}$.

2.3.4 Synchronisation

Synchronisation can be seen as a more general phenomenon than consensus [52, 62, 63, 64]. However, rather than agents in a network coming to agreement on a particular equilibrium value, agents in a network of nonlinear dynamical systems interact with each other to share a similar pattern of behaviour. In the case of full state synchronisation, each agent's state vector $\mathbf{x}_i(t)$ is equal to all others in the network for all time, $\mathbf{x}_1(t) = \mathbf{x}_2(t) = \cdots = \mathbf{x}_n(t)$; and all systems follow the same trajectory. The set of $n-1$ constraints $\mathbf{x}_1 = \mathbf{x}_2 = \cdots = \mathbf{x}_n$ define the synchronisation manifold [43].

In the case of chaotic oscillators, any small deviation from the synchronisation manifold will grow over time, and this growth can be measured by the maximum Lyapunov exponent of the system, which is positive for chaotic systems [65]. However, with suitable coupling strategies, the synchronous solution can be made to be locally transversally stable; the maximum Lyapunov exponent of the equation describing the dynamics of the errors between nearby trajectories is then negative. One of the fundamental tools in studying the local stability of the synchronous solution in a network of chaotic oscillators is the *Master Stability Function (MSF)* [43, 44] which tracks the maximum Lyapunov exponent of the system as the coupling strength between subsytems is increased.

We can consider the generic homogeneous networked system of n identical agents:

$$\dot{\mathbf{x}}_i = \mathbf{F}(\mathbf{x}_i) - \sigma \sum_{j=1}^{n} l_{i,j}\mathbf{H}(\mathbf{x}_j) \tag{2.28}$$

where $\mathbf{x}_i = [x_j]_{d\times 1}$ is the d-dimensional state vector of the i^{th} subsystem, and $\mathbf{F}(\mathbf{x}_i) : \mathbb{R}^d \mapsto \mathbb{R}^d$ is the vector field of the i^{th} isolated system. The global coupling strength between subsystems is controlled by the scalar parameter σ, and $\mathbf{H}(\mathbf{x}_j) : \mathbb{R}^d \mapsto \mathbb{R}^d$ is the coupling function between nodes. The topology and local strength of the pairwise interactions are determined through the weighted graph Laplacian matrix $\mathbf{L} = [l_{i,j}]_{n\times n}$. Evidently[10], the synchronous solution, $\mathbf{x}_1(t) = \mathbf{x}_2(t) = \cdots = \mathbf{x}_n(t) = \mathbf{s}(t)$, where,

$$\dot{\mathbf{s}} = \mathbf{F}(\mathbf{s}) \tag{2.29}$$

is a solution to (2.28). The Master Stability Function is then found by plotting the maximum Lyapunov exponent (the so-called Largest Lyapunov Exponent (LLE)) of the block diagonalised variational equation, over a range of coupling strengths, as described in Sections 2 and 3 of [13], which we briefly reiterate here for completeness. The variational equation gives the dynamics of a small virtual displacement in the vicinity of the synchronous solution. Defining the virtual displacement $\delta\mathbf{x}_i(t) = \mathbf{x}_i(t) - \mathbf{s}(t)$, its linearised dynamics are found to be:

$$\frac{d\delta\mathbf{x}_i}{dt} = \mathbf{DF}(\mathbf{s})\delta\mathbf{x}_i - \sigma \sum_{j=1}^{n} l_{i,j}\mathbf{DH}(\mathbf{s})\delta\mathbf{x}_j \tag{2.30}$$

[10]This follows from the zero row sum property of the Graph Laplacian matrix, $\mathbf{L}$.

where the matrices **DF** and **DH** are the Jacobians of the vector field of the isolated system **F**(**x**) and coupling function **H**(**x**) respectively, evaluated on the synchronous solution **s**(t). The modes of this variational equation can then be block decoupled through the following linear transformation:

$$\begin{bmatrix} \delta\mathbf{y}_1^\top \\ \delta\mathbf{y}_2^\top \\ \vdots \\ \delta\mathbf{y}_n^\top \end{bmatrix} = \mathbf{V}^\top \begin{bmatrix} \delta\mathbf{x}_1^\top \\ \delta\mathbf{x}_2^\top \\ \vdots \\ \delta\mathbf{x}_n^\top \end{bmatrix} \tag{2.31}$$

where **V** is the $n \times n$ unitary matrix of right column eigenvectors of **L**, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$ such that $\mathbf{\Lambda} = \mathbf{V}^\top \mathbf{L}\mathbf{V}$. Each block of the block diagonalised variational equation is then given by,

$$\frac{d\delta\mathbf{y}_i}{dt} = (\mathbf{DF}(\mathbf{s}) - \sigma\lambda_i\mathbf{DH}(\mathbf{s}))\delta\mathbf{y}_i \tag{2.32}$$

with λ_i being the eigenvalues of **L**. It can then be seen that each mode $\delta\mathbf{y}_i$ follows similarly structured dynamics, with the strength of the coupling term being proportional to the mode's associated eigenvalue, λ_i. By using a dummy variable α, as in (2.33), to substitute for the quantity $\sigma\lambda_i$, the transversal stability of the synchronous solution can be characterised of a range of possible coupling strengths and Laplacian eigenvalues.

$$\frac{d\delta\mathbf{y}}{dt} = (\mathbf{DF}(\mathbf{s}) - \alpha\mathbf{DH}(\mathbf{s}))\delta\mathbf{y} \tag{2.33}$$

The MSF $\Psi(\alpha)$ achieves this by plotting the Largest Lyapunov Exponent (LLE) of the system governed by Equation (2.33), as a function of the coupling strength α. In intervals where α is such that the LLE of Equation (2.33) is negative (the MSF $\Psi(\alpha)$ is negative), any small distance between trajectories near to the synchronous solution will decay away, and thus the synchronous solution is locally transversally stable. For more details on the MSF, or how the LLE is calculated, see for example [43, 13].

Example 2.7. *As a representative example, we consider a network of coupled Rössler oscillators with parameters as in [13]. In the fashion of Equation* (2.28)*, the isolated system is given by*

$$\mathbf{x} \triangleq [x, y, z]^\top$$

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}) \triangleq \begin{bmatrix} -y - z \\ x + \zeta y \\ \beta + z(x - \gamma) \end{bmatrix}, \qquad \zeta = 0.2, \beta = 0.2, \gamma = 9$$

A solution of the uncoupled oscillator is presented below, in Figure 2.6. We then investigate two different coupling regimes:

x-diffusive coupling:

$$\mathbf{H}(\mathbf{x}) \triangleq [x, 0, 0]^\top$$

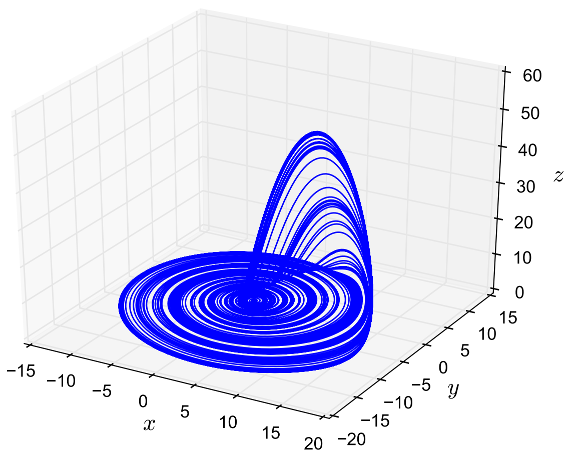


Figure 2.6: A phase-space depiction of the chaotic Rössler attractor.

y-diffusive coupling:

$$\mathbf{H}(\mathbf{x}) \triangleq [0, y, 0]^\top$$

each with its own MSF $\Psi(\alpha)$, Figure 2.7, where α is a dummy variable representing $\sigma\lambda_i$.

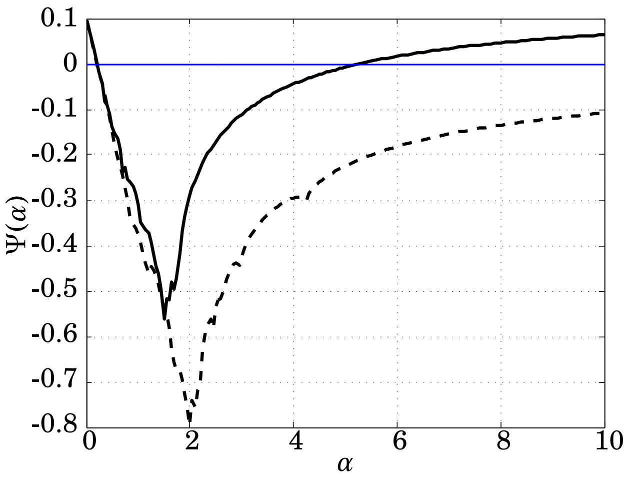


Figure 2.7: The MSFs $\Psi(\alpha)$ for the Rössler oscillator with two different couplings are shown, showing class Γ_1 (y-diffusive coupling, dashed line) and Γ_2 (x-diffusive coupling, solid line) characteristic shapes. The negative interval for the x-diffusive coupling is found to be $(0.187, 4.615)$ to 3 decimal places, and the negative interval for the y-diffusive coupling is found to be $(0.156, \infty)$ to 3 decimal places. This is in close agreement to the results published in [13].

In general, for a given MSF $\Psi(\alpha)$, intervals in the scalar argument α for which its value is negative fall into two categories: right-unbounded $\Psi(\alpha) < 0$ for $\alpha \in (\alpha_1, \infty)$, or proper bounded $\Psi(\alpha) < 0$ for $\alpha \in (\alpha_1, \alpha_2)$. Using the classification system of Huang et al. [13], those MSFs which contain only one negative interval and that interval is right-unbounded are called Γ_1, and those for which their sole negative interval is proper bounded are called Γ_2. These two types of negative interval lead to two ideas of synchronisability. For systems with a Γ_1 MSF, the synchronous solution is stable if $\sigma\lambda_2(\mathbf{L}) > \alpha_1$ (this is seen to be the case for the y-diffusive coupling of the Rössler oscillators in Example 2.7). That is, networks with greater algebraic connectivity λ_2 require a lower global coupling strength, and are thus more easily synchronized; less control effort is required.

On the other hand, for systems with a Γ_2 MSF, the synchronous solution can only be stable if $\alpha_1 < \sigma\lambda_2 \le \sigma\lambda_n < \alpha_2$ (as seen to be the case for the x-diffusive coupling of the Rössler oscillators in Example 2.7). Thus, such systems can only permit a locally transversally stable synchronous solution if $\lambda_n/\lambda_2 < \alpha_2/\alpha_1$. Graphs with lower eigenratio λ_n/λ_2 are then deemed more synchronizable [44], as a greater range of global coupling strengths will allow a stable synchronous solution. Thus, depending on the shape of the specific MSF for a network of coupled chaotic oscillators, one of these two spectral functions of the graph Laplacian, the algebraic connectivity or spectral radius, determines its synchronisability.

Example 2.8. *To show the effect of network structure on synchronisation, we compare two networks of y-diffusively coupled Rössler oscillators, with parameters as described in Example 2.7. The continuous time dynamics were discretised for the purposes of simulation using the forward Euler method with a time step of* 1×10^{-2}. *To show the large impact that network structure can have on the graph Laplacian eigenvalues, we choose two network topologies, both with* $n = 10$ *nodes, and* $m = 15$ *edges, and moreover both having identical degree distributions*[11]. *The two network topologies chosen are shown in Figure 2.8, with their respective Laplacian spectra.*

From the MSF computed in Example 2.7, we would expect the synchronisation manifold to be locally transversally stable if $\sigma\lambda_2 > 0.156$. *As such, we pick* $\sigma = 0.25$, *arbitrarily, giving a value for* $\alpha = 0.5 > 0.156$ *for the Petersen graph, and* $\alpha = 0.05538 < 0.156$ *for the second graph with lower algebraic connectivity.*

Starting with initial condition $\mathbf{x}_i = [X \sim \mathcal{N}(5, 0.1^2), 0, 0]$ *(X is a random variable drawn from the normal distribution with mean* 5 *and standard deviation of* 0.1*), the network of coupled Rössler oscillators is simulated on both graphs.*

To measure the level of synchronisation, we choose to observe the log[12] *of the mean Euclidean distance from the centroid of all oscillator states. The centroid of all oscillator states is given by* $\boldsymbol{\mu} = \frac{1}{n}\sum_i \mathbf{x}_i$, *and the synchronisation level is given by:*

$$\left[\log_{10}\left(\tfrac{1}{n}\textstyle\sum_i \sqrt{(\mathbf{x}_i - \boldsymbol{\mu})^\top(\mathbf{x}_i - \boldsymbol{\mu})}\right)\right] \tag{2.34}$$

[11]In fact, both graphs are regular, with a valency of 3.

[12]We choose to plot the log of the mean distance as straight lines will then correspond to exponential increase or decrease in the mean distance.

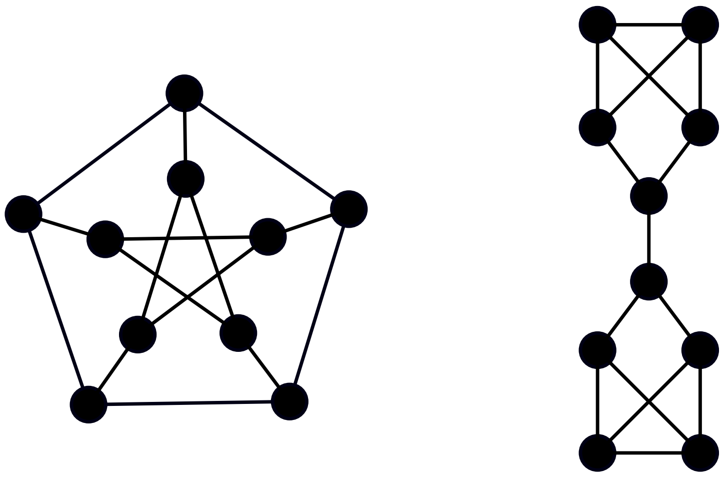


Figure 2.8: The Petersen graph (left) has Laplacian spectrum 0,2,2,2,2,2,5,5,5,5; its algebraic connectivity is 2. The other 3-regular graph on 10 nodes (right) has Laplacian spectrum 0,0.2215,2,3,3,3.2892,4,4,5,5.4893 with non-integer values rounded to 4 decimal places; its algebraic connectivity is approximately 0.2215.

For the Petersen graph we see that the synchronous solution is locally transversally stable, as the logarithm of the mean Euclidean distance to the centroid decays quickly, and approximately exponentially.

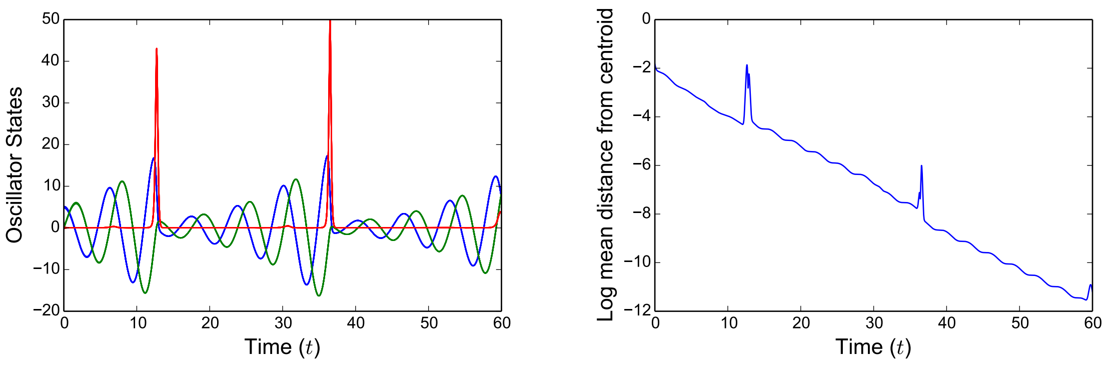


Figure 2.9: We see that the synchronous solution is maintained (left) over the 60 second simulation time. Each oscillator state is plotted in a separate colour (x - blue, y - green, z - red). We can see in the rightmost pane that the mean Euclidean distance to the centroid of the oscillators' states decreases over time asymptoically approaching zero, and for the mostpart, this decrease appears to be exponentially fast. This is also apparent from the negative LLE evaluated on the synchronous solution, at this coupling strength.

For the second graph with lower algebraic connectivity, the variance in state grows over time, and synchronisation is quickly lost.

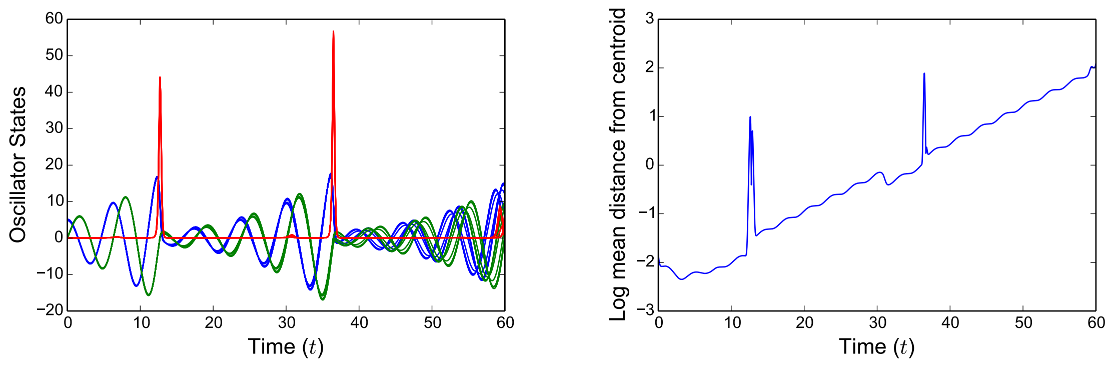


Figure 2.10: When the simulation is run on the second network topology (with lower algebraic connectivity) it can be seen that synchronisation is lost, and the synchronisation manifold is unstable. The mean Euclidean distance to the centroid of the system's states grows over time, as seen in the rightmost plot.

2.4 Summary

In this chapter, we first introduced some important definitions and notations relating to directed and undirected, weighted and unweighted graphs. From this, we introduced three important matrix representations of a graph: the adjacency matrix, the incidence matrix, and the Laplacian matrix, and stated some important properties of these matrices. Finally, we looked at some motivating results from spectral graph theory, particularly in how the eigenvalues of the Laplacian matrix relate to properties of the graph: namely relating to connectivity, the cost of partitioning the graph, the speed of consensus on the graph, and the synchronisability of complex networks. In the next Chapter, we will explore some of the approaches taken to optimise and control the spectra of graphs, particularly that of the graph Laplacian matrix, using both centralised and decentralised approaches.

2.5 Table of Notation

Symbol	**Referring to**
Capital, bold letters	Matrices
$\mathbf{I}$	The Identity Matrix
$\mathbf{L}$	The Graph Laplacian Matrix
$\mathbf{A}$	The Adjacency Matrix
$\mathbf{M}$	The Unoriented Incidence Matrix
$\mathbf{P}$	The Oriented Incidence Matrix
$\mathbf{V}$	The matrix of right column eigenvectors of $\mathbf{L}$. $\mathbf{V} = [\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}]$
Lowercase, bold letters	Vectors
$\mathbf{1}$	The vector of all ones.
$\mathbf{0}$	The vector of all zeroes. This may also refer to a matrix of zeroes. If dimensions are unclear, they will be indicated with subscripts.
$\mathbf{v_i}$	The lowercase, bold, $\mathbf{v}$ refers to the right (column) eigenvectors of the matrix $\mathbf{L}$. Bold subscripts are used to indicate the associated eigenvalue, and as such $\mathbf{v_2}$ is the eigenvector associated with the algebraic connectivity λ_2 (commonly known as the Fiedler vector). When referring to a single element of an eigenvector, we use the lowercase $v_{i,j}$ to refer to the j^{th} element of $\mathbf{v_i}$.
$\mathbf{u_i}$	The left (row) eigenvectors of the Graph Laplacian matrix $\mathbf{L}$.
Lowercase, non-bold letters	Scalar quantities
λ_2	The second smallest eigenvalue of the Graph Laplacian matrix $\mathbf{L}$. This quantity is commonly known as the algebraic connectivity of a graph, or the Fiedler eigenvalue, and is referred to in this thesis as the algebraic connectivity.
λ_n	The largest eigenvalue of the Graph Laplacian matrix $\mathbf{L}$; its spectral radius. This quantity is referred to as the spectral radius of $\mathbf{L}$, or often, when the meaning is clear, simply, the spectral radius.
λ_i	The i^{th} eigenvalue of $\mathbf{L}$.
i, j	i and j are used as indexes, especially when referring to elements of a vector or matrix. a_i is the i^{th} element of the vector $\mathbf{a}$. $b_{i,j}$ is likewise the i,j^{th} element of the matrix $\mathbf{B}$.
m	The number of edges in a graph.
n	The number of nodes in a graph.
k_i	The weighted degree of node i

Calligraphic, capital letters	Sets, especially relating to graphs
$\mathscr{G}$	A graph. Graphs may be directed or undirected, weighted or unweighted, see Definitions 2.1 to 2.3.
$\mathscr{V}$	The vertex set of a graph.
$\mathscr{E}$	The edge set of a graph.
	Functions
$w(\{i,j\})$	The weight function, that assigns a numeric value to the undirected edge $\{i,j\}$ or directed edge (i,j), typically a real scalar. For convenience, the weight of the edge $\{i,j\}$ is often written in the more compact form $w_{\{i,j\}}$. See Definition 2.3.
$q(\{i,j\})$	A bijective function that maps the edge set to the set of integers $[1,\ldots,m]$. In this way, we can order the edge weights of a weighted graph, and construct an edge weight vector $\mathbf{w}$.
$\text{diag}(\mathbf{x})$	The diag function maps the elements of a vector (in this case $\mathbf{x}$) to the elements of a diagonal matrix, in the same order.
$\mathscr{P}(\mathscr{V})$	A partition of the vertex set.
	Other symbols
$\langle\mathbf{x}\rangle$	The arithmetic mean of the vector $\mathbf{x}$.
$\prec$, $\succ$, $\preceq$, $\succeq$	Matrix inequality symbols. The relation $\mathbf{A} \succ \mathbf{B}$ is equivalent to $\mathbf{A}-\mathbf{B} \succ 0$; hence $\mathbf{A}-\mathbf{B}$ is positive definite. Similarly, $\mathbf{A} \preceq 0$ is the statement that the matrix $\mathbf{A}$ is negative semidefinite.
$\mathscr{N}(\mu,\sigma^2)$	The normal distribution with mean μ and variance σ^2. The notation $X \sim \mathscr{N}(\mu,\sigma^2)$ signifies a random variable X drawn from the normal distribution $\mathscr{N}(\mu,\sigma^2)$.
$\dot{\mathbf{x}}$	The time derivative of the vector $\mathbf{x}$, $\frac{d\mathbf{x}}{dt}$

Chapter 3

Optimisation and Control of Graph Eigenvalues

In the previous Chapter, we saw several matrices that can be induced from a network, including both the Adjacency matrix and the Laplacian matrix. A number of useful properties of the spectra of these matrices were demonstrated, especially relating to the connectivity properties of the network. The uses of these spectra are numerous, from spectral partitioning [66, 49, 67, 68], allocation of computing resources [69], bounding graph colouring problems [70], bounding travelling salesman problems [71, 72], and many more. We refer the reader to the following textbooks [66, 35] for a wider review on the applications of graph spectra.

As these spectra are so useful for bounding bulk properties of networks, or estimating the performance of networks, it is often expedient to design networks with desirable spectra. In practice, this often boils down to finding graphs which maximise some function of the spectra, subject to constraints on which graphs are allowed. Such graphs are often called optimal graphs, and can be found using numerical methods as we will explore later in this Chapter, or alternatively for specific sets of constraints, e.g. strongly regular graphs, families of optimal or near-optimal graphs may be found using extremal graph theory [19, 73, 74].

In this Chapter, we will review some results for finding both weighted and unweighted graphs with desirable eigenvalues, starting from closed form solutions for specific classes of graphs, to centralised optimisation methods for more general graphs (predominantly semi-definite programming), and finally on to decentralised strategies for the estimation and control of graph spectra, where agents in a network may communicate only with their neighbours and cooperate to infer and adapt the desired properties of the network. We pay special attention to the method presented in [29] for the decentralised estimation and control of the algebraic connectivity of a network, a continuous time formulation based upon power iteration, as we use this method extensively throughout this thesis, and develop it in subsequent chapters for estimation and optimisation of other spectral functions of the graph Laplacian.

3.1 Closed form solutions

As mentioned in Chapter 2, it was Miroslav Fielder who in 1973 first connected the second smallest eigenvalue of the graph Laplacian to connectivity properties of the graph [42], and it is he in [75] who is responsible for finding one of the few closed form solutions for a maximal spectral function of a graph. In [75], Fiedler addresses the problem of finding the maximum algebraic connectivity of a weighted, undirected graph subject to the constraint that the sum of edge weights is equal to the number of edges in the graph (alternatively that the mean edge weight is exactly 1). Fiedler calls this value the *absolute algebraic connectivity* of a graph, and provides a closed form solution for the absolute algebraic connectivity when the graph is a tree.

Expressing this problem in the standard form of an optimisation problem, Fiedler finds the maximal value of λ_2 for the tree, subject to the constraint that all weights in the tree sum to the number of edges:

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximise}} \quad & \lambda_2\big(\mathcal{G}(\mathcal{V},\mathcal{E},\mathbf{w})\big) \\ \text{subject to} \quad & \mathbf{w}^{\mathrm{T}}\mathbf{1} = |\mathcal{E}| = m \end{aligned} \tag{3.1}$$

where the weighted, undirected graph $\mathcal{G}(\mathcal{V},\mathcal{E},\mathbf{w})$ is a tree.

In fact, it is proven in [75] that the absolute algebraic connectivity of a tree is exactly the reciprocal of the variance of the tree (see [75, 41] for the definition of the variance of a tree). Unfortunately, however, it is unknown whether there exists a closed form solution for finding the edge weights that give this maximal algebraic connectivity.

3.2 Centralised Optimisation

3.2.1 Weighted graphs

Some of the seminal work in optimising graph Laplacian eigenvalues on weighted, undirected graphs using semi-definite programming comes from Stephen Boyd and his collaborators. In particular in [54], Xiao and Boyd formulated a semi-definite program (SDP) to find the edge weights in a network that give the fastest linear averaging in a network, and showed surprisingly that optimal edge weights may be negative. This problem can be formulated as an eigenvalue optimisation problem, minimising the spectral radius of the matrix $(\mathbf{I}-\mathbf{L}(\mathbf{w}))$, when edge weights are constrained so that the weighted degree of each node is not more than 1: i.e. $\text{diag}(\mathbf{L}) \le \mathbf{1}$. In [76], the effective graph resistance (a convex function of the spectrum of the graph Laplacian) is minimised over the edge weights using an SDP[1]. These problems share many similarities, and a number of similar problems are summarised succinctly in the conference paper by Boyd

[1]We propose a decentralised strategy for minimising the effective graph resistance in Chapter 6.

[77]. Techniques from convex optimisation were also used to find upper bounds on the algebraic connectivity of a number of families of graphs in [78], recovering some previously known upper bounds on the algebraic connectivity.

Much work has also been undertaken on Laplacian matrices where edge weights are functions of some state of the nodes (normally the relative positions and the distance between nodes). Frequently, this introduces non-convex constraints into the optimisation problem which makes solution via SDPs more difficult. Nevertheless, finding the network formation that maximises the algebraic connectivity, when the weight of edges is determined by the relative distance between two nodes is explored in [79]. The solution is found by iterated solutions of SDPs where the non-convex constraints are replaced locally by linear constraints. Such a process is guaranteed to find locally optimal solutions, but in practice appears to find globally optimal solutions often.

3.2.2 Unweighted graphs

The problem of selecting a number of edges to add to a graph to maximise algebraic connectivity was explored in [80]. This is an NP-hard combinatorial problem [81], and so an SDP relaxation was solved. A greedy heuristic based on the eigenvector associated with the algebraic connectivity[2], was then used to select the candidate edges, resulting in a good (but sub-optimal) graph.

Using the SDP relaxation with a branch and bound algorithm can be used to find globally optimal networks (this is known as Mixed-Integer Semidefinite Programming or MISDP) and is presented for undirected graphs in [82] and directed graphs in [83]. However, this method can be slow due to solving the SDP at each branch, and so a relaxation of the semi-definite constraint to a group of quadratic contraints was also proposed in [82] to accelerate the performance, to the detriment of no longer guaranteeing that the globally optimal solution is found.

Related to the problem of edge addition to maximise algebraic connectivity is the edge rewiring problem, where at each step one edge is chosen to be removed and another added. In [84] a heuristic method for selecting edges to be removed and added based on the difference between elements in the Fiedler vector was compared for some common network models. This method of adding edges based on the difference between elements in the Fiedler vector is effective for the same reasons that the Fiedler vector can be used to find a low cost partition of a graph. The problem of optimal edge addition and deletion for the effective graph resistance is explored in [85] using heuristics based upon the degree distribution of nodes, the Fiedler vector and the resistance distance between nodes. Edge addition was also looked at in [86] in terms of maximising the weighted sum of the algebraic connecitivity and the cost of a link, and again the focus was on augmenting the node with lowest degree to speed up the algorithm.

[2]Often called the Fiedler vector.

3.3 Decentralised Methods

Efforts have been made to develop strategies and heuristics for improving the algebraic connectivity using only information local to each node. For example, in [87] the effect of adding a link between the lowest degree node and a random node (which only requires local information) is compared to adding a link based upon the maximum difference in elements of the Fiedler vector (which is a global variable of the entire graph). A similar degree-based method for edge rewiring is analysed in [88], and this degree based method is also tested in [85] for reducing the total effective graph resistance of the network.

In [89] the global SDP was broken down into a number of local SDPs: one for each agent, with each agent solving an SDP for its two-hop neighbourhood. At each time step, the new positions of agents were updated as the average of the local SDPs. However, even for small graphs ($n = 6$), this distributed SDP strategy does not converge onto the centralised solution in roughly half of the simulated cases.

Decentralised control of networks is most crucial in collections of autonomous units, for example in connectivity maintenance in robot formations [90, 91, 92] where connectivity is maintained through the use of local potential fields around each agent and local estimates of network connectivity through consensus [91]. Potential fields are also used to preserve network connectivity in [93, 94]. In [95] decentralised formation control of a network of mobile robots that seeks to maximise the algebraic connectivity is achieved using the decentralised computation of the eigenvectors of the graph Laplacian, following the 'Decentralized Orthogonal Iteration Algorithm' proposed in [96]. The unfortunate drawback of this method is that all n eigenvectors of the graph Laplacian need to be estimated, and so the number of variables that each node must store grows linearly with the number of nodes in the network.

On the other hand, it is possible to estimate all the non-trivial eigenvalues of a symmetric graph Laplacian using just two local variables at each node and some signal processing techniques, as is done in [97]. In this decentralised method, each node i in the network interacts with its neighbours and adapts its internal state according to the local interaction rule:

$$\begin{aligned} \dot{x}_i &= z_i + \sum_{j \in \mathcal{N}_i} \left(z_i - z_j\right) \\ \dot{z}_i &= -x_i - \sum_{j \in \mathcal{N}_i} \left(x_i - x_j\right) \end{aligned} \tag{3.2}$$

so that the dynamics of the entire network can be described by the matrix differential equation:

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{z}} \end{bmatrix} = \mathbf{S} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} \mathbf{0} & \mathbf{I} + \mathbf{L} \\ -\mathbf{I} - \mathbf{L} & \mathbf{0} \end{bmatrix} \tag{3.3}$$

The eigenvalues of the skew symmetric matrix $\mathbf{S}$ are purely imaginary and are functions of the eigenvalues of the graph Laplacian matrix, such that all nodes' internal states x_i and z_i follow trajectories which are linear combinations of sinusoids with frequencies governed by the eigenvalues of the graph Laplacian. In fact the frequencies of these sinusoids are simply the eigenvalues of the graph Laplacian, shifted by 1, i.e. $1, 1+\lambda_2, \dots, 1+\lambda_n$. Each node i can then make an estimate of the entire spectrum of the graph Laplacian by estimating the frequencies of the sinusoids of which the signal $x_i(t)$ is composed in a finite time window. This can be achieved for example offline using Fourier analysis tools, or using online methods, as in [97]. The great advantage of this method is that the entire spectrum of the graph Laplacian can be deduced together using a relatively simple system, but there are also a number of significant drawbacks compared to other methods: primarily, no information is obtained about the associated eigenvectors of the Laplacian eigenvalues, and these can be used to estimate the partial derivatives of the eigenvalues (as seen later in Section 3.3.1.2); also, if the i^{th} component of the eigenvector associated with a particular eigenvalue is zero, then this mode will not excite the i^{th} agent's states, and node i will be blind to this particular mode; finally, eigenvalues with multiplicity greater than one will not be discernible from distinct eigenvalues.

Due to the lack of information to be gained about the eigenvectors of the graph Laplacian using the method of [97], this algorithm was not chosen to be utilised for estimating and optimising the algebraic connectivity of a network, and instead we focus on the method presented in [29], where nodes communicate in continuous time with their neighbours so as to maintain an estimate of just their element in the Fiedler vector (the eigenvector associated with the algebraic connectivity), rather than all vectors, and we proceed to look at more closely at this algorithm in the next section. A related discrete time algorithm is proposed in [98] which makes an estimate of the algebraic connectivity and the Fiedler eigenvector, but runs into numerical stability issues when the algebraic connectivity λ_2 is not well separated from λ_3.

3.3.1 Decentralised Estimation of the Algebraic Connectivity

In Yang et al. [29], the following ordinary differential equation was proposed to facilitate decentralised estimation of the algebraic connectivity in a network of autonomous agents:

$$\dot{\mathbf{a}} = -k_1\langle\mathbf{a}\rangle\mathbf{1} - k_2\mathbf{L}\mathbf{a} + k_3\left(1 - \langle\mathbf{a}^{\circ 2}\rangle\right)\mathbf{a} \tag{3.4}$$

where the angle brackets indicate the arithmetic mean of a vector, and Hadamard exponentiation is used so that $\langle\mathbf{a}\rangle = \frac{1}{n}\sum_i a_i$ and $\langle\mathbf{a}^{\circ 2}\rangle = \frac{1}{n}\sum_i a_i^2$. In this system, $\mathbf{a}$ is an estimate vector whose dynamics are such that, when the algebraic connectivity is a distinct eigenvalue, $\mathbf{a}$ tends towards aligning with the Fielder eigenvector, $\mathbf{v_2}$. The control parameters k_1, k_2 and k_3 each determine the relative strength of each of the three terms on the right hand side. The first term provides an action that deflates the consensus mode, which is the mode associated with the simple eigenvalue $\lambda_1 = 0$, $\mathbf{v}_1 = \beta\mathbf{1}$ (where β is any non-zero real number). This ensures that the estimate vector $\mathbf{a}$ is orthogonal to the consensus manifold. The second term is a direction update in which the algebraic connectivity is the dominant eigenvalue. Finally, the third term provides a normalising force which prevents the estimate vector converging towards the origin. Furthermore, at the stable equilibrium point of the system $\mathbf{a}^*$, it is proven in [29] that,

$$||\mathbf{a}^*||_2 = \sqrt{\frac{n(k_3 - k_2\lambda_2)}{k_3}} \tag{3.5}$$

so that an estimate of the algebraic connectivity λ_2 can be made by rearranging:

$$\lambda_2 = \frac{k_3}{k_2}\left(1 - \frac{\mathbf{a}^{*\mathrm{T}}\mathbf{a}^*}{n}\right) \tag{3.6}$$

The system (3.4) is essentially a continuous-time formulation of a power-iteration method for calculating the largest eigenvalue of the matrix $-\mathbf{L}$, which has been deflated on the consensus mode so that $-\lambda_2$ is the leading eigenvalue. A similar, discrete-time, power-iteration approach for the decentralised estimation of the algebraic connectivity is used in [99] by Li, Zhang and Xi.

Alternatively, using more conventional matrix notation, the system (3.4) can be written:

$$\dot{\mathbf{a}} = -\frac{k_1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}\mathbf{a} - k_2\mathbf{L}\mathbf{a} + k_3\mathbf{a} - \frac{k_3}{n}\mathbf{a}\mathbf{a}^{\mathrm{T}}\mathbf{a} \tag{3.7}$$

This matrix differential equation effectively describes the *macroscopic* view of the system, but equally we can focus on the update law each individual node follows; the *microscopic* view:

$$\dot{a}_i = -k_1\langle\mathbf{a}\rangle + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(a_j - a_i) + k_3\left(1 - \langle\mathbf{a}^{\circ 2}\rangle\right)a_i \tag{3.8}$$

From this viewpoint, it is clear that this update law is not completely decentralised; the arithmetic means, $\langle\mathbf{a}\rangle$ and $\langle\mathbf{a}^{\circ 2}\rangle$, are functions of all components of $\mathbf{a}$ which are not available to each node, following the constraint of using only local communication. In [29], this problem is circumvented by employing two Proportional-Integral (PI) average-consensus estimators [100], to make decentralised estimates of the averages.

3.3.1.1 PI average consensus

To make a decentralised estimate of the two arithmetic means: $\langle\mathbf{a}\rangle = \frac{1}{n}\sum_i a_i$ and $\langle\mathbf{a}^{\circ 2}\rangle = \frac{1}{n}\sum_i a_i^2$, two PI average-consensus estimators, as described in [100] are utilised. Such an estimator tracks the arithmetic mean of a dynamic input vector $\mathbf{u}(t)$ through the following coupled ODEs:

$$\dot{y}_i = k_\gamma(u_i - y_i) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_j - y_i) + k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_j - z_i) \tag{3.9}$$

$$\dot{z}_i = -k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_j - y_i) \tag{3.10}$$

with k_γ being the input gain, k_P being the proportional control gain, and k_I being the integral control gain. The variable y_i is the proportional variable, and z_i is the integral variable located at node i. This method is clearly fully decentralised as each node requires only variables located at its neighbours and incident edges, to compute the update to its proportional and integral variables.

It is shown in [100] that for positive control parameters k_γ, k_P and k_I, the stable equilibrium for each y_i is the mean of all components of the input vector $\mathbf{u}$, that is $y_i^* = \langle \mathbf{u} \rangle = \frac{1}{n}\mathbf{1}^\mathrm{T}\mathbf{u}$. Thus, to decentralise the system (3.8), two PI average consensus systems are required, the first with $\mathbf{u} \triangleq \mathbf{a}$, and the second with $\mathbf{u} \triangleq \mathbf{a}^{\circ 2}$. To distinguish these two estimators, we use subscripts to designate which the vector is being averaged:

$$\dot{\mathbf{y}}_\mathbf{a} = k_\gamma(\mathbf{a} - \mathbf{y}_\mathbf{a}) - k_P\mathbf{L}\mathbf{y}_\mathbf{a} - k_P\mathbf{L}\mathbf{z}_\mathbf{a} \tag{3.11}$$

$$\dot{\mathbf{z}}_\mathbf{a} = k_P\mathbf{L}\mathbf{y}_\mathbf{a} \tag{3.12}$$

and

$$\dot{\mathbf{y}}_{\mathbf{a}^{\circ 2}} = k_\gamma\left(\mathbf{a}^{\circ 2} - \mathbf{y}_{\mathbf{a}^{\circ 2}}\right) - k_P\mathbf{L}\mathbf{y}_{\mathbf{a}^{\circ 2}} - k_P\mathbf{L}\mathbf{z}_{\mathbf{a}^{\circ 2}} \tag{3.13}$$

$$\dot{\mathbf{z}}_{\mathbf{a}^{\circ 2}} = k_P\mathbf{L}\mathbf{y}_{\mathbf{a}^{\circ 2}} \tag{3.14}$$

Substituting in these two PI average consensus estimators for the global functions $\langle \mathbf{a} \rangle$ and $\langle \mathbf{a}^{\circ 2} \rangle$, we arrive at the full set of decentralised equations that each agent i follows, in order to update its five internal states:

$$\begin{aligned}
\dot{a}_i &= -k_1 y_{\mathbf{a},i} + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(a_j - a_i) + k_3(1 - y_{\mathbf{a}^{\circ 2},i})a_i \\
\dot{y}_{\mathbf{a},i} &= k_\gamma(a_i - y_{\mathbf{a},i}) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a},j} - y_{\mathbf{a},i}) + k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_{\mathbf{a},j} - z_{\mathbf{a},i}) \\
\dot{z}_{\mathbf{a},i} &= -k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a},j} - y_{\mathbf{a},i}) \\
\dot{y}_{\mathbf{a}^{\circ 2},i} &= k_\gamma(a_i^2 - y_{\mathbf{a}^{\circ 2},i}) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a}^{\circ 2},j} - y_{\mathbf{a}^{\circ 2},i}) + k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_{\mathbf{a}^{\circ 2},j} - z_{\mathbf{a}^{\circ 2},i}) \\
\dot{z}_{\mathbf{a}^{\circ 2},i} &= -k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a}^{\circ 2},j} - y_{\mathbf{a}^{\circ 2},i})
\end{aligned} \tag{3.15}$$

and each node can maintain its own estimate of the algebraic connectivity, $\widetilde{\lambda_2}^{(i)}$:

$$\widetilde{\lambda_2}^{(i)} = \frac{k_3}{k_2}\left(1 - y_{\mathbf{a}^{\circ 2},i}\right) \tag{3.16}$$

In [29] it is argued that there needs to be a sufficient time scale separation between the eigenvalue estimator and the two PI average consensus estimators for stability of the decentralised system, however no bound on this separation is given. In Chapter 5, we look at a similar system for estimating the largest eigenvalue of the graph Laplacian matrix, and derive a lower bound on a sufficient time-scale separation.

3.3.1.2 Partial derivatives of the algebraic connectivity

An interesting and important feature of the symmetric graph Laplacian matrix is that the partial derivatives of any particular eigenvalue with respect to the edge weights can be easily calculated from the normalised eigenvector associated with the particular eigenvalue. For clarity, we provide a proof found in [99], though this property was found earlier, [101]:

Lemma 3.1 (Theorem 3.3 [101]). *For a simple eigenvalue λ_k of* $\mathbf{L(w)}$*, of an undirected graph, with associated unit eigenvector* $\mathbf{v_k}$*, such that* $\mathbf{Lv_k} = \lambda_k \mathbf{v_k}$*, the partial derivative of the eigenvalue with respect to an edge* $w_{\{i,j\}}$ *may be calculated as:*

$$\frac{\partial \lambda_k}{\partial w_{\{i,j\}}} = (v_{k,i} - v_{k,j})^2 \leq 2 \tag{3.17}$$

Proof. As $\mathbf{L(w)}$ is symmetric, we can deduce the unit left eigenvector corresponding to λ_k, $\mathbf{u_k}$, such that $\mathbf{u_k v_k} = 1$: $\mathbf{u_k} = \mathbf{v_k}^{\mathrm{T}}$. Premultiplying the eigenvector relation, $\mathbf{Lv_k} = \lambda_k \mathbf{v_k}$, by this left eigenvector yields:

$$\mathbf{v_k}^{\mathrm{T}}\mathbf{L v_k} = \lambda_k \mathbf{v_k}^{\mathrm{T}}\mathbf{v_k} = \lambda_k$$

Then, taking the derivative with respect to the edge weight $w_{\{i,j\}}$ of both sides, we find that:

$$\frac{\partial \lambda_k}{\partial w_{\{i,j\}}} = \frac{\partial \mathbf{v_k}^{\mathrm{T}}}{\partial w_{\{i,j\}}}\mathbf{L v_k} + \mathbf{v_k}^{\mathrm{T}}\frac{\partial \mathbf{L}}{\partial w_{\{i,j\}}}\mathbf{v_k} + \mathbf{v_k}^{\mathrm{T}}\mathbf{L}\frac{\partial \mathbf{v_k}}{\partial w_{\{i,j\}}}$$

However, as $\mathbf{L}$ is symmetric, it is ensured that:

$$\frac{\partial \mathbf{v_k}^{\mathrm{T}}}{\partial w_{\{i,j\}}}\mathbf{L v_k} + \mathbf{v_k}^{\mathrm{T}}\mathbf{L}\frac{\partial \mathbf{v_k}}{\partial w_{\{i,j\}}} = \lambda_k \frac{\partial (\mathbf{v_k}^{\mathrm{T}}\mathbf{v_k})}{\partial w_{\{i,j\}}} = 0$$

Thus,

$$\frac{\partial \lambda_k}{\partial w_{\{i,j\}}} = \mathbf{v_k}^{\mathrm{T}}\frac{\partial \mathbf{L}}{\partial w_{\{i,j\}}}\mathbf{v_k} \tag{3.18}$$

Without loss of generality, we can relabel the nodes i and j to 1 and 2, revealing that

$$\begin{aligned}\frac{\partial \lambda_k}{\partial w_{\{1,2\}}} &= \mathbf{v_k}^{\mathrm{T}}\left(\begin{array}{cc|c} +1 & -1 & \multirow{2}{*}{$\varnothing$} \\ -1 & +1 & \\ \hline \multicolumn{2}{c|}{\varnothing} & \varnothing \end{array}\right)\mathbf{v_k} \\ &= \left(v_{k,1} - v_{k,2}\right)^2\end{aligned}$$

As any two nodes i and j could have been labelled 1 and 2, we then recover the original equation:

$$\frac{\partial \lambda_k}{\partial w_{\{i,j\}}} = (v_{k,i} - v_{k,j})^2 \leq 2 \tag{3.19}$$

Finally, the inequality follows from the fact that $||\mathbf{v_k}||_2 = 1$. Equality is only achieved when $v_{k,i} = \frac{1}{\sqrt{2}}$, $v_{k,j} = -\frac{1}{\sqrt{2}}$, and all other elements of $\mathbf{v_k}$ are zero. □

Lemma 3.1 has the corollary that each node can make decentralised estimates of the partial derivative of the algebraic connectivity with respect to the weight of each of its incident edges. This is achieved using only its own component of the estimate vector a_i, its one-hop neighbour's component of the estimate vector a_j and its own estimate for the Euclidean norm of the estimate vector. Specifically, noting that at the equilibrium point of the second PI average consensus subsystem, $y^*_{\mathbf{a}^{\circ 2},i} = \langle \mathbf{a}^2 \rangle = \frac{1}{n}||\mathbf{a}||_2^2$:

$$\widetilde{\left(\frac{\partial \lambda_2}{\partial w_{\{i,j\}}}\right)}^{(i)} = \frac{(a_i - a_j)^2}{n y_{\mathbf{a}^{\circ 2},i}} \tag{3.20}$$

Note 3.1. *Note that the denominator in the equation* (3.20) *will be equal at equilibrium for all nodes. Therefore, this common factor only affects the magnitude of the gradient, and not the direction of steepest descent. Hence, this factor may be omitted in many gradient descent algorithms.*

The partial derivative of the algebraic connectivity with respect to any edge weight is proportional to the square of the difference between the incident components of the Fiedler vector. For this reason, either the square of the difference in components, or simply the absolute difference between any two components[3], is often used as a heuristic for choosing which edges to add in, or remove from, a network, for example [80, 84].

3.3.2 A discrete time power iteration method for estimation of the algebraic connectivity

A power iteration method in discrete time for estimating the algebraic connectivity and the Fiedler vector is also proposed by Bertrand and Moonen [98, 102, 30]. The algorithm proposed is based upon the power iteration of the matrix:

$$\mathbf{M} = \mathbf{I} - \frac{1}{\alpha}\mathbf{L} \tag{3.21}$$

where the constant $\alpha \geq \lambda_k$, which is guaranteed if $\alpha = n$ (for unweighted networks) or $\alpha = 2\Delta$, with Δ being the greatest weighted degree of the network. To estimate the algebraic connectivity we then consider the difference equation:

$$\mathbf{x}(i+1) = \mathbf{M}\mathbf{x}(i) \tag{3.22}$$

Note that the update step (3.22) is mean preserving, so a further update step is needed to ensure that the mean of the estimator vector is returned to zero, from any intitial offset introduced in the initialising procedure, or introduced through floating point error. The update step which removes any nonzero mean is simply:

$$\mathbf{x}(i+1) = \mathbf{L}\mathbf{x}(i) \tag{3.23}$$

[3]The square of the difference is a monotonic increasing function of the absolute difference between elements.

and this update is carried out every N^{th} iteration, with the power iteration (3.22) occuring at all other times. Thus, the combined system can be written as:

$$\mathbf{x}(i+1) = \begin{cases} \mathbf{Lx}(i) & \text{if } i \bmod(N) = 0 \\ \mathbf{Mx}(i) & \text{elsewise} \end{cases} \tag{3.24}$$

A bound on N is given in Theorem 3.1 in [30] so that the estimate vector $\mathbf{x}$ will eventually converge onto the Fielder vector:

$$N > \frac{\log\left(\frac{\lambda_3}{\lambda_2}\right)}{\log\left(\frac{\alpha-\lambda_2}{\alpha-\lambda_3}\right)} + 1 \tag{3.25}$$

A less tight lower bound is also provided in [30] as $N > \frac{\alpha}{\lambda_2}$.

However, it is noted in [30] that both operations do not preserve the norm of $\mathbf{x}$. Thus, a separate estimate is made in a decentralised manner for the rate at which the norm decreases:

$$r(i) = \frac{||\mathbf{Mx}(i)||}{||\mathbf{x}(i)||} \tag{3.26}$$

The estimate of the rate $r(i)$ is denoted by p and is broadcast across the network (not necessarily at every time step). This estimate p is then used to compensate for the change in norm caused by the operations (3.22) and (3.23), resulting in the system:

$$\mathbf{x}(i+1) = \begin{cases} \frac{1}{\alpha|1-p|}\mathbf{Lx}(i) & \text{if } i \bmod(N) = 0 \\ \frac{1}{p}\mathbf{Mx}(i) & \text{elsewise} \end{cases} \tag{3.27}$$

The details by which $r(i)$ is estimated by each node, and each node receives the estimate p can be found in Section $3-A$ of [30]. Furthermore, Table 2 in [30] contains complete details of the distributed update, mean-shifting, and renormalisation procedure.

It can be seen that the structure of this algorithm bears a strong similarity to the method presented in [29] - there are three updates: an update to deflate on the consensus mode, ensuring that the mean of the estimate vector is zero; a direction update which causes the mode associated with the algebraic connectivity to dominate; and a renormalisation measure to stop the estimate vector growing or shrinking without bound.

However, there are a number of disadvantages associated with this method. In particular the norm of the estimate vector is not known, so a further decentralised estimate of the norm of the vector is required, if it is desired that the normalised Fiedler eigenvector be estimated. Furthermore, convergence speed of this method, as calculated in Remark 3 of [30], is determined by the quantity:

$$\left(\frac{\lambda_2}{\lambda_3}\right)^{\frac{1}{N}} \left(\frac{\alpha-\lambda_2}{\alpha-\lambda_3}\right)^{\frac{N}{N-1}} \tag{3.28}$$

In particular, this means that when λ_2 is not simple, the system often exhibits numerical instability, with the estimate vector rapidly diverging to infinity, or converging to zero.

3.4 Summary

In this Chapter, we reviewed a number of centralised and decentralised methods for adapting edge weights in networks, in order to control or optimise the graph Laplacian eigenvalues. When centralised methods are suitable for an application, techniques from semi-definite programming (SDP) can prove can lead to efficient solution of many network optimisation problems [77]. In recent years, SDP has been combined with integer programming techniques, resulting in Mixed Integer Semidefinite Programs (MISDPs) which have been used to successfully optimise a number of unweighted networks [83, 82]. Heuristics using the Fiedler eigenvector, calculated centrally, have also been used to choose which edges to add or rewire in an unweighted network [80, 84].

When centralised solutions are not feasible, a number of decentralised heuristics have been developed to find sub-optimal networks with improved performance, particularly when dealing with edge addition and rewiring problems in unweighted networks. Many of these use local degree information as the heuristic [88, 87]. We then paid special attention to two decentralised methods for estimating the algebraic connectivity and Fiedler vector of a network, based upon power iteration [29, 30]. However, one of the methods [30] showed signs of instability when the algebraic connectivity was not a simple eigenvalue. For this reason, we choose to use the decentralised method for the estimation of the algebraic connectivity presented in [29] as the basis for the decentralised optimisation of the algebraic connectivity presented in Chapter 4. In subsequent Chapters, we also use the continuous time power-iteration ideas from [29] to develop three further decentralised eigenvalue estimators for estimating important spectral functions of the graph Laplacian: the spectral radius in Chapter 5, and the graph resistance and reduced Laplacian determinant in Chapter 6.

CHAPTER 4

WEIGHT ADAPTATION LAWS

In the previous Chapter, it was seen how a decentralised estimate of the algebraic connectivity and its associated eigenvector, the Fiedler vector, could be inferred by a network of agents communicating only with their neighbours [29]. In the paper in which this method was proposed, [29], this estimator was then used to formulate a gradient controller to maintain connectivity in a network of mobile agents, with multiple leaders. In this Chapter, we will show that by direct control of the edge weights, the algebraic connectivity estimator may be utilised in conjunction with a suitable weight adaptation law to solve a number of decentralised constrained connectivity optimisation problems.

4.1 Algebraic Connectivity Maximisation

We begin by looking at the following optimisation problem: maximising the algebraic connectivity of a weighted undirected graph, subject to constraints on the edge weights. We choose to bound the region of feasible edge weights by adding two constraints: non-negativity of the edge weights and upper bounds on the weighted degrees of nodes, the diagonal entries of $\mathbf{L}$, $l_{i,i}$. Maximum allowed weighted degrees $\kappa_i \in \mathbb{R}_{>0}$ are chosen to model a group of individual agents, making decisions on allocating their own resources, of which they have a limited amount. Non-negativity is chosen to bound the feasible set of edge weights $\mathscr{W}$, such that $\mathscr{W}$ is the compact convex set,

$$\mathscr{W} = \{\mathbf{w} : \mathbf{w} \geq \mathbf{0} \cap l_{i,i} \leq \kappa_i, \forall i\} \tag{4.1}$$

corresponding to a polytope in the positive orthant of $\mathbb{R}^m$, which guarantees that $\lambda_2 > 0$ for connected graphs.

Setting this problem in the standard form of a continuous optimisation problem, we have:

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximise}} \quad & \lambda_2\big(\mathbf{L}(\mathbf{w})\big) \\ \text{subject to} \quad & \mathbf{w} \in \mathscr{W} \end{aligned} \tag{4.2}$$

The problem is a convex optimisation problem, and can be formulated as a semidefinite program [77] for efficient numerical computation by using, for example, a primal-dual interior point method [103]. It can be seen that the optimisation problem (4.2) is equivalent to the following formulation:

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximise}} \quad & \gamma \\ \text{subject to} \quad & \mathbf{X} = \mathbf{L}(\mathbf{w}) + \beta \mathbf{1}\mathbf{1}^T - \gamma \mathbf{I} \succeq 0, \\ & \mathbf{w} \in \mathscr{W}, \quad \gamma, \beta \in \mathbb{R} \end{aligned} \tag{4.3}$$

This is because $\mathbf{X}$ is a matrix with eigenvalues $n\beta - \gamma, \lambda_2 - \gamma, \lambda_3 - \gamma, \ldots, \lambda_n - \gamma$: the addition of $\beta \mathbf{1}\mathbf{1}^\top$ to $\mathbf{L}$ which has $\lambda_1 = 0$, with associated eigenvector $\mathbf{v_1} = \mathbf{1}$, increases the value of this eigenvalue by $n\beta$; and the subtraction of $\gamma\mathbf{I}$ reduces the value of all eigenvalues of $\mathbf{L}$ by γ. This result is due to the fact that the matrices $\mathbf{L}$, $\mathbf{1}\mathbf{1}^T$, and $\mathbf{I}$ are simultaneously diagonalisable by the matrix $\mathbf{V}$, being the matrix of column eigenvectors of $\mathbf{L}$.

The quantity β is a free variable, controlling the eigenvalue of the consensus mode, which when greater than $\frac{\lambda_2}{n}$ makes the eigenvalue associated with the Fiedler vector, $\lambda_2 - \gamma$, the smallest eigenvalue of $\mathbf{X}$. Then, maximising the variable γ, under the restriction that $\lambda_2 - \gamma > 0$, induces maximisation of λ_2.

Note that the feasibility region can also be included in a *linear matrix inequality* (LMI), using a block diagonal formulation. Then the optimisation problem can be formulated as the eigenvalue problem (EVP) [104]:

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximise}} \quad & \gamma \\ \text{subject to} \quad & \mathbf{Y} = \begin{bmatrix} \mathbf{L}(\mathbf{w}) + \beta\mathbf{1}\mathbf{1}^T - \gamma I & 0 & 0 \\ 0 & \operatorname{diag}(\mathbf{w}) & 0 \\ 0 & 0 & \operatorname{diag}(\boldsymbol{\kappa}) - \operatorname{diag}(\mathbf{L}(\mathbf{w})) \end{bmatrix} \succeq 0, \\ & \beta = \text{const.} > \frac{\min_i \kappa_i}{n-1} \end{aligned} \tag{4.4}$$

The pressing open problem is to find a strategy to perform such an optimisation on-line and in a distributed manner, as opposed to the centralised optimisation strategies employed for the efficient solution of SDPs [105, 106, 107]. Then, edge weights can be modified continuously to converge onto the optimal distribution by agents lacking knowledge of the entire network, solely through communication with their neighbours.

To this end, we present two continuous-time update laws for the adaptation of edge weights. The first weight update law is based on the method of steepest descent where the inequality constraints are enforced through the use of *logarithmic barrier functions* [108]. The second method converges on the optimal edge weights through the satisfaction of the Karush-Kuhn-Tucker (KKT) conditions [109, 110]. Each method has its own advantages and disadvantages: the logarithmic barrier method guarantees that edge weights remain feasible for all time if initialised in the feasible region, but is undefined if the edge weights are forced into the infeasible region by an external factor[1]; satisfaction of the KKT system gives no such guarantee on the feasibility of the edge weights in the transient response, but is more robust to both initialisation outside of the feasible region, and to perturbations which may force edge weights into an infeasible regime.

4.2 Steepest Descent with Adaptive Logarithmic Barriers

Initially, we will assume that the algebraic connectivity and its partial derivative with respect to the edge weights are known perfectly to all agents, before removing this assumption later in Section 4.3. When $\lambda_2(\mathbf{L})$ is not simple, and the sensitivity of the algebraic connectivity with respect to the edge weights $\frac{\partial \lambda_2}{\partial w_{i,j}}$ is not well defined, we instead take the minimum of the sub-differential. This strong assumption will be relaxed later, in subsequent chapters.

To find an on-line solution to the constrained optimisation problem, (4.2), we instead perform an unconstrained optimisation by steepest descent on a modified objective function $F(\mathbf{w})$ incorporating the inequality constraints into the objective function through the addition of *logarithmic barrier functions*. Typically, the value of a barrier function, say $\Phi(\cdot)$, rapidly and smoothly approaches $+\infty$ as an inequality constraint is encroached upon, and is evaluated at $+\infty$ when breached. A logarithmic function may be employed to achieve this purpose as discussed in [108]: for a generic inequality, say $g(\mathbf{w}) \leq 0$, such a logarithmic barrier function may be defined as:

$$\Phi(\mathbf{w}, q) = \begin{cases} -\frac{1}{q}\log(-g(\mathbf{w})) & \text{if } g(\mathbf{w}) < 0 \\ +\infty & \text{otherwise} \end{cases} \tag{4.5}$$

where the variable $q > 0$ controls the severity of the barrier terms. Incorporating the inequality constraints as expressed in (5.64), the modified objective function we wish to minimise then becomes:

$$F(\mathbf{w}, q) = -\lambda_2(\mathbf{L}) + \Phi_1(\mathbf{w}, q) + \Phi_2(\mathbf{w}, q) \tag{4.6}$$

where the two logarithmic barrier functions are given by:

$$\Phi_1(\mathbf{w}, q) = -\frac{1}{q}\sum_{i=1}^{m} \log(w_i), \ \mathbf{w} \geq 0, \ \text{else}, \Phi_1 = \infty \tag{4.7}$$

$$\Phi_2(\mathbf{w}, q) = -\frac{1}{q}\sum_{i=1}^{n} \log(\kappa_i - l_{i,i}), \ l_{i,i} \leq \kappa_i, \ \forall i, \ \text{else}, \Phi_2 = \infty \tag{4.8}$$

[1]For example, the maximal allowable weighted degree κ_i of a node could fall due an external disturbance.

Φ_1 enforces the non-negativity weight constraints, and Φ_2 enforces the constraint on the maximum allowable weighted degrees. We will first state some simple results that hold for any $q > 0$.

Lemma 4.1. *$F(\mathbf{w}, q)$ is finite valued for $\mathbf{w} \in \text{int}(\mathscr{W})$ and evaluates to $+\infty$ for $\mathbf{w} \in \overline{\mathscr{W}^c}$, the closure of the complement of the feasible set.*

Proof. For $\mathbf{w} \in \mathscr{W}$, $\lambda_2(\mathbf{L})$ is finite. As are $\Phi_1(\mathbf{w}, q)$, $\Phi_2(\mathbf{w}, q), \forall \mathbf{w} \in \text{int}(\mathscr{W})$. Thus $F(\mathbf{w}, q)$ is finite for all $\mathbf{w} \in \text{int}(\mathscr{W})$. For $\mathbf{w} \notin \text{int}(\mathscr{W})$, then at least one barrier function must evaluate to $+\infty$ from the definition of logarithmic barrier functions given in (4.7)-(4.8). □

Lemma 4.2. *$F(\mathbf{w})$ is strictly convex in* $\text{int}(\mathscr{W})$.

Proof. It is known that $\lambda_2(\mathbf{L})$ is a concave function of edge weights; $-\lambda_2(\mathbf{L})$ is convex. Also, the sum of all logarithmic barrier functions is strictly convex, over the region $\text{int}(\mathscr{W})$, where second order derivatives are properly defined. Thus the objective function, a sum of convex and strictly convex functions, must be strictly convex. □

Theorem 4.1. *For any fixed $q > 0$, there exists a unique global minimum of $F(\mathbf{w}, q)$ with minimiser $\mathbf{w}^*(q) \in \text{int}(\mathscr{W})$.*

Proof. From Lemma 4.1, the objective function can be lower bounded by some finite value, say $z(q)$, which is the largest scalar satisfying $z(q) \le F(\mathbf{w}, q)$, $\forall \mathbf{w}$. Then $z(q)$ is a minimum of $F(\mathbf{w}, q)$, and the minimiser is in $\text{int}(\mathscr{W})$. By Fermat's theorem [111], any extremum of a differentiable function is a critical point; the minimiser $\mathbf{w}^*(q) : F(\mathbf{w}^*, q) = z(q)$ is then a critical point such that $\frac{\partial}{\partial \mathbf{w}} F(\mathbf{w}^*, q) = \mathbf{0}$. And by strict convexity of the objective function, Lemma 4.2, any minimum of the function is indeed global and unique. □

Now, a simple gradient descent law can be applied to the edge weights so that they adapt towards this global minimum for any positive constant q and positive control constants a and c_1. For example, and without loss of generality, we choose:

$$\ddot{\mathbf{w}} = -k_a \frac{\partial}{\partial \mathbf{w}} F(\mathbf{w}, q) - c_1 \dot{\mathbf{w}}, \qquad a > 0, c_1 > 0 \tag{4.9}$$

which indeed has one stationary point as the unique minimiser,

$$\mathbf{w} = \mathbf{w}^*(q) \text{ s.t. } \ddot{\mathbf{w}} = \dot{\mathbf{w}} = 0 \quad \text{yields} \quad \frac{\partial}{\partial \mathbf{w}} F(\mathbf{w}^*, q) = \mathbf{0} \tag{4.10}$$

Note 4.1. *Note that we use second order dynamics here, treating the weight vector $\mathbf{w}$ as a point mass in the feasible region $\mathscr{W}$, forced down the slope of the potential well of the objective function in the direction of steepest descent. Energy in the system is dissipated through the damping coefficient c_1, which effects a drag on the point mass. It would also be entirely reasonable to use first order dynamics, but it was found that second order dynamics performed better (both converging faster* *and giving greater flexibility in tuning the control constants) in simulation. The use of second order dynamics here can be seen to be an application of the heavy ball with friction method [112, 113, 114], which can offer significant acceleration over steepest descent optimisation methods [115].*

Before proving the stability of the weight update law (4.9), we investigate how the location of the minimiser, $\mathbf{w}^*(q)$, changes as the strength of the logarithmic barrier functions is altered, through varying q.

4.2.1 The Location of $\mathbf{w}^*(q)$

The location of the minimiser $\mathbf{w}^*(q)$ may be calculated directly from (4.10). Finding the partial derivative of the objective function,

$$F(\mathbf{w},q) = -\lambda_2(\mathbf{L}) - \frac{1}{q}\left(\sum_{i=1}^{m}\log(w_i) + \sum_{i=1}^{n}\log(\kappa_i - l_{i,i})\right) \tag{4.11}$$

with respect to the i^{th} edge weight, yields:

$$\begin{aligned}\frac{\partial}{\partial w_i}F(\mathbf{w}) &= -\frac{\partial}{\partial w_i}\lambda_2(\mathbf{L}) - \frac{1}{q}\sum_{j=1}^{m}\frac{\partial}{\partial w_i}\log(w_j) - \frac{1}{q}\sum_{j=1}^{n}\frac{\partial}{\partial w_i}\log(\kappa_j - l_{j,j}) \\ &= -\frac{\partial}{\partial w_i}\lambda_2(\mathbf{L}) - \frac{1}{q}\left(\frac{1}{w_i} - \sum_{j=1}^{n}\frac{\frac{\partial}{\partial w_i}l_{j,j}}{\kappa_j - l_{j,j}}\right) \end{aligned} \tag{4.12}$$

It can be seen that in (4.12), the barrier functions are entirely local to each edge; the only information required by each edge is its own weight, and the current weighted degree and maximum degree of its two incident nodes. Equation (4.12) also reveals that the minimiser $\mathbf{w}^*(q)$ must be such that the gradient of $\lambda_2(\mathbf{L})$ is perfectly balanced by the gradients of the barrier functions, as can be schematically seen in Figure 4.1, so that for all w_i:

$$\left.\frac{\partial F(\mathbf{w})}{\partial w_i}\right|_{\mathbf{w}^*} = -\frac{\partial}{\partial w_i}\lambda_2(\mathbf{L}) - \frac{1}{q}\left(\frac{1}{w_i} - \sum_{j=1}^{n}\frac{\frac{\partial}{\partial w_i}l_{j,j}}{\kappa_j - l_{j,j}}\right) = 0 \tag{4.13}$$

This relation fully determines the location of the stationary point $\mathbf{w}^*$ dependent upon q, the severity of the barrier functions.

At this point, it is prudent to characterise how the distance from $\mathbf{w}^*$ to the asymptote representing the limiting inequality constraint changes as a function of q.

Lemma 4.3. *As $q \to \infty$, the stationary point $\mathbf{w}^*(q)$ asymptotically converges onto an asymptote of the barrier functions, $\sum_i \Phi_i(\mathbf{w})$.*

Proof. We have already shown that $\mathbf{w}^*$ is the argument at which the gradient of $-\lambda_2(\mathbf{L})$ exactly balances the gradient of the sum of all barrier functions. This is equivalent to saying that $\mathbf{w}^*$ is the argument at which $\lambda_2(\mathbf{L})$ is tangent to the sum of all barrier functions and some constant c such that the tangential point exists, $\sum_i \Phi_i(\mathbf{w}) + c$, see Figure 4.1.

Then, it is clear to see that there can be only one $\mathbf{w}^*$ and this $\mathbf{w}^*$ must always exist as $\lambda_2(\mathbf{L})$ is concave and $\sum_i \Phi(\mathbf{w})$ is a smooth, strictly convex function with gradient spanning the entire vector space $\mathbb{R}^m$.

The gradient of the barrier functions with respect to w_i at any given point is inversely proportional to the value of q,

$$\frac{\partial}{\partial w_i} \sum_j \Phi_j(\mathbf{w}) = \frac{1}{q}\left(\sum_{j=1}^{n} \frac{\frac{\partial}{\partial w_i} l_{j,j}}{\kappa_j - l_{j,j}} - \frac{1}{w_i}\right) \tag{4.14}$$

thus, for any positive gradient $\nabla\lambda_2(\mathbf{L}) > \mathbf{0}$, as q^{-1} becomes small, we require

$$\forall i, \exists j : \frac{\frac{\partial}{\partial w_i} l_{j,j}}{\kappa_j - l_{j,j}} \to \infty \tag{4.15}$$

That is each edge weight must tend towards a limit enforced by at least one of the inequality constraints $l_{j,j} \le \kappa_j$, for which it is at least partly responsible, $\frac{\partial l_{j,j}}{\partial w_i} \neq 0$. Each edge weight tends to the bound of at least one asymptote, and $\exists j : l_{j,j} \to \kappa_j$ as $q \to \infty$; at least one node must be a choking point for each weight. This is also apparent as $\lambda_2(\mathbf{L})$ is a non-decreasing function of edge weights. □

Lemma 4.4. *As $q \to \infty$ the stationary point $\mathbf{w}^*$ tends towards $\mathbf{w}_{\text{opt}}$, the maximiser of $\lambda_2(\mathbf{L}(\mathbf{w}))$, $\forall \mathbf{w} \in int\mathscr{W}$.*

Proof. From Theorem 4.1, we know that the stationary point $\mathbf{w}^*$ is the unique global minimiser of $F(\mathbf{w}, q)$. Considering the case when q^{-1} becomes very small, the objective function $F(\mathbf{w})$ becomes

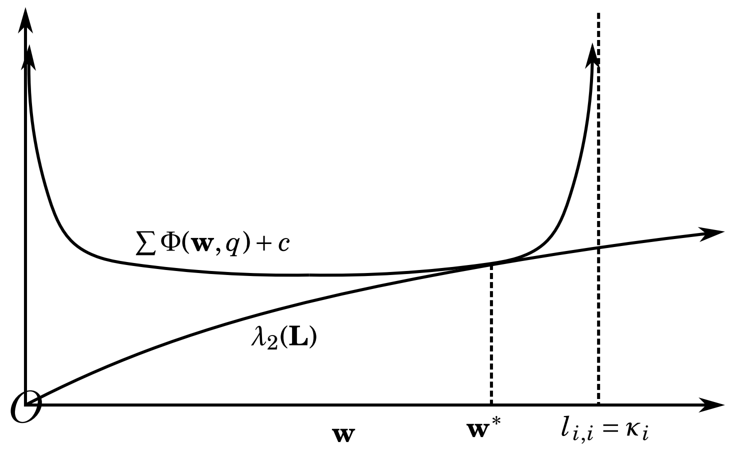


Figure 4.1: Graphical representation of the one-dimensional case, illustrating $\mathbf{w}^*$ as the argument at the tangent of the sum of barrier functions and $\lambda_2(\mathbf{L})$. As the barrier function becomes vertically compressed by the increase of q, $\mathbf{w}^*$ must move towards the asymptote at positive $\mathbf{w}$.

dominated by the term $-\lambda_2(\mathbf{L}(\mathbf{w}))$ on the interior of the feasible set. Yet on the boundary of the feasible set, $\mathbf{w} \in \partial \mathscr{W}$, the barrier terms do not diminish as they evaluate to $+\infty$. Therefore, for diminishingly small q^{-1} the global minimum of the objective function $F(\mathbf{w}, q)$ tends to the minimum of $-\lambda_2(\mathbf{L}(\mathbf{w}))\ \forall \mathbf{w} \in \mathrm{int}(\mathscr{W})$. That is, the minimiser of $F(\mathbf{w}, q)$, $\mathbf{w}^*(q)$, tends towards the maximiser, $\mathbf{w}_{\mathrm{opt}}$, of $\lambda_2(\mathbf{L}(\mathbf{w}))$ subject to strictly satisfying the non-negativity and bandwidth constraints, $l_{i,i} \leq \kappa_i$, as q becomes large. □

4.2.2 Stability of the weight adaptation law with fixed q

Now that the location of the minimiser $\mathbf{w}^*$ has been established in (4.13), a Lyapunov stability analysis may be carried out to prove the convergence of $\mathbf{w}$ towards the minimiser $\mathbf{w}^*$.

Theorem 4.2. *The adaptation algorithm (4.9) guarantees that*

$$\lim_{t\to\infty} \mathbf{w} = \mathbf{w}^* \tag{4.16}$$

Proof. The objective function we wish to minimise can be seen as a potential well, with a global minimum of $F(\mathbf{w}^*)$, located at $\mathbf{w} = \mathbf{w}^*$. We can then define a Lyapunov function for the system governed by Equation (4.9) in terms of the displacement from the minimiser. Defining the edge weight displacement (error from the minimiser) $\boldsymbol{\psi} = \mathbf{w} - \mathbf{w}^*$, so that $\dot{\boldsymbol{\psi}} = \dot{\mathbf{w}}$, $\ddot{\boldsymbol{\psi}} = \ddot{\mathbf{w}}$, a valid Lyapunov function is:

$$V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) = k_a(F(\boldsymbol{\psi} + \mathbf{w}^*) - F(\mathbf{w}^*)) + \frac{1}{2}||\dot{\boldsymbol{\psi}}||_2^2 \tag{4.17}$$

which is positive definite and radially unbounded. Differentiating with respect to time we find:

$$\begin{aligned}\dot{V}(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) &= \dot{\boldsymbol{\psi}}^T \cdot \frac{\partial V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}})}{\partial \boldsymbol{\psi}} + \ddot{\boldsymbol{\psi}}^T \cdot \frac{\partial V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}})}{\partial \dot{\boldsymbol{\psi}}} \\ &= k_a \dot{\boldsymbol{\psi}}^T \cdot \nabla F(\boldsymbol{\psi} + \mathbf{w}^*) + \ddot{\boldsymbol{\psi}}^T \cdot \dot{\boldsymbol{\psi}} \\ &= k_a \dot{\boldsymbol{\psi}}^T \cdot \nabla F(\boldsymbol{\psi} + \mathbf{w}^*) - k_a \dot{\boldsymbol{\psi}}^T \cdot \nabla F(\mathbf{w}) - c_1 \dot{\boldsymbol{\psi}}^T \cdot \dot{\mathbf{w}} \\ &= -c_1 ||\dot{\boldsymbol{\psi}}||_2^2\end{aligned} \tag{4.18}$$

which is negative semi-definite, for $c_1 > 0$. Applying the Krasovskii-Lasalle principle [116], we observe the set,

$$S = \{(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) : \dot{V}(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) = 0\} = \{(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) : \dot{\boldsymbol{\psi}} = 0\}$$

contains no trajectories of the system aside from the trivial trajectory $(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}) = 0$. If at some point in time $\dot{\boldsymbol{\psi}} = 0$ and $\boldsymbol{\psi} \neq 0$ ($\mathbf{w} \neq \mathbf{w}^*$) then by (4.9), $\ddot{\boldsymbol{\psi}} \neq 0$ and the trajectory will leave the set S.

This satisfies all conditions of the Krasovskii-Lasalle principle, hence the system is asymptotically stable on $\boldsymbol{\psi} = \mathbf{0}$, $\mathbf{w} = \mathbf{w}^*$. □

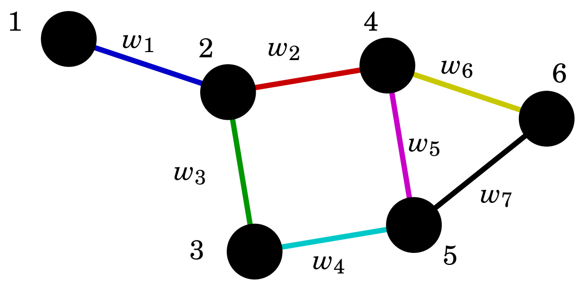


Figure 4.2: A small, undirected graph with 6 nodes and 7 edges.

Example 4.1. *Now that convergence of the weight adaptation law has been proven for fixed q, we present a simple example on a small, randomly chosen, undirected graph, illustrated in Figure 4.2, with edges colour-coded.*

Arbitrarily chosen weighted degree constraints are enforced on the nodes:

$$\boldsymbol{\kappa} = [2, 3, 2, 4, 5, 2]^T$$

The static barrier algorithm is run, discretised for the purposes of simulation using the forward Euler method, with a time step of 1×10^{-2}, and the results are shown in Figure 4.3, with the effect of increasing the parameter q on the algebraic connectivity in the steady state shown in Figure 4.4. As q increases, the algebraic connectivity of the graph at $\mathbf{w}^*$ tends towards the optimal value, $\lambda_2(\mathbf{w}_{\text{opt}}) = 0.8$, verified using the SDP formulation, (4.4), [77]. The trajectory that $\mathbf{w}^*$ follows as q increases, originates at the analytic centre of the optimisation problem, constraining the weights to the feasible region $\mathcal{W}$, and follows the path of centres towards the optimal solution of the problem as $q \to \infty$, as can be seen in Figure 4.4.

To illustrate the effect of changing the control parameters, the simulation is run again, in Figure 4.5, with a lower damping value of $c_1 = 0.5$. All other simulation parameters remain constant at $k_a = 1$, $q = 10$.

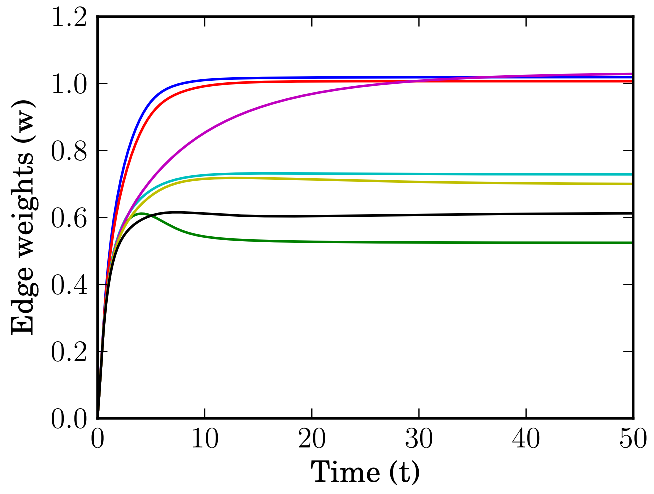


Figure 4.3: Edge weights are modified in time using steepest descent with static logarithmic barriers. The parameters for this simulation are given: $k_a = 1$, $c_1 = 2$, $q = 10$.

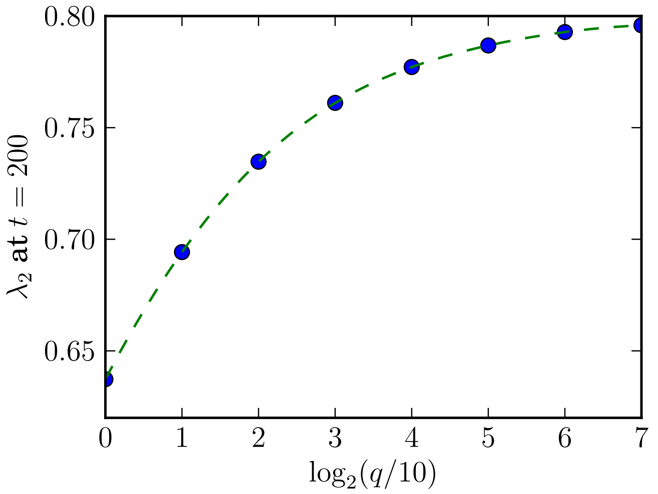


Figure 4.4: Values of λ_2 at $t = 200$ (when $\mathbf{w}$ has settled to $\mathbf{w}^*$) for different values of q. Again, $k_a = 1$, $c_1 = 2$.

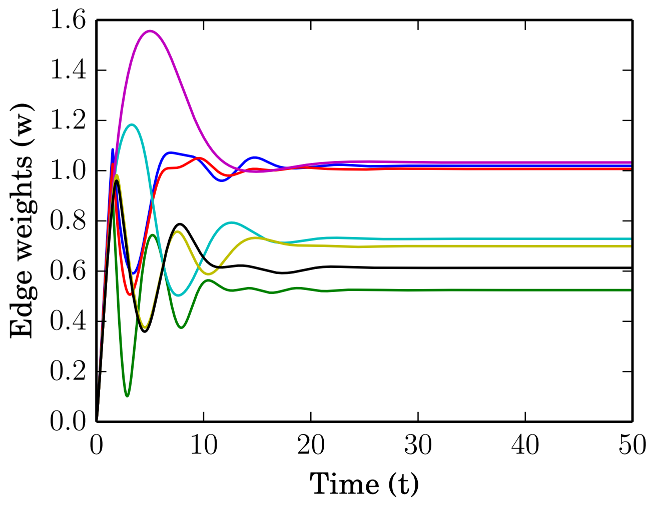


Figure 4.5: Running the simulation with a lower damping parameter on the edge weights, we see that undesirable oscillations are present. It can be seen that the frequency of each edge weight oscillation is different.

4.2.3 Adaptive Logarithmic Barriers

In Lemma 4.4, it was shown that for $\mathbf{w}^*$ to approach the solution of the optimisation problem, $\mathbf{w}_{\text{opt}}$, we require that $q \to \infty$. However, it can be impractical simply to set q very large initially. For example, in the numerical implementation of the algorithm, if q is very large, such that the barriers are excessively steep, and $\dot{\mathbf{w}}$ is sufficiently large, the edge weights can move in one time-step outside of the feasible region, where the gradient of the objective function is no longer properly defined. To overcome this problem, we propose to adapt the severity of the penalty from the barrier functions, increasing q as edge weights approach the critical point for any given q, $\mathbf{w} \to \mathbf{w}^*(q)$; equivalently, $\nabla_{\mathbf{w}} F(\mathbf{w}, q) \to \mathbf{0}$.

Specifically, we propose the following adaptive method which increases the severity of the logarithmic barriers, as the global minimum for given q is approached. Namely, we choose

$$\ddot{q} = \frac{k_b}{||\nabla F||_2 + k_d} - c_2 \dot{q}, \; \dot{q}(0) = 0, \; q(0) = 1, \; k_b, k_d, c_2 > 0 \tag{4.19}$$

where the constant parameter, k_d, limits $\ddot{q}$ by removing the singularity when the gradient of the objective function becomes zero. The control parameters k_b and c_2 control the strength of the driving force on q and its damping respectively. Note that (4.19) guarantees that, for positive control parameters k_b, c_2 and k_d, $q(t)$ is a monotonically increasing function that grows without bound. Intuitively, (4.19) can be seen as a dynamical system describing a mass resting on a surface with viscous friction and a strictly positive, time-varying external force applied. Since there is no mechanism to restore the position of the mass to a previously visited location, it is apparent that $q(t)$ is a monotonically increasing function diverging to $+\infty$.

Lemma 4.5. *The adapted value $q(t)$ defined by (4.19) over the interval $[0, \infty)$ is a monotonically increasing function diverging to $+\infty$ as $t \to +\infty$.*

Proof. It is sufficient to show that $\dot{q}(t) > 0$, $\forall t \in [0, \infty)$ and that $q(t)$ cannot be bounded from above. For the first part of the claim, we use the substitution $\dot{q}(t) = r(t)$, and $p(t) = \frac{k_b}{||\nabla_{\mathbf{w}} F||_2 + k_d}$:

$$\dot{r}(t) + c_2 r(t) = p(t) > 0$$

Taking the Laplace transform,

$$R(s) = \frac{1}{s + c_2} r(0) + \frac{1}{s + c_2} P(s)$$
$$r(t) = r(0) e^{-c_2 t} + \left\{ e^{-c_2 t} \right\} * \left\{ p(t) \right\}$$

As the convolution of two non-negative functions is also non-negative, we can write:

$$r(t) > r(0) e^{-c_2 t} > 0 \quad \forall r(0) \geq 0, t \in [0, \infty) \tag{4.20}$$

hence $q(t)$ is monotonically increasing. Now, for the purpose of contradiction, assume that $q(t)$ were bounded by some value q_{max}. As $q(t)$ is a monotonically increasing function this would imply $q(t) \to q_{max}$ and $\dot{q}(t) \to 0$ as $t \to \infty$. By the stability argument given in Theorem 4.2 for fixed q, we reason that in the limit, $||\nabla F||_2 \to 0$, thus $\ddot{q}$ becomes large and positive. Therefore, $\dot{q}(t) \not\to 0$ contradicting the initial assumption that $q(t)$ is bounded. □

By combining the weight adaptation by direction of steepest descent, as described in (4.9) so that $\mathbf{w} \to \mathbf{w}^*$, with the adaptive logarithmic barrier severity given in (4.19), so that $\mathbf{w}^* \to \mathbf{w}_{\text{opt}}$, we can formulate an adaptive weight update law that causes edge weights to converge on the solution to the optimisation problem.

Theorem 4.3. *The adaptive algorithm for* $\mathbf{w}$ *given by* (4.9) *with the adaptive barrier weight* $q(t)$ *given by* (4.19) *guarantees that,*

$$\lim_{t\to\infty} \mathbf{w} = \mathbf{w}_{\text{opt}} \tag{4.21}$$

Proof. We can use a similar Lyapunov function to prove asymptotic stability onto the optimal solution to the problem, $\mathbf{w}_{\text{opt}}$, for the case where the barrier weight $q(t)$ is adapted according to (4.19). We redefine the error $\boldsymbol{\psi}$ to be the difference between the current edge weights and the optimal edge weights $\boldsymbol{\psi} = \mathbf{w} - \mathbf{w}_{\text{opt}}$. The optimal edge weights are simply $\mathbf{w}_{\text{opt}} = \lim_{q\to\infty} \mathbf{w}^*$, in accordance with Lemma 4.4. To keep with the normal definition of Lyapunov functions, we set $\chi := \frac{1}{q}$, so that asymptotic convergence onto $\mathbf{w}_{\text{opt}}$ occurs for $\chi \to 0$. We consider the candidate Lyapunov function defined as:

$$V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) = k_a(F(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - \overbrace{F(\mathbf{w}_{\text{opt}})}^{-\lambda_2(\mathbf{w}_{\text{opt}})}) + \frac{1}{2}(||\dot{\boldsymbol{\psi}}||_2)^2 - k_a \chi \overbrace{\min_{\boldsymbol{\gamma}\in\mathscr{W}} \left\{ (\Phi_1(\boldsymbol{\gamma}) + \Phi_2(\boldsymbol{\gamma})) \right\}}^{\phi} \tag{4.22}$$

$$= k_a \Big(\lambda_2(\mathbf{w}_{\text{opt}}) - \lambda_2(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) + \chi \big(\Phi_1(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) + \Phi_2(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - \phi \big) \Big) + \frac{1}{2}(||\dot{\boldsymbol{\psi}}||_2)^2 \tag{4.23}$$

which is positive definite[2] for $\chi \geq 0$ and $V(\mathbf{0}) = 0$. Differentiating with respect to time, we find:

$$\begin{aligned} \dot{V}(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) &= \dot{\boldsymbol{\psi}}^T \cdot \frac{\partial V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi)}{\partial \boldsymbol{\psi}} + \ddot{\boldsymbol{\psi}}^T \cdot \frac{\partial V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi)}{\partial \dot{\boldsymbol{\psi}}} + \dot{\chi} \frac{\partial V(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi)}{\partial \chi} \\ &= k_a \dot{\boldsymbol{\psi}}^T \cdot \nabla F(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - k_a \dot{\boldsymbol{\psi}}^T \cdot \nabla F(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - c_1(||\dot{\boldsymbol{\psi}}||_2)^2 \\ &\quad \cdots + k_a \dot{\chi} \big(\Phi_1(\mathbf{s} + \mathbf{w}_{\text{opt}}) + \Phi_2(\mathbf{s} + \mathbf{w}_{\text{opt}}) - \phi \big) \\ &= -c_1(||\dot{\boldsymbol{\psi}}||_2)^2 + k_a \dot{\chi} \big(\Phi_1(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) + \Phi_2(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - \phi \big) \end{aligned} \tag{4.24}$$

From our conclusions that $q(t)$ is a monotonically increasing $\dot{q} > 0$, non-negative function, where $\dot{q} \not\to 0$, it is clear that $\dot{\chi} \leq 0$:

$$\frac{d\chi}{dt} = \frac{d\chi}{dq}\frac{dq}{dt} = -\frac{1}{q^2}\dot{q} \leq 0, \ \forall t \geq 0 \tag{4.25}$$

We have also chosen ϕ such that $\big(\Phi_1(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) + \Phi_2(\boldsymbol{\psi} + \mathbf{w}_{\text{opt}}) - \phi \big) \geq 0$, $\forall \boldsymbol{\psi}$, thus $\dot{V}$ is negative semi-definite.

[2]Subtraction of the (possibly negative) constant ϕ ensures the positive definiteness of V.

Again, we can appeal to the Krasovskii-Lasalle priniciple, as the system is autonomous. Specifically, we consider the set where the derivative of the Lyapunov function is zero:

$$\begin{aligned} S &= \{(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) : \dot{V}(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) = 0\} \\ &\equiv \{(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) : \dot{\boldsymbol{\psi}} = \mathbf{0}, \chi = 0\} \end{aligned} \tag{4.26}$$

We observe that S contains no trajectories of the system save the trivial trajectory $(\boldsymbol{\psi}, \dot{\boldsymbol{\psi}}, \chi) = \mathbf{0}$. As before, $\ddot{\boldsymbol{\psi}} \neq \mathbf{0}$, $\forall \boldsymbol{\psi} \neq \mathbf{0}$, so any point defined by $\dot{\boldsymbol{\psi}} = \mathbf{0}$, $\chi = 0$, $\boldsymbol{\psi} \neq \mathbf{0}$ leaves the set S, fulfilling the conditions of the Krasovskii-Lasalle principle. Therefore, the system is asymptotically stable on $\boldsymbol{\psi} = \mathbf{0}$, $\chi = 0$, $\mathbf{w} = \mathbf{w}_{\text{opt}}$. □

Example 4.2. *We now numerically implement the adaptive barrier algorithm using the same example graph and degree constraints as illustrated in Example 4.1; results are presented in Figures 4.6 and 4.7. Again, for the simulation, the continuous time equations are discretised using the forward Euler method and time step of* 1×10^{-2}*. An interesting feature to note is that the weight* w_7 *has gone to zero, therefore the algorithm is suggesting that the edge* $\{5, 6\}$ *is redundant and can be removed from the network with no loss to algebraic connectivity. Figure 4.8 shows the graph and its optimal edge weights; it is evident that nodes 2, 3, 4, and 6 are the limiting nodes (bottlenecks) due to bandwidth constraints. Such bounds would need to be relaxed for faster average consensus under a linear regime.*

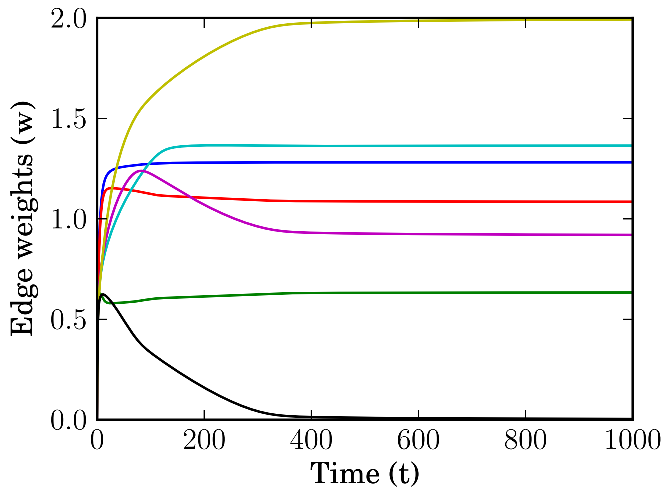


Figure 4.6: Edge weights change in time with dynamics dictated by the adaptive logarithmic barrier algorithm. Parameters for this simulation are: $k_a = k_b = 1$, $c_1 = c_2 = 2$, $k_d = 0.01$, $q(0) = 10$. At $t = 1000$: $\mathbf{w} \approx [1.281, , 1.085, 0.633, 1.365, 0.920, 1.992, 0.004]^T$, $\chi \approx 2.38 \times 10^{-5}$.

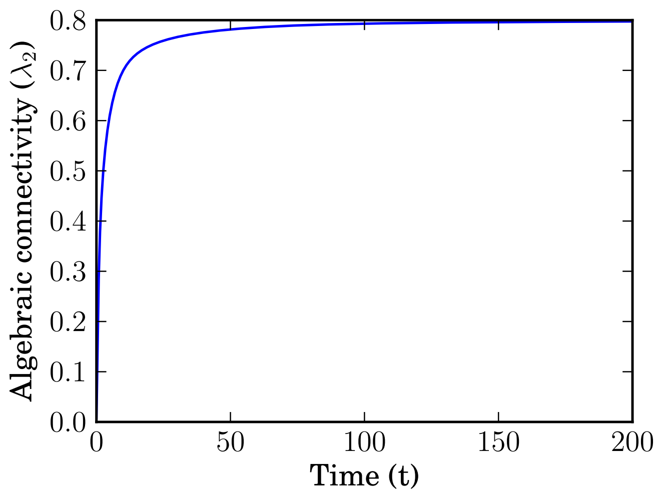


Figure 4.7: Evolution of the algebraic connectivity of the graph in time as weights are adapted. The trajectory until $t = 200$ is shown, with λ_2 at $t = 200$ being 0.7974 to 4 d.p. At $t = 1000$ the connectivity has progressed to 0.7997 to 4 d.p. Algebraic connectivity approaches the optimal value much more rapidly than the slowest edge weights, indicating that some edges play a far more vital role in determining connectivity of the network.

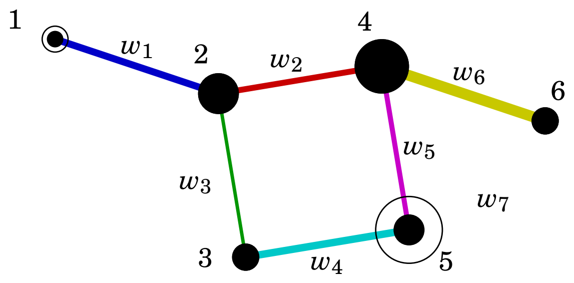


Figure 4.8: The state of the weighted graph at $t = 1000$ (near optimal). Line widths are proportional to edge weights, the radius of the outer circle on each node is proportional to the maximum allowable weighted degree, and the radius of the inner, filled circle is proportional to the current weighted degree. Completely filled circles indicate that the node is a throttle on the connectivity of the network.

It can be seen in Figure 4.8 that each edge has at least one incident node that is a throttle in accordance with Lemma 4.3. It can also be clearly be seen that the edge weight w_7 has tended to zero implying the edge $\{5,6\}$ is redundant. Furthermore, it is evident that node 5 has a great deal of spare capacity, while its neighbouring nodes are all bottlenecked. In a scenario where we could design the maximum allowable weighted degree at each node, it would prove beneficial to increase weighted degree at nodes 3, 4 and 6, at the expense of node 5.

To compare the effect of changing the parameters k_b and c_2, we run the same simulation again first with $k_a = 1, k_b = 0.2, c_1 = c_2 = 2, k_d = 0.01$ and $q(0) = 10$, and secondly with $k_a = 1, k_b = 1, c_1 = 2, c_2 = 0.5, k_d = 0.01$ and $q(0) = 10$. All other simulation parameters remain constant.

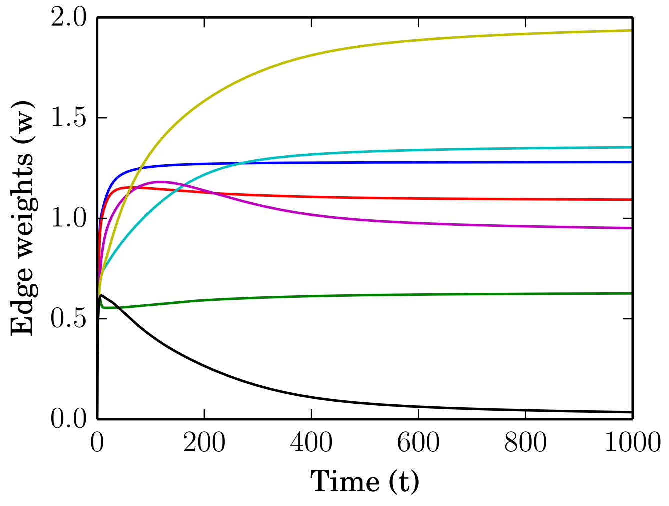


Figure 4.9: It can be seen that reducing the control parameter k_b has the effect of slowing the rate at which q grows, and hence edge weights approach the optimal values more slowly.

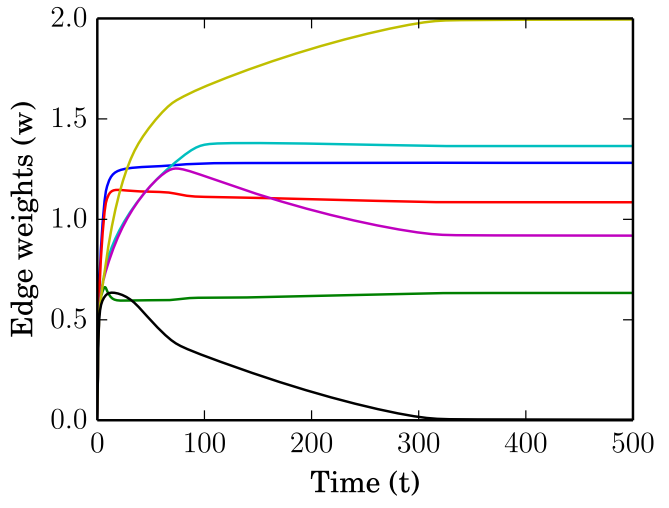


Figure 4.10: By reducing the control parameter c_2, damping on the severity of the logarithmic barriers is reduced, and thus q may grow much more quickly. This is reflected in the edge weights approaching optimal values more rapidly. However, for the same reason that q cannot be set immediately very high in the static logarithmic barrier case, making c_2 too small can result in numerical problems, such as edge weights leaving the feasible region.

4.3 Distributed implementation

It can be seen in Equations (4.9) and (4.19) that in the adaptation law, not all variables are local to each edge and node. Specifically the partial derivative of the algebraic connectivity, $\frac{\partial \lambda_2}{\partial w_i}$, is a function of all edge weights in the network, and this variable appears as part of $\frac{\partial}{\partial \mathbf{w}}F(\mathbf{w},\mathbf{q})$ in Equation (4.9), and $||\frac{\partial}{\partial \mathbf{w}}F||$ in Equation (4.19), which determines, q, the severity of the logarithmic barriers.

Therefore, to implement this algorithm in a fully decentralised manner, decentralised estimates of the derivative of the algebraic connectivity and the severity of the logarithmic barriers q must be made. One possible solution so that a decentralised estimate of $||\frac{\partial}{\partial \mathbf{w}}F||$ need not also be made, is to associate with each edge $\{i,j\}$ an individual $q_{\{i,j\}}$, which grows as a function of the local derivative of $|\frac{\partial F(\mathbf{w},q)}{\partial w_{\{i,j\}}}|$. As such, we now have:

$$\ddot{q}_{\{i,j\}} = \frac{k_b}{\left|\frac{\partial F(\mathbf{w},q)}{\partial w_{\{i,j\}}}\right| + k_d} - c_2 \dot{q}_{\{i,j\}},\ \dot{q}_{\{i,j\}}(0) = 0,\ q_{\{i,j\}}(0) > 0,\ k_b, k_d, c_2 > 0 \tag{4.27}$$

with $\frac{\partial F(\mathbf{w},q)}{\partial w_{\{i,j\}}}$, as given previously in Equation (4.12). These local $q_{\{i,j\}}$ dynamics can be seen as a local analogue to Equation (4.19). The only term in $\frac{\partial F(\mathbf{w},q)}{\partial w_{\{i,j\}}}$, in (4.9) and (4.27) which remains a function of all edge weights, and hence non-local, is the partial derivative of the algebraic connectivity with respect to each edge weight. Previously it was shown in Section 3.3.1 and its subsections, that a local estimate of this quantity can be made using the decentralised algebraic connectivity estimator of [29]. Henceforth, we can implement this decentralised algebraic connectivity estimator, replacing the function $\frac{\partial \lambda_2}{\partial w_{\{i,j\}}}$, with the local estimate $\frac{(a_i - a_j)^2}{n y_{\mathbf{a}^{\circ 2},i}}$ from (3.20) and the fully decentralised system (3.15). Note that as we are only interested in the direction of the gradient of λ_2, the denominator of the estimate (3.20) may be omitted. Substituting (3.15) and (3.20), into (4.27) and (4.9), we arrive at the complete set of fully decentralised equations:

$$\begin{aligned}
\ddot{w}_{\{i,j\}} &= k_a\left((a_i - a_j)^2 + \frac{1}{q_{\{i,j\}}}\left(\frac{1}{w_{\{i,j\}}} - \frac{1}{\kappa_i - \sum_{k\in\mathcal{N}_i} w_{\{i,k\}}} - \frac{1}{\kappa_j - \sum_{k\in\mathcal{N}_j} w_{\{j,k\}}}\right)\right) - c_1 \dot{w}_{\{i,j\}}\\
\ddot{q}_{\{i,j\}} &= \frac{k_b}{\left|(a_i - a_j)^2 + \frac{1}{q_{\{i,j\}}}\left(\frac{1}{w_{\{i,j\}}} - \frac{1}{\kappa_i - \sum_{k\in\mathcal{N}_i} w_{\{i,k\}}} - \frac{1}{\kappa_j - \sum_{k\in\mathcal{N}_j} w_{\{j,k\}}}\right)\right| + k_d} - c_2 \dot{q}_{\{i,j\}}\\
\dot{a}_i &= -k_1 y_{\mathbf{a},i} + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(a_j - a_i) + k_3(1 - y_{\mathbf{a}^{\circ 2},i})a_i\\
\dot{y}_{\mathbf{a},i} &= k_\gamma(a_i - y_{\mathbf{a},i}) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a},j} - y_{\mathbf{a},i}) + k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_{\mathbf{a},j} - z_{\mathbf{a},i})\\
\dot{z}_{\mathbf{a},i} &= -k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a},j} - y_{\mathbf{a},i})\\
\dot{y}_{\mathbf{a}^{\circ 2},i} &= k_\gamma(a_i^2 - y_{\mathbf{a}^{\circ 2},i}) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a}^{\circ 2},j} - y_{\mathbf{a}^{\circ 2},i}) + k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_{\mathbf{a}^{\circ 2},j} - z_{\mathbf{a}^{\circ 2},i})\\
\dot{z}_{\mathbf{a}^{\circ 2},i} &= -k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_{\mathbf{a}^{\circ 2},j} - y_{\mathbf{a}^{\circ 2},i})
\end{aligned} \tag{4.28}$$

Example 4.3. *In order to demonstrate the efficacy of the fully decentralised algebraic connectivity maximisation system* (4.28)*, we seek to adapt edge weights in a random network of $n = 20$ nodes, where the weighted degree of each node may not exceed the total number of neighbours that each node has: $\kappa_i = \sum_{j\in\mathcal{N}_i} 1$, and no edge weight may be negative. We initially choose each edge weight to be equal to $1-\epsilon$, where $\epsilon << 1$, so that the initial edge weights are in the interior of the feasible set, but total edge weight in the network may only increase by $m\epsilon$, which is small. Thus, we can be sure that any increase in the algebraic connectivity of the network must be due to better distribution of edge weight, rather than the absolute increase of their total value. Again, for this simulation discretisation is performed using the forward Euler method, with a time step of* 2×10^{-3}.

When all edge weights are equal to 1, it is found that the algebraic connectivity $\lambda_2(t=0)\approx 0.3344$. *Edge weights are then adapted according to the system governed by* (4.28)*, and control gains are chosen so that the PI consensus estimator layer* ($k_P = 20,\ k_I = 20,\ k_\gamma = 5$) *converges approximately 5 times faster than the algebraic connectivity estimator* ($k_1 = 5,\ k_2 = 1,\ k_3 = 4$)*, and the algebraic connectivity estimator converges more quickly than the weight adaptation layer* ($k_a = 1,\ k_b = 1,\ c_1 = 1,\ c_2 = 1,\ k_d = 1\times10^{-6}$).

This gives us sufficient separation in time scale between the layers.

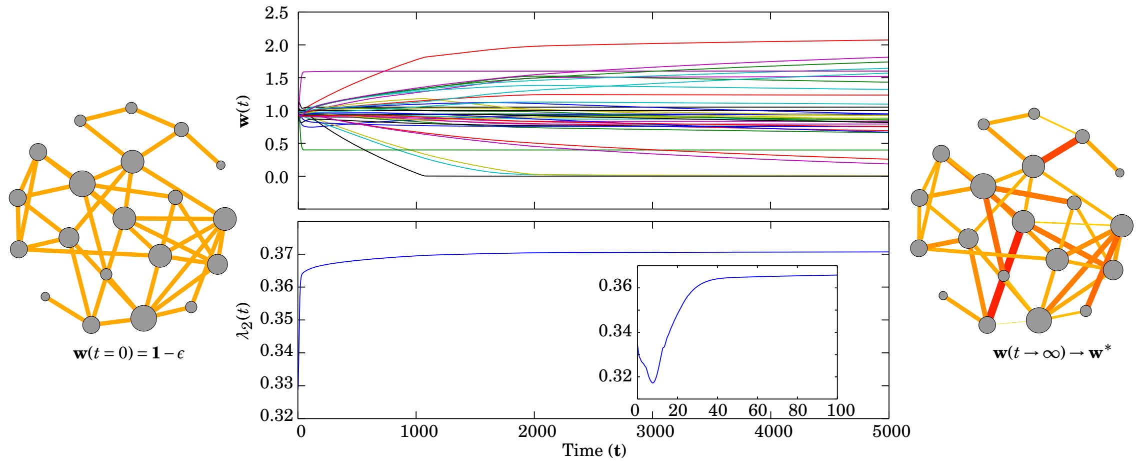


Figure 4.11: Edge weights are adapted according to (4.28) in an entirely decentralised manner to maximise the algebraic connectivity of the network under the condition that no edge weights may be negative, and the weighted degree of each node may not be greater than its (unweighted) degree. In the diagrams showing the initial and end state of the network, node diameter is proportional to maximum allowed weighted degree, κ_i, and edge thickness and colour is proportional to weight, higher weights are redder and thicker.

As edge weights are adapted, shown in Figure 4.11, the algebraic connectivity initially decreases until the estimator layers have properly converged, whereupon it increases rapidly, as the nadir of the potential well of the objective function for a given set of $q_{i,j}$ is reached. Finally, as the $q_{i,j}$ increase and the logarithmic barriers enforcing the feasible set become steeper, the edge weights slowly converge to their optimal values, see Figure 4.11.

It should be noticed that the algebraic connectivity converges more rapidly than the slowest converging edges due to weak sensitivity of λ_2 with respect to some edges. After 5000 seconds of simulated time, the algebraic connectivity has reached a value of $\lambda_2(5000) \approx 0.3707$, which is over 99.9% of the result for optimal λ_2 found using the SDP method described in [77]: $\lambda_2^* \approx 0.3708$. At this time, some of the edges have yet to converge, but it is known that three edges will tend to zero value, and could be removed from the network at no detriment to the algebraic connectivity.

4.4 Weight adaptation to satisfy the KKT conditions

One of the major disadvantages of the previously described weight adaptation law is that the gradients of the logarithmic barrier functions are not defined outside of the feasible set of edge weights. This means that if a perturbation forces the edge weights outside of the feasible set, the system cannot recover. A subtler disadvantage is that because all $q_i(t)$ are monotonically increasing functions, the steepness of the barrier functions can only become more severe. This means that there is a gradual stiffening of the system over time, and this means that without a method for relaxing the barrier functions when topology changes in the network are rapid[3], we may run into a similar problem to setting the values of q_i too large initially. To combat these problems we will now investigate a second, complementary method for adapting the edge weights on-line. The premise of this method is to design a dynamical system so that the stable equilibrium point of the system satisfies the Karush-Kuhn-Tucker conditions, thus the stable equilibrium of the system is the solution to the optimisation problem.

Recall that the optimisation problem, (4.2), is a convex, constrained optimisation problem of the form:

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}} && f(\mathbf{w}) \\ &\text{subject to} && g_i(\mathbf{w}) \leq 0 \; \forall i \in 1,\dots,p \end{aligned} \tag{4.29}$$

where $f(\mathbf{w}) = -\lambda_2(\mathbf{L}(\mathbf{w}))$, and there are $p = m+n$ affine inequality constraints, $g_i(\mathbf{w})$. The first m constraints enforce the non-negativity of edge weights: $g_i(\mathbf{w}) = -w_i, \forall i \in 1\dots m$. The next n constraints enforce the upper bound on the weighted degree of nodes: $g_{m+i}(\mathbf{w}) = \left(\sum_{j\in\mathcal{N}_i} w_{\{i,j\}}\right) - \kappa_i, \forall i \in 1\dots n$.

As an alternative to the adaptation law based on the use of logarithmic barrier functions described in Section 4.2, here we consider the weight adaptation method based on the set of ordinary differential equations,

$$\dot{w}_i = -\frac{\partial f(\mathbf{w})}{\partial w_i} - \frac{1}{2}\sum_j \frac{\partial g_j(\mathbf{w})}{\partial w_i} v_j^2 \tag{4.30,a}$$

$$\dot{v}_j = g_j(\mathbf{w}) v_j, \quad \forall i \in 1,\dots,m,\ j \in 1,\dots,m+n \tag{4.30,b}$$

[3]It may be interesting to explore this avenue, introducing a mechanism by which q_i may decrease if a rapid change in the geometry of the optimisation problem is detected.

This system is designed so that the stable equilibrium point satisfies the first-order necessary Karush-Kuhn-Tucker (KKT) conditions [109, 110, 105]. In this equation, the variables $\frac{1}{2}\nu_j^2$ behave as KKT multipliers, the dual variables, where the edge weights w_i are the primal variables.

The first-order necessary KKT conditions for a local minimum $\mathbf{w}^*$ are simply:

- Stationarity: $\nabla f(\mathbf{w}^*) + \frac{1}{2}\sum_j \nabla g_j(\mathbf{w}^*)\nu_j^{*2} = \mathbf{0}$ which is equivalent to the stationarity of the primal variables $\dot{\mathbf{w}} = \mathbf{0}$ in Equation (4.30,a).

- Dual Feasibility: $\frac{1}{2}\nu_j^{*2} \geq 0$ for all j, which is trivially satisfied in our case as all ν_j are real.

- Primal Feasibility: $g_j(\mathbf{w}^*) \leq 0$ for all j. If we assume that the primal variables are infeasible, i.e. there is some $g_j(\mathbf{w}) > 0$, then ν_j will not be stationary so long as $\nu_j \neq 0$. (In the case that $\nu_j^* = 0$, $g_j(\mathbf{w}^*) > 0$, this stationary point is unstable.)

- Complementary slackness: $g_j(\mathbf{w}^*) \cdot \frac{1}{2}\nu_j^{*2} = 0$ for all j. This is equivalent to the condition that $g_j(\mathbf{w}^*)\nu_j^* = 0$ which is simply stationarity of ν_j, for all j in Equation (4.30,b).

Substituting the appropriate objective function and inequality constraints into (4.30,a)-(4.30,b), we arrive at the weight adaptation law for solving the optimisation problem posed in (4.2):

$$\dot{w}_{\{i,j\}} = \frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}} + \frac{1}{2}\mu_{\{i,j\}}^2 - \frac{1}{2}\nu_i^2 - \frac{1}{2}\nu_j^2,\ \forall \{i,j\} \in \mathscr{E} \tag{4.31}$$

$$\dot{\mu}_{\{i,j\}} = -w_{\{i,j\}}\mu_{\{i,j\}},\ \forall \{i,j\} \in \mathscr{E} \tag{4.32}$$

$$\dot{\nu}_i = \left(\left(\sum_{j \in \mathscr{N}_i} w_{\{i,j\}}\right) - \kappa_i\right)\nu_i,\ \forall i \in \mathscr{V} \tag{4.33}$$

Note 4.2. *For clarity we have chosen to split the $m+n$ KKT multipliers, previously denoted by ν_i, into the m multipliers governing the non-negativity of edge weights, labelling each with the edge to which it is associated, i.e. $\mu_{\{i,j\}}$, and the n multipliers governing the weighted degree at each node, labelled each according to its associated node, ν_i.*

This set of coupled scalar equations can be represented in a more compact matrix differential equation form, using the unoriented incidence matrix $\mathbf{M}$ (introduced previously in Chapter 2) which happens to be equal to $\nabla_{\mathbf{w}}\text{diag}(\mathbf{L})$, and componentwise multiplication and exponentiation of vectors. Componentwise multiplication and exponentiation of vectors is designated using Hadamard notation: $\mathbf{a} = \mathbf{b} \circ \mathbf{c} \Leftrightarrow a_i = b_i c_i,\ \forall i$, $\mathbf{a} = \mathbf{b}^{\circ c} \Leftrightarrow a_i = (b_i)^c,\ \forall i$. The edge weights $w_{\{i,j\}}$ and the variables $\mu_{\{i,j\}}$ are collected in vectors according to an arbitrary labelling function, as described previously in Chapter 2, Definition 2.7. The resulting set of equations in vector form then becomes:

$$\dot{\mathbf{w}} = \frac{\partial \lambda_2(\mathbf{w})}{\partial \mathbf{w}} + \frac{1}{2}\left(\boldsymbol{\mu}^{\circ 2} - \mathbf{M}\boldsymbol{\nu}^{\circ 2}\right) \tag{4.34}$$

$$\dot{\boldsymbol{\mu}} = -\mathbf{w} \circ \boldsymbol{\mu} \tag{4.35}$$

$$\dot{\boldsymbol{\nu}} = (\mathbf{M}^\top \mathbf{w} - \mathbf{k}) \circ \boldsymbol{\nu} \tag{4.36}$$

This matrix representation of the system of coupled differential equations will prove to be useful in further analysis.

Lemma 4.6. *For the optimisation problem defined in* (4.2) *the first order necessary KKT conditions are also sufficient.*

Proof. Proof follows from the fact that the primal problem is convex and strictly feasible, thus Slater's condition holds. See [110] for details. □

Theorem 4.4. *The equilibrium point* $[\mathbf{w}_{\text{opt}}^\top, \boldsymbol{\mu}_{\text{opt}}^\top, \boldsymbol{\nu}_{\text{opt}}^\top]^\top$ *is locally exponentially stable if* $\nabla^2 \lambda_2(\mathbf{w}_{\text{opt}}) < 0$.

Proof. Consider the generic convex minimisation problem (4.29). If the inequality constraints are such that $g_i(\mathbf{x}) \le 0$, $\forall i, \mathbf{x} \in \mathscr{X}$, and the primal feasible set $\mathscr{X}$ is convex and compact, $f(\mathbf{x})$ is strictly convex, there exists a unique optimal point $\mathbf{x}_{\text{opt}}$. At this point some $r_1 \ge 1$ constraints will be tight (as $f(\mathbf{x})$ is monotonically decreasing we can be sure that at least one inequality will be tight), and some $r_2 \ge 1$ inequality constraints may be slack (there must be at least one slack inequality; by contradiction, if all were tight then $\mathbf{w}_{\text{opt}} = \mathbf{0}$ and thus $\lambda_2 = 0$ which is infeasible).

We can divide these tight and slack inequalities into strict inequalities $p_j(\mathbf{x}) < 0$ and equality constraints $h_j(\mathbf{x}) = 0$ by redefining:

$$\forall i \text{ s.t. } g_i(\mathbf{x}_{\text{opt}}) = 0, h_j(\mathbf{x}) \triangleq g_i(\mathbf{x}), j = 1, \ldots, r_1 \tag{4.37}$$

$$\forall i \text{ s.t. } g_i(\mathbf{x}_{\text{opt}}) < 0, p_j(\mathbf{x}) \triangleq g_i(\mathbf{x}), j = 1, \ldots, r_2 \tag{4.38}$$

Now the equivalent optimisation problem can be formulated:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & h_i(\mathbf{x}) = 0, \forall i = 1 \ldots r_1 \\ & p_i(\mathbf{x}) < 0, \forall i = 1 \ldots r_2, \\ & \text{where } r_1 + r_2 = m + 1 \end{aligned} \tag{4.39}$$

The solution to this optimisation problem can be obtained from the following set of ODEs:

$$\dot{\mathbf{x}} = -\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} - \frac{1}{2}\sum_i \frac{\partial h_i(\mathbf{x})}{\partial \mathbf{x}} \alpha_i^2 - \frac{1}{2}\sum_i \frac{\partial p_i(\mathbf{x})}{\partial \mathbf{x}} \beta_i^2 \tag{4.40}$$

$$\dot{\boldsymbol{\alpha}} = \mathbf{h}(\mathbf{x}) \circ \boldsymbol{\alpha}, \quad \boldsymbol{\alpha}(t=0) > 0 \tag{4.41}$$

$$\dot{\boldsymbol{\beta}} = \mathbf{p}(\mathbf{x}) \circ \boldsymbol{\beta}, \quad \boldsymbol{\beta}(t=0) > 0 \tag{4.42}$$

Linearisation of the system about the optimal point where, $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}_{\text{opt}}$, $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha} - \boldsymbol{\alpha}_{\text{opt}}$, $\tilde{\boldsymbol{\beta}} = \boldsymbol{\beta} - \boldsymbol{\beta}_{\text{opt}}$, is:

$$\begin{bmatrix} \dot{\tilde{\mathbf{x}}} \\ \dot{\tilde{\boldsymbol{\alpha}}} \\ \dot{\tilde{\boldsymbol{\beta}}} \end{bmatrix} \approx \mathbf{J_0} \begin{bmatrix} \tilde{\mathbf{x}} \\ \tilde{\boldsymbol{\alpha}} \\ \tilde{\boldsymbol{\beta}} \end{bmatrix} \tag{4.43}$$

$$\mathbf{J_0} = \begin{bmatrix} -\nabla^2 f(\mathbf{x}) - \frac{1}{2}\sum_i \alpha_i^2 \nabla^2 h_i(\mathbf{x}) - \frac{1}{2}\sum_i \beta_i^2 \nabla^2 p_i(\mathbf{x}) & -(\text{diag}\{\boldsymbol{\alpha}\}\nabla \mathbf{h}(\mathbf{x}))^\top & -(\text{diag}\{\boldsymbol{\beta}\}\nabla \mathbf{p}(\mathbf{x}))^\top \\ \text{diag}\{\boldsymbol{\alpha}\}\nabla \mathbf{h}(\mathbf{x}) & \text{diag}\{\mathbf{h}(\mathbf{x})\} & \mathbf{0} \\ \text{diag}\{\boldsymbol{\beta}\}\nabla \mathbf{p}(\mathbf{x}) & \mathbf{0} & \text{diag}\{\mathbf{p}(\mathbf{x})\} \end{bmatrix} \tag{4.44}$$

evaluated at $\mathbf{x} = \mathbf{x}_{\text{opt}}$, $\boldsymbol{\alpha} = \boldsymbol{\alpha}_{\text{opt}}$, $\boldsymbol{\beta} = \boldsymbol{\beta}_{\text{opt}}$, which gives:

(4.45)

$$\mathbf{J_0} = \begin{bmatrix} -\nabla^2 f(\mathbf{x}_{\text{opt}}) - \frac{1}{2}\sum_i \alpha_{opt,i}^2 \nabla^2 h_i(\mathbf{x}_{\text{opt}}) & -(\text{diag}\{\boldsymbol{\alpha}_{\text{opt}}\}\nabla \mathbf{h}(\mathbf{x}_{\text{opt}}))^\top & \mathbf{0} \\ \text{diag}\{\boldsymbol{\alpha}_{\text{opt}}\}\nabla \mathbf{h}(\mathbf{x}_{\text{opt}}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \text{diag}\{\mathbf{p}(\mathbf{x}_{\text{opt}})\} \end{bmatrix} = \begin{bmatrix} \mathbf{J_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{J_2} \end{bmatrix}$$

as $\boldsymbol{\beta}_{\text{opt}} = \mathbf{0}$ and $\mathbf{h}(\mathbf{x}_{\text{opt}}) = \mathbf{0}$.

It is evident that this Jacobian matrix has a block diagonal structure, indicating the decoupling of the dual variables whose associated inequality constraints are slack at the optimal point. Looking first at the $\mathbf{J_2}$ block, in the vicinity of the optimal point, the square root of the (redundant) slack dual variables β_i will decay to zero exponentially fast with rate constants $p_i(\mathbf{x}_{\text{opt}}) < 0$. That is, the 'more slack' the constraint is, the faster its dual variable will decay away.

The $\mathbf{J_1}$ block is more interesting and has the form:

$$\mathbf{J_1} = \begin{bmatrix} \mathbf{A} & -\mathbf{B} \\ \mathbf{B}^\top & \mathbf{0} \end{bmatrix} \tag{4.46}$$

$$\mathbf{A} = -\nabla^2 f(\mathbf{x}_{\text{opt}}) - \frac{1}{2}\sum_i \alpha_{opt,i}^2 \nabla^2 h_i(\mathbf{x}_{\text{opt}})$$

$$\mathbf{B} = (\text{diag}\{\boldsymbol{\alpha}_{\text{opt}}\}\nabla \mathbf{h}(\mathbf{x}_{\text{opt}}))^\top$$

In the vicinity of the optimal point, the primal variables $\mathbf{x}$ will converge to $\mathbf{x}_{\text{opt}}$ exponentially fast provided $\mathbf{A}$ is negative definite. This follows from the following Lemma.

□

Lemma 4.7. *The matrix* $\mathbf{J_1} = \begin{bmatrix} \mathbf{A} & -\mathbf{B} \\ \mathbf{B}^\top & \mathbf{0} \end{bmatrix}$ *has eigenvalues with negative real part provided the symmetric part of* $\mathbf{A}$ *is negative definite and* $\mathbf{B}$ *is full column rank.*

Proof.

$$\det\{\mathbf{J_1}\} = \det\{\mathbf{A}\}\det\{\mathbf{B}^\top \mathbf{A}^{-1}\mathbf{B}\} \tag{4.47}$$

It can be seen that $\det\{\mathbf{A}\} \neq 0$ as $\mathbf{A}$ is nonsingular, and thus $\mathbf{A}^{-1}$ exists. Furthermore, $\det\{\mathbf{B}^\top \mathbf{A}^{-1}\mathbf{B}\}$ is only full rank if $\mathbf{B}$ is full column rank, i.e. $\mathbf{B}^\top\mathbf{B}$ is invertible. Therefore, $\det\{\mathbf{J_1}\} \neq 0$; $\mathbf{J_1}$ has no zero eigenvalues.

Considering the eigenvector equation $\mathbf{J_1 v_i} = \lambda_i \mathbf{v_i}$, with $\mathbf{v_i} \triangleq [\mathbf{x_i}^\top, \mathbf{y_i}^\top]^\top$, we have:

$$\begin{bmatrix} \mathbf{A} & -\mathbf{B} \\ \mathbf{B}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x_i} \\ \mathbf{y_i} \end{bmatrix} = \lambda_i \begin{bmatrix} \mathbf{x_i} \\ \mathbf{y_i} \end{bmatrix} \tag{4.48}$$

$$\mathbf{A x_i} - \mathbf{B y_i} = \lambda_i \mathbf{x_i} \tag{4.49}$$

$$\mathbf{B}^\top \mathbf{x_i} = \lambda_i \mathbf{y_i} \tag{4.50}$$

Looking solely at Equation (4.50), we can see that if $\mathbf{x_i} = \mathbf{0}$, then either $\lambda_i = 0$, which we have previously shown not to be true, or $\mathbf{y_i} = \mathbf{0}$, so that $\mathbf{v_i} = \mathbf{0}$ which is simply the trivial solution. Therefore, we can conclude that for all eigenvectors of $\mathbf{J_1}$, $\mathbf{x_i} \neq \mathbf{0}$.

Now we drop the subscripts and assume that $\mathbf{v}$ is a unit eigenvector so that

$$\lambda = \mathbf{v}^* \mathbf{J_1} \mathbf{v} \tag{4.51}$$

$$= [\mathbf{x}^*, \mathbf{y}^*] \begin{bmatrix} \mathbf{A} & -\mathbf{B} \\ \mathbf{B}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \tag{4.52}$$

$$= \mathbf{x}^* \mathbf{A x} - \mathbf{x}^* \mathbf{B y} + \mathbf{y}^* \mathbf{B}^\top \mathbf{x} \tag{4.53}$$

$$= \mathbf{x}^* \mathbf{A x} - \mathbf{x}^* \mathbf{B y} + (\mathbf{y}^* \mathbf{B}^\top \mathbf{x})^\top \tag{4.54}$$

$$= \mathbf{x}^* \mathbf{A x} - \mathbf{x}^* \mathbf{B y} + \mathbf{x}^\top \mathbf{B} \overline{\mathbf{y}} \tag{4.55}$$

$$= \mathbf{x}^* \mathbf{A x} - \mathbf{x}^* \mathbf{B y} + \overline{\mathbf{x}^* \mathbf{B y}} \tag{4.56}$$

$$= \mathbf{x}^* \mathbf{A x} - 2\Im(\mathbf{x}^* \mathbf{B y}) \cdot i, \quad i = \sqrt{-1} \tag{4.57}$$

As $\mathbf{x} \neq \mathbf{0}$ and the symmetric part of $\mathbf{A}$ is negative definite, $\Re(\mathbf{x}^* \mathbf{A x}) < 0$. This is the only contribution to the real part of the eigenvalue, thus $\mathbf{J_1}$ is a stable matrix. □

Note 4.3. *If $f(\mathbf{x})$ and all $h_i(\mathbf{x})$ are convex, then it is apparent that $\mathbf{A}$ is negative definite only if $f(\mathbf{x})$ or any $h_i(\mathbf{x})$ is strictly convex near $\mathbf{x}_{\text{opt}}$. As all inequality constraints in this example are affine, the condition that $f(\mathbf{x})$ or any $h_i(\mathbf{x})$ is strictly convex near $\mathbf{x}_{\text{opt}}$ simplifies to the necessary and sufficient condition that $\nabla^2 \lambda_2(\mathbf{w}_{\text{opt}}) < 0$.*

4.4.1 Enforcing Strict Convexity

In Note 4.3, it was discussed that we require $\lambda_2(\mathbf{w})$ to be strictly concave in the vicinity of the optimal point for exponential convergence. However, $-\lambda_2(\mathbf{w})$ is not necessarily strictly convex. To account for this we choose to minimise a modified function $f(\mathbf{x}) = (\zeta - \lambda_2(\mathbf{w}))^2$, where ζ is a constant, greater than $\lambda_2(\mathbf{w}_{\text{opt}})$, so that the modified function is positive. It is apparent that minimising $(\zeta - \lambda_2(\mathbf{w}))^2$ is equivalent to maximising λ_2 when $\zeta - \lambda_2 > 0$, so the two optimisation problems are equivalent.

In the following Lemma, we show that as $\zeta - \lambda_2(\mathbf{w})$ is positive and convex (near $\mathbf{w}_{\text{opt}}$), then its square will be strictly convex in the vicinity of the optimal point if that optimal point is distinct.

Lemma 4.8. *If $f(\mathbf{x})$ is (not necessarily strictly) convex and positive, then $f(\mathbf{x})^2$ is strictly convex over non-flat regions.*

Proof. $f(\mathbf{x})$ is convex so Jensen's inequality holds:

$$f(\theta \mathbf{x_1} + (1-\theta)\mathbf{x_2}) \le \theta f(\mathbf{x_1}) + (1-\theta) f(\mathbf{x_2}),\ \forall \theta \in (0,1) \tag{4.58}$$

Squaring both sides, so long as $f(\mathbf{x})$ is non-negative, we can be sure that

$$f(\theta \mathbf{x_1} + (1-\theta)\mathbf{x_2})^2 \le \theta^2 f(\mathbf{x_1})^2 + 2\theta(1-\theta) f(\mathbf{x_1}) f(\mathbf{x_2}) + (1-\theta)^2 f(\mathbf{x_2})^2,\ \forall \theta \in (0,1) \tag{4.59}$$

But to prove that $f(\mathbf{x})^2$ is strictly convex, we need to show that:

$$f(\theta \mathbf{x_1} + (1-\theta)\mathbf{x_2})^2 < \theta f(\mathbf{x_1})^2 + (1-\theta) f(\mathbf{x_2})^2,\ \forall \theta \in (0,1) \tag{4.60}$$

By considering the right-hand side of (4.59) and (4.60), strict convexity follows if the following inequality holds:

$$\theta^2 f(\mathbf{x_1})^2 + 2\theta(1-\theta) f(\mathbf{x_1}) f(\mathbf{x_2}) + (1-\theta)^2 f(\mathbf{x_2})^2 < \theta f(\mathbf{x_1})^2 + (1-\theta) f(\mathbf{x_2})^2 \tag{4.61}$$

$$(\theta^2 - \theta) f(\mathbf{x_1})^2 + 2\theta(1-\theta) f(\mathbf{x_1}) f(\mathbf{x_2}) + ((1-\theta)^2 - (1-\theta)) f(\mathbf{x_2})^2 < 0 \tag{4.62}$$

$$\theta(\theta - 1)(f(\mathbf{x_1}) - f(\mathbf{x_2}))^2 < 0 \tag{4.63}$$

which clearly holds in the interval so long as $f(\mathbf{x_1}) \neq f(\mathbf{x_2})$.

Note 4.4. *If $f(\mathbf{x_1}) = f(\mathbf{x_2})$, then there may be a manifold of optimal edge weights. In this case, $(\zeta - \lambda_2(\mathbf{w}))^2$ will not be strictly convex in this region. Instead, the set of optimal points will be a neutrally stable manifold for the set of ODEs, but local convergence to this manifold will be exponentially fast.*

□

Substituting in the altered objective function $f(\mathbf{x}) = (\zeta - \lambda_2(\mathbf{w}))^2$ in Equation (4.29), we arrive at the modified weight adaptation law which guarantees that the equilibrium point is locally exponentially stable.

$$\begin{aligned} \dot{w}_{\{i,j\}} &= -\frac{\partial(\zeta - \lambda_2(\mathbf{w}))^2}{\partial w_{\{i,j\}}} + \frac{1}{2}\left(\mu_{\{i,j\}}^2 + \nu_i^2 + \nu_j^2\right) \\ &= 2(\zeta - \lambda_2(\mathbf{w}))\frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}} + \frac{1}{2}\left(\mu_{\{i,j\}}^2 + \nu_i^2 + \nu_j^2\right) \end{aligned} \tag{4.64}$$

$$\dot{\mu}_{\{i,j\}} = -w_{\{i,j\}}\mu_{\{i,j\}}, \qquad \mu_{\{i,j\}}(0) > 0 \tag{4.65}$$

$$\dot{\nu}_i = \left(\sum_{j\in\mathcal{N}_i} w_{i,j} - \kappa_i\right)\nu_i, \qquad \nu_i(0) > 0 \tag{4.66}$$

where the collection of all ODEs for each node and edge may be written in the more compact form:

$$\dot{\mathbf{w}} = 2(\zeta - \lambda_2(\mathbf{w}))\frac{\partial \lambda_2(\mathbf{w})}{\partial \mathbf{w}} + \frac{1}{2}\left(\boldsymbol{\mu}^{\circ 2} + \mathbf{M}\boldsymbol{\nu}^{\circ 2}\right) \tag{4.67}$$

$$\dot{\boldsymbol{\mu}} = -\mathbf{w} \circ \boldsymbol{\mu} \tag{4.68}$$

$$\dot{\boldsymbol{\nu}} = (\mathbf{M}^\top \mathbf{w} - \mathbf{k}) \circ \boldsymbol{\nu} \tag{4.69}$$

Observing the set of ODEs, Equations (4.64) to (4.66), it is clear to see that these are not fully distributed. The function $\lambda_2(\mathbf{w})$ and its partial derivatives with respect to the edge weights are global functions of all the edge weights. Thus to achieve a fully distributed strategy, we must make distributed estimates of these functions. This is achieved again using the distributed algebraic connectivity estimator from [29], as in Section 4.3, though we omit the full set of decentralised equations for brevity.

Example 4.4. *We now implement the KKT satisfaction weight adaptation system, as described in* (4.64) – (4.66)*, on the same weighted graph presented earlier in Figure 4.2, in order to maximise the algebraic connectivity under the same set of constraints. Again, the distributed algebraic connectivity estimator of [29], and presented in Chapter 3, is utilised with the same control parameters:* $k_1 = 6$, $k_2 = 1$, $k_3 = 5$, $k_P = 20$, $k_I = 10$, $k_\gamma = 5$*, as in Examples 4.1 and 4.2. For this example, it is chosen that* $\nu_i(0) = 1, \mu_{\{i,j\}}(0) = 1, w_{\{i,j\}}(0) = 1, \zeta = 2$*. For this simulation, the continuous time differential equations are again discretised using the forward Euler method with a time step of* 1×10^{-2}.

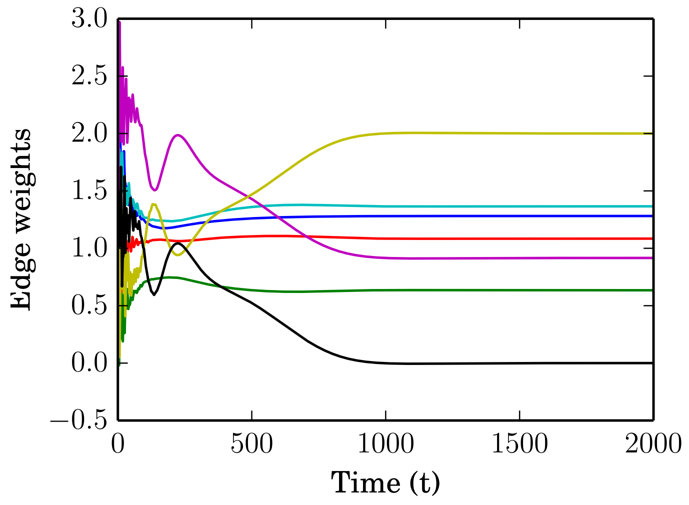


Figure 4.12: Edge weights are adapted using the second presented adaptation system (4.64) – (4.66): that designed to satisfy the first order necessary KKT conditions for the solution to the optimisation problem.

The local exponential convergence of this method allows the optimal solution to be approached more rapidly than in the adaptive logarithmic barrier method, which shows only local asymptotic convergence. However, during the transient period before the dual variables $\mu_{\{i,j\}}$ and ν_i have properly settled, rapid oscillations occur, and this behaviour may be undesirable. In the next section we will address how to improve the transient performance of the convergence of these dual variables. Nevertheless, edge weights do eventually converge onto the optimal values, as can be seen by comparing Figure 4.6 to Figure 4.12.

4.4.2 Improving Performance Using the Ramp Function

Using the original KKT satisfaction weight adaptation rule, Equations (4.30,a) and (4.30,b), we have a choice to make on setting the initial values of the dual variables, ν_j. One issue which may be run into in implementation is that the dual variables ν_i decay away exponentially fast when the constraints are slack, $g_i(\mathbf{w})$, even when they may not be slack at the optimal point, $g_i(\mathbf{w}^*) = 0$. In practice, this means that initialising edge weights in the interior of the feasible region means that dual variables may have decayed to very small quantities in the transient period required for edge weights to reach the boundary of the feasible set. Thus, it may take an equally long period of time for the dual variables to grow back in magnitude to appropriate levels to satisfy the stationarity condition: $\nabla f(\mathbf{w}^*) + \frac{1}{2}\sum_j \nabla g_j(\mathbf{w}^*)\nu_j^{*2} = \mathbf{0}$.

One solution is to add an extra growth factor to the dual variables when the primal feasibility condition $g_i(\mathbf{w}) \le 0$ is breached. This can be achieved through the use of the ramp function ramp(x) defined as:

$$\text{ramp}(x) = \begin{cases} x & \text{if}\, x > 0 \\ 0 & \text{if}\, x \le 0 \end{cases} \tag{4.70}$$

Modifying the original KKT satisfaction weight adaptation rule ((4.30,a) and (4.30,b)) we have:

$$\dot{w}_i = -\frac{\partial f(\mathbf{w})}{\partial w_i} - \frac{1}{2}\sum_j \frac{\partial g_j(\mathbf{w})}{\partial w_i}\nu_j^2 \tag{4.70,a}$$

$$\dot{\nu}_j = g_j(\mathbf{w})\nu_j + \text{ramp}\left(g_j(\mathbf{w})\right), \quad \forall i \in 1,\dots,m,\ j \in 1,\dots,m+n \tag{4.70,b}$$

Now, when the feasibility constraint is breached $g_j(\mathbf{w}) > 0$, the variables ν_j grow with a rate independent from the current value of ν_j (which may be small for the reasons discussed previously). This alteration also means that the variables ν_j may be initialised with a value of zero, $\nu_j(0) = 0$, and thus the dual variables initially will cause no effect for edges initialised in the interior of the feasible set, and until the adaptation of edge weights cause the boundary of the feasible set to be reached.

Example 4.5. *To illustrate the effect of the addition of the ramp function to the dynamics of the dual variables, we again run the same simulation as described in Example 4.4, with the same* *control parameters and time step. Now however, edge weight and dual variable dynamics are described by the following equations:*

$$\dot{w}_{\{i,j\}} = 2(\zeta - \lambda_2(\mathbf{w}))\frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}} + \frac{1}{2}\left(\mu^2_{\{i,j\}} + \nu_i^2 + \nu_j^2\right) \tag{4.71}$$

$$\dot{\mu}_{\{i,j\}} = -w_{\{i,j\}}\mu_{\{i,j\}} + \text{ramp}\left(-w_{\{i,j\}}\right), \qquad \mu_{\{i,j\}}(0) = 0 \tag{4.72}$$

$$\dot{\nu}_i = \left(\sum_{j\in\mathcal{N}_i} w_{i,j} - \kappa_i\right)\nu_i + \text{ramp}\left(\sum_{j\in\mathcal{N}_i} w_{i,j} - \kappa_i\right), \qquad \nu_i(0) = 0 \tag{4.73}$$

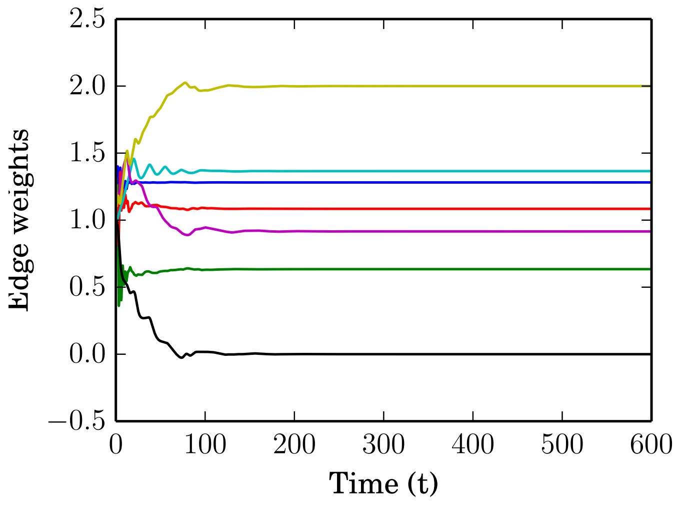


Figure 4.13: With the addition of the ramp term, the transient response of the edge weights is significantly improved. High frequency oscillations do remain, but these oscillations are quickly damped out, and the optimal edge weights are approached very rapidly compared both to the original KKT method, and the previously described logaritmic barrier method. All control parameters remain the same as in Example 4.4, but now $\nu_i(0) = 0$, and $\mu_{\{i,j\}}(0) = 0$.

The addition of the ramp term significantly improves the transient performance of the edge weights, reducing high frequency oscillation amplitude, and making these undesirable oscillations decay away faster. Furthermore, the edge weights converge much more rapidly on the optimal values, when all other control parameters remain constant. It can be seen that the addition of the ramp term maintains the local exponential stability of the equilibrium. Firstly, it can be seen that position of the equilibrium is not changed, the complementary slackness condition still governs the stationarity of the variables ν_j. For $\dot{\nu}_j = 0$, either $g_j(\mathbf{w}^*) = 0$ or $\nu_j = 0$ and $g_j(\mathbf{w}^*) \le 0$. Secondly, we proved in Theorem 4.4 that the equilibrium point of the original system is locally exponentially stable. The effect of the ramp term is to act like a one sided spring, pushing the dual variables back in a direction that ensures that $g_j(\mathbf{w}) \le 0$; this acts as another avenue for losing potential energy, and hence the oscillations are damped.

4.5 Summary

In this Chapter, two edge weight adaptation methods were proposed so that a network may self-organise its edge weights in order to maximise its algebraic connectivity: the second smallest eigenvalue of the graph Laplacian. As we saw in Chapter 2, this eigenvalue can serve as a measure of consensus speed in a network, and also as a measure of how strongly connected a network is: the algebraic connectivity provides a lower bound on the minimum cut of a weighted graph.

The first method proposed utilised logarithmic barriers to enforce the constraints on the edge weights. These barriers were then adapted as the local minimum of the potential well was reached, so that this minimum converged upon the solution to the optimisation problem. Edge weights followed a second-order gradient descent algorithm in this potential well. The nature of this method meant that if a feasible set of edge weights was chosen at initialisation, the trajectories of edge weights would remain feasible for all time, barring any discontinuous change in the feasible region. However, the method is not well-defined outside of the feasible region. Thus, if any perturbation were to move the edge weights into an infeasible regime, the method would fail.

This problem was addressed in the second method: a first order system designed so that the sole stable stationary point of the system satisfied the KKT conditions. Unlike the first method, this system is well defined outside of the feasible region, and does not suffer from the stiffening caused by monotonically increasing severity of barrier functions. However, with this method, there is no guarantee that edge weights will be feasible in the transient to the solution. As such, both methods can be used to complement each other, depending upon the specific requirements of an application.

The stability of each of the methods, assuming knowledge of the objective function and its partial derivatives, was proven in this Chapter. However, for fully decentralised optimisation, both methods, themselves, require decentralised estimates of the objective function (the algebraic connectivity) and its partial derivatives with respect to edge weights. For this, the decentralised algebraic connecitivity estimator presented in [29], and described in Chapter 3, was utilised. It is required that this estimator converge faster than the weight adaptation layer so that the estimates made can be treated as sufficiently accurate. The algebraic connectivity estimator itself relies on two average consensus systems, which must converge sufficiently faster than the algebraic connectivity estimator. Thus, for fully decentralised optimisation of the network's algebraic connectivity, a three-timescale multi-layer networked system is proposed. Each of the layers, themselves, is proven to be stable in isolation given true values for the estimates made in faster subsystems. However, stability of the joint system remains without rigorous proof. Of particular interest is the question of what separation in time-scale is sufficient for the stability of the joint decentralised optimisation system.

In the next Chapter, we look closely at this problem of separation of time-scales between the estimation and optimisation layers. First, we extend the algebraic connectivity estimator [29], so that the spectral radius of the network may also be estimated, allowing the solution of further interesting and useful network optimisation problems: including improving synchronisability, and increasing robustness to time-delay in linear consensus. We then proceed to provide a rigorous proof for the existence of a sufficient time-scale separation, under some moderate assumptions, using techniques from singular perturbation theory.

CHAPTER 5

Decentralised Estimation and Control of the Spectral Radius

Previously we have looked at decentralised estimation and control of the algebraic connectivity λ_2, the smallest non-trivial eigenvalue of the graph Laplacian, using the method presented in [29]. Further to this, the largest eigenvalue of the graph Laplacian, the spectral radius λ_n, is also of critical importance in a number of applications: in the synchronisation of nonlinear oscillators, the synchronisability ratio $\frac{\lambda_n}{\lambda_2}$ must be less than a certain system dependent threshold[1], for local transversal stability of the synchronous solution to be achievable for any coupling strength [12, 13]; in discrete time linear averaging the quantity $\max\{|1-\lambda_2|, |1-\lambda_n|\}$ is the exponent of the rate at which consensus is reached [77]; under simple linear consensus with time-delay, the largest delay that can be tolerated is inversely proportional to λ_n [45]. As such, it is important for a network of locally communicating and cooperating autonomous agents to be able to make local estimates of the spectral radius to best achieve the desired performance of the network.

In this Chapter, we show that the algebraic connectivity estimator presented in [29] can be modified to obtain a decentralised estimator for the spectral radius. We then go on to perform a detailed time scale separation analysis between the estimator system and the PI average consensus subsystems, firstly showing that there exists a sufficient time scale separation for exponential nearly global stability, then continuing the analysis to provide a sufficient condition for local stability using tools from singular perturbation theory. We conclude the Chapter with some simple motivating examples, utilising the weight adaptation laws presented in Chapter 4.

[1]This value is determined by the Master Stability Function, see Chapter 2, Section 2.3.4.

5.1 Estimation of the Spectral Radius

To estimate the spectral radius we propose to use the following reduced system:

(5.1) $$\dot{\mathbf{b}} = -\frac{k_1}{n}\mathbf{1}\mathbf{1}^\top\mathbf{b} + k_2\mathbf{L}\mathbf{b} - k_3\left(\frac{\mathbf{b}^\top\mathbf{b}}{n} - 1\right)\mathbf{b}$$

Here, the vector $\mathbf{b}$ serves as an estimate for the eigenvector associated with the spectral radius. The control variables k_1, k_2, and k_3 each control the strength of a particular term: the first term acts to deflate the consensus mode of the system; the second term forces the vector $\mathbf{b}$ away from the origin, with the mode associated with λ_n dominating; and the final term provides a restorative force when $\mathbf{b}^\top\mathbf{b} > n$ so that $\mathbf{b}$ does not grow without bound under the action of the second term.

Comparing this system to that used in [29] (and described in (3.7), Chapter 3, Section 3.3.1) for estimation of the algebraic connectivity, we can notice a number of similarities. The action of the first term again deflates on the consensus mode, and the third term again acts to normalise the vector by providing a restorative force so that the vector $\mathbf{b}$ remains bounded, whilst forcing the system away from the trivial stationary point at the origin. However, instead of the action from the second term, dependent on $\mathbf{L}$, being used to drive the system towards consensus so that the slowest mode dominates (that associated with λ_2), in (5.1) the second term on the right-hand side provides a diverging force so that the fastest mode (associated with λ_n) dominates. Taking a local view of the system, every node follows the update law,

(5.2) $$\dot{b}_i = -k_1\langle\mathbf{b}\rangle - k_2\sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) - k_3(\langle\mathbf{b}^{\circ 2}\rangle - 1)b_i$$

using the means $\langle\mathbf{b}\rangle = \frac{1}{n}\sum_i b_i$ and $\langle\mathbf{b}^{\circ 2}\rangle = \frac{1}{n}\sum_i b_i^2$, which are global functions. As in the algebraic connectivity estimator, to carry out a decentralised estimation these means will need to be estimated using local update strategies, and PI average consensus estimators [100] will be employed to achieve this, as done in Section 3.3.1.1 and [29]. Firstly, however, we will analyse the properties of the ideal system (5.1) (that with perfect estimation of the means $\langle\mathbf{b}\rangle$ and $\langle\mathbf{b}^{\circ 2}\rangle$), before coupling the PI consensus subsystems, and performing a time-scale analysis of the full system. We show that under suitable conditions on the control parameters, $k_1 > k_3 > 0$, $k_2 > 0$, the sole stable equilibrium manifold is that associated with the spectral radius, and that this equilibrium is almost-globally exponentially stable. The location of this equilibrium can be used to infer the spectral radius of the graph Laplacian, and furthermore, under the condition that the spectral radius is distinct so that its partial derivative with respect to the edge weights is well defined, the partial derivatives $\frac{\partial\lambda_n}{\partial w_{i,j}}$ can also be determined.

Finally, we couple the spectral radius estimator to the edge weight adaptation law presented in Chapter 4, and show that under certain assumptions, there exists a sufficient time-scale separation to guarantee convergence between the edge weight adaptation subsystem, the eigenvalue estimation subsystem, and the PI average consensus subsystems. This is accomplished using results from singular perturbation theory [117, 118, 119].

5.2 Stability of the Reduced System

5.2.1 Location of the Equilibria

Before analysing the stability of the equilibria of the reduced system, (5.1), it is first useful to locate and classify the equilibria, and characterise their nature[2]. Taking the orthogonal matrix $\mathbf{V} = [\frac{1}{\sqrt{n}}\mathbf{1}, \mathbf{v_2}, \ldots, \mathbf{v_n}]$ which diagonalises the graph Laplacian $\mathbf{\Lambda} = \mathbf{V}^\top \mathbf{L} \mathbf{V}$ as a change of basis, the system may be easily diagonalised using the transformation $\mathbf{b}_d = \mathbf{V}^\top \mathbf{b}$, yielding:

$$\dot{\mathbf{b}}_d = \left(-k_1 \begin{bmatrix} 1 & \mathbf{0}^\top \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + k_2 \mathbf{\Lambda} + k_3 \mathbf{I} \right) \mathbf{b}_d - \frac{k_3}{n} \mathbf{b}_d \mathbf{b}_d^\top \mathbf{b}_d \tag{5.3}$$

Each mode can be taken in turn, so that for the consensus mode $b_{d,1}$, we have:

$$\dot{b}_{d,1} = \left(k_3 - k_1 - \frac{k_3}{n} \mathbf{b}_d^\top \mathbf{b}_d \right) b_{d,1} \tag{5.4}$$

which has stationary points at $b^*_{d,1} = 0$ or when $\mathbf{b}^{*\top}_d \mathbf{b}^*_d = \frac{n(k_3 - k_1)}{k_3}$. Clearly, if $k_1 > k_3$, then this second equilibrium point does not exist, and the first equilibrium point is globally exponentially stable; the consensus mode will deflate.

Other modes follow equations of the form:

$$\dot{b}_{d,i} = \left(k_3 + k_2 \lambda_i - \frac{k_3}{n} \mathbf{b}_d^\top \mathbf{b}_d \right) b_{d,i} \tag{5.5}$$

for $i = 2, \ldots, n$

At a stationary point, for each of the non-consensus modes, either $b^*_{d,i} = 0$, or $\mathbf{b}^{*\top}_d \mathbf{b}^*_d = \frac{n(k_3 + k_2 \lambda_i)}{k_3}$. Thus, there is one equilibrium point located at the origin $\mathbf{b}^*_d = \mathbf{0}$, and for each unique eigenvalue λ_i there is also an associated equilibrium space such that $\mathbf{b}^{*\top}_d \mathbf{b}^*_d = \frac{n(k_3 + k_2 \lambda_i)}{k_3}$, with zero elements $b_{d,j} = 0$ for all $\lambda_j \neq \lambda_i$.

In the special case that λ_i has unitary multiplicity (λ_i is a distinct eigenvalue), the associated equilibrium for this mode is a set of two points equidistant from the origin (a 0-sphere) given by:

$$b^*_{d,i} = \pm \sqrt{\frac{n(k_3 + k_2 \lambda_i)}{k_3}} \tag{5.6}$$

$$b^*_{d,j} = 0, \ \forall j \neq i \tag{5.7}$$

In the general case that the eigenvalue is not distinct $\lambda_i = \cdots = \lambda_{i+p}$, having algebraic multiplicity $p + 1$, then the associated equilibrium set of the corresponding mode is a p-sphere of radius $\sqrt{\frac{n(k_3 + k_2 \lambda_i)}{k_3}}$ defined by the set of solutions:

$$\sum_{k \in \{i, \ldots, i+p\}} b^{*2}_{d,k} = \frac{n(k_3 + k_2 \lambda_i)}{k_3} \tag{5.8}$$

$$b^*_{d,j} = 0, \ \forall j \notin \{i, \ldots, i+p\} \tag{5.9}$$

[2]It will become apparent that not all equilibrium sets need be points, or sets of points, but may be continua in higher dimensional manifolds.

For example, for an eigenvalue of algebraic multiplicity two, then the associated equilibrium set is simply a circle in the space spanned by its eigenvectors. When the algebraic multiplicity of an eigenvalue is three, then the equilibrium manifold is the surface of sphere, and so on.

5.2.2 Stability of the Equilibria

Now that the equilibrium manifolds of the ideal spectral radius estimator have been located, we can ascertain the local stability of each through linearisation. Importantly, it can be shown that under the condition that the control parameters satisfy $k_1 > k_3 > 0$, $k_2 > 0$, the only stable equilibrium is that associated with the spectral radius of the graph Laplacian. To deduce global stability results from the linearisation results, we use the fact the estimator is a gradient system, as stated in the following Lemma:

Lemma 5.1. *The system defined in Equation* (5.1) *is bounded, contains no periodic orbits, and there exists at least one stable equilibrium.*

Proof. We notice that the diagonalised system, Equation (5.3), is a real analytic gradient system $\dot{\mathbf{b}}_d = -\nabla V(\mathbf{b}_d)$, where

$$V(\mathbf{b}_d) = \frac{1}{2}k_1 b_{d,1}^2 - \frac{1}{2}k_2 \sum_{i=2}^{n} \lambda_i b_{d,i}^2 - \frac{1}{2}k_3 \sum_{i=1}^{n} b_{d,i}^2 + \frac{k_3}{4n}\sum_{i=1}^{n}\sum_{j=1}^{n} b_{d,i}^2 b_{d,j}^2 \tag{5.10}$$

As such, we can be sure that the system contains no periodic orbits, see [120] for further details. Further to this, we can immediately see that the system is bounded, as for large $\mathbf{b}_d$ the positive quartic terms will dominate in (5.10) (assuming that $k_3 > 0$). The Lyapunov-like potential function $V(\mathbf{b}_d)$ is quartic, thus, by continuity, there must exist a global minimum of the function, and this critical point will be a stable equilibrium. □

Now that it has been shown that at least one stable equilibrium set exists, we can use linearisation to characterise the stability of each equilibrium set, and show that under appropriate constraints on the control parameters, only the equilibrium set associated with the spectral radius is stable. Thus, any trajectory not starting at an equilibrium must converge onto this set, and moreover, convergence will be locally exponentially fast.

Theorem 5.1. *If λ_n is a distinct eigenvalue, and the control parameters satisfy $k_1 > k_3 > 0$, $k_2 > 0$, the equilibrium set associated with λ_n, which consists of two distinct points,*

$$\mathbf{b}^* = \pm\mathbf{v_n}\sqrt{\frac{n(k_3 + k_2\lambda_n)}{k_3}} \tag{5.11}$$

is the sole stable equilibrium set. Moreover, any trajectory that does not originate at an equilibrium will locally exponentially tend to one of these two points with a constant rate of convergence given by:

$$c_2 = -k_2(\lambda_n - \lambda_{n-1}) \tag{5.12}$$

Moreover, if λ_n is not distinct, $\lambda_{n-p} = \cdots = \lambda_n$, having algebraic multiplicity $p+1$, then the sole stable equilibrium manifold is given by the p-sphere:

$$\mathbf{b}^* = \sum_{i=n-p}^{n} \alpha_i \mathbf{v_i}, \qquad \sum_{i=n-p}^{n} \alpha_i^2 = \frac{n(k_3 + k_2\lambda_n)}{k_3} \tag{5.13}$$

Proof. Taking a small perturbation $\widetilde{\mathbf{b}}_d = \mathbf{b}_d^* - \mathbf{b}_d$, the system defined in Equation (5.3) can be linearised about the equilibrium $\mathbf{b}_d^*$, yielding:

$$\dot{\widetilde{\mathbf{b}}}_d \approx \left(-k_1 \begin{bmatrix} 1 & \mathbf{0}^\top \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + k_2\mathbf{\Lambda} + k_3\mathbf{I} - \frac{k_3}{n}\left(2\mathbf{b}_d^*\mathbf{b}_d^{*\top} + (\mathbf{b}_d^{*\top}\mathbf{b}_d^*)\mathbf{I}\right)\right)\widetilde{\mathbf{b}}_d \tag{5.14}$$

In the simplest case where all eigenvalues of the graph Laplacian are distinct, and thus the equilibria associated with each eigenvalue is a set of two distinct points (a 0-sphere), the equilibrium $\mathbf{b}_{d,i}$ associated to each mode of the linearised equation (5.14) will have one nonzero element only at the i^{th} position. Thus, the only non-diagonal term in Equation (5.14), $2\mathbf{b}_d^*\mathbf{b}_d^{*\top}$, will simplify to a diagonal matrix with the value $2b_{d,i}^{*2}$ located at the (i,i) element, and zeros elsewhere. Therefore, the linearised system is diagonal, and eigenvalues may be simply read off, noting that $\mathbf{b}_d^{*\top}\mathbf{b}_d^* = b_{d,i}^{*2} = \frac{n(k_3+k_2\lambda_i)}{k_3}$.

At the 1st equilibrium point (at the origin, $\mathbf{b}_d = \mathbf{0}$), the eigenvalues are:

$$\{-k_1 + k_3, k_3 + k_2\lambda_2, \ldots, k_3 + k_2\lambda_n\} \tag{5.15}$$

The eigenvalues at the $i^{\mathbf{th}}$ equilibrium ($i \in \{2, \ldots, n\}$) are:

$$\left\{-k_1 - k_2\lambda_i, k_2(\lambda_j - \lambda_i)\; \forall j \neq i, -2(k_3 + k_2\lambda_i)\right\} \tag{5.16}$$

It can be seen that if $k_1 > k_3$, with $k_1, k_2, k_3 > 0$, then all eigenvalues are negative only at the n^{th} equilibrium. Again, we stress that this corresponds to an equilibrium set of two distinct points under the condition that λ_n is distinct, and thus each of these two points will be an exponentially stable equilibrium point. For all other equilibria, there is at least one positive eigenvalue and the points will be unstable. Therefore, in the case that all eigenvalues are distinct, almost any trajectory will converge exponentially onto one of the two points in the n^{th} equilibrium set.

When eigenvalues are not distinct, the analysis becomes somewhat more complicated. Consider the nontrivial eigenvalue $\lambda_i = \cdots = \lambda_{i+p} \neq 0$ having multiplicity $p+1$, then the matrix $2\mathbf{b}_d^*\mathbf{b}_d^{*\top}$ in the linearised system (5.14), about any point on the p-sphere equilibrium, will not be diagonal. It will however, be block diagonal, with all elements being zero except for the square block between the (i,i) and $(i+p, i+p)$ elements. Therefore, it is still simple to determine the $n-(p+1)$ eigenvalues which are not affected by this block:

$$\left\{-k_1 - k_2\lambda_i, k_2(\lambda_j - \lambda_i)\; \forall j \notin \{i, \ldots, i+p\}\right\} \tag{5.17}$$

Thus, if there is any eigenvalue λ_j larger than λ_i, the associated p-sphere equilibrium will be unstable in at least one direction. By the fact that the system is a bounded gradient system, there must be at least one stable equilibrium, and through elimination we deduce that this is the equilibrium associated with the spectral radius λ_n.

We therefore conclude that for a Laplacian where λ_n has multiplicity $p+1$, i.e. $\lambda_{n-p} = \cdots = \lambda_n$, from any initial condition that is not in the equilibrium set, the trajectory will converge to the p-sphere equilibrium defined by:

$$\sum_{i\in\{n-p,\dots,n\}} b_{d,i}^{*2} = \frac{n(k_3 + k_2\lambda_n)}{k_3} \tag{5.18}$$

$$b_{d,j}^{*} = 0,\ \forall j \notin \{n-p,\dots,n\} \tag{5.19}$$

Reversing the diagonal transformation, we find the stable equilibrium set for the system described by Equation (5.1):

$$\mathbf{b}^* = \sum_{i=n-p}^{n} \alpha_i \mathbf{v_i}, \qquad \sum_{i=n-p}^{n} \alpha_i^2 = \frac{n(k_3 + k_2\lambda_n)}{k_3} \tag{5.20}$$

In the special case where λ_n is distinct, then $\mathbf{b}^*$ simplifies to:

$$\mathbf{b}^* = \pm\mathbf{v_n}\sqrt{\frac{n(k_3 + k_2\lambda_n)}{k_3}} \tag{5.21}$$

□

5.2.3 Use as an estimator

From this stable equilibrium, we see that $\mathbf{b}$ is an estimate for the eigenvector associated with λ_n. In the case that λ_n is distinct (Assumption 5.1 holds) then $\mathbf{b}$ almost surely converges to the associated eigenvector, while when λ_n is not distinct then $\mathbf{b}$ almost surely converges onto a vector in the associated eigenspace[3].

Whether distinct or not, the magnitude of $\mathbf{b}^*$ is dependent on λ_n so by rearranging Equation (5.20) we have:

$$||\mathbf{b}^*||_2^2 = \frac{n(k_3 + k_2\lambda_n)}{k_3} \tag{5.22}$$

$$\lambda_n = \frac{k_3}{k_2}\left(\frac{||\mathbf{b}^*||_2^2}{n} - 1\right) \tag{5.23}$$

and an estimate of λ_n can be obtained as:

$$\widehat{\lambda_n} = \frac{k_3}{k_2}\left(\frac{||\mathbf{b}||_2^2}{n} - 1\right) \tag{5.24}$$

[3]Under the assumption that the probability distribution on the set of initial conditions contains no delta functions. For example, if the probability distribution on the initial condition were such that the intial condition $\mathbf{b}_0 = \mathbf{0}$ were selected with some non-infinitesimal probability, then the estimator would then not converge almost surely onto the stable equilibrium set, as $\mathbf{0}$ (though unstable) is a stationary point

Estimates of the partial derivatives with repect to the edge weights $\frac{\partial \lambda_n}{\partial w_{\{i,j\}}}$ may also be obtained using this estimator. Taking the eigenvalue equation for eigenvalue $\lambda(\mathbf{w})$, with associated unit eigenvector $\hat{\mathbf{v}}(\mathbf{w})$, and taking the derivative with respect to the edge weight yields:

$$\lambda \hat{\mathbf{v}}(\mathbf{w}) = \mathbf{L}(\mathbf{w})\hat{\mathbf{v}}(\mathbf{w}) \tag{5.25}$$

$$\lambda = \hat{\mathbf{v}}^\top \mathbf{L}\hat{\mathbf{v}} \tag{5.26}$$

$$\frac{\partial \lambda}{\partial w_{\{i,j\}}} = \frac{\partial \hat{\mathbf{v}}^\top}{\partial w_{\{i,j\}}}\mathbf{L}\hat{\mathbf{v}} + \hat{\mathbf{v}}^\top \frac{\partial \mathbf{L}}{\partial w_{\{i,j\}}}\hat{\mathbf{v}} + \hat{\mathbf{v}}^\top \mathbf{L}\frac{\partial \hat{\mathbf{v}}}{\partial w_{\{i,j\}}} \tag{5.27}$$

As $\mathbf{L}$ is symmetric, this ensures that, for the unit eigenvector $\hat{\mathbf{v}}$:

$$\frac{\partial \hat{\mathbf{v}}^\top}{\partial w_{\{i,j\}}}\mathbf{L}\hat{\mathbf{v}} + \hat{\mathbf{v}}^\top \mathbf{L}\frac{\partial \hat{\mathbf{v}}}{\partial w_{\{i,j\}}} = \lambda \frac{\partial (\hat{\mathbf{v}}^\top \hat{\mathbf{v}})}{\partial w_{\{i,j\}}} = 0 \tag{5.28}$$

Therefore,

$$\frac{\partial \lambda}{\partial w_{\{i,j\}}} = \hat{\mathbf{v}}^\top \frac{\partial \mathbf{L}}{\partial w_{\{i,j\}}}\hat{\mathbf{v}} = (\hat{v}_i - \hat{v}_j)^2 \tag{5.29}$$

Using Equation (5.29) and $\frac{\mathbf{b}}{||\mathbf{b}||_2}$ as an estimate of the unit eigenvector $\mathbf{v_n}$, an estimate of the partial derivative can be obtained as,

$$\widehat{\frac{\partial \lambda_n}{\partial w_{i,j}}} = \frac{(b_i - b_j)^2}{||\mathbf{b}||_2^2} \tag{5.30}$$

extending to λ_n the approach to estimate algebraic connectivity sensitivities described in Section 3.3.1.2 and presented in [29].

5.3 Decentralised Spectral Radius Estimator

As for the algebraic connectivity estimator, the spectral radius estimator (5.1, 5.24) requires access to the following global functions: the mean of the components of the estimator vector $\langle \mathbf{b} \rangle$, and the mean of the components squared $\langle \mathbf{b}^{\circ 2} \rangle$. Again, a solution to fully decentralise the estimator is to use local estimates for these means as provided by PI average consensus, as done in Chapter 3 and [100].

In [100] and presented previously in Chapter 3, Section 3.3.1.1, it is shown that the arithmetic mean of a vector, say $\mathbf{u}$, can be estimated in a distributed way on a network, using the linear system:

$$\begin{aligned} \dot{\pi}_i^{\mathbf{u}} &= k_\gamma(u_i - \pi_i^{\mathbf{u}}) + k_P \sum_{j \in \mathcal{N}_i} w_{\{i,j\}}(\pi_j^{\mathbf{u}} - \pi_i^{\mathbf{u}}) + k_I \sum_{j \in \mathcal{N}_i} w_{\{i,j\}}(\zeta_j^{\mathbf{u}} - \zeta_i^{\mathbf{u}}) \\ \dot{\zeta}_i^{\mathbf{u}} &= -k_I \sum_{j \in \mathcal{N}_i} w_{\{i,j\}}(\pi_j^{\mathbf{u}} - \pi_i^{\mathbf{u}}) \end{aligned} \tag{5.31}$$

Here, we use the notation $\pi_i^{\mathbf{u}}$ to designate the i^{th} node's proportional variable for estimating the mean of the vector $\mathbf{u}$, with $\zeta_i^{\mathbf{u}}$ being the corresponding integrator variable.

Note 5.1. *This notation is introduced as we will need to use four PI average consensus systems for estimating the means:*

$$\langle \mathbf{a} \rangle = \frac{\mathbf{1}^\top \mathbf{a}}{n}, \quad \langle \mathbf{a}^{\circ 2} \rangle = \frac{\mathbf{a}^\top \mathbf{a}}{n}, \quad \langle \mathbf{b} \rangle = \frac{\mathbf{1}^\top \mathbf{b}}{n}, \quad \langle \mathbf{b}^{\circ 2} \rangle = \frac{\mathbf{b}^\top \mathbf{b}}{n} \tag{5.32}$$

As such, the superscripts for the four PI consensus systems will be: $\mathbf{a}, \mathbf{a}^{\circ 2}$*,* $\mathbf{b}$*, and* $\mathbf{b}^{\circ 2}$*, according to each system. However, for the moment, we will recall some results about the generic PI system, given in* (5.31).

The collection of all agents proportional and integrator variable dynamics can be concisely written as the matrix differential equation:

$$\begin{bmatrix} \dot{\boldsymbol{\pi}}^{\mathbf{u}} \\ \dot{\boldsymbol{\zeta}}^{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} -k_P \mathbf{L}(\mathbf{w}) - k_\gamma \mathbf{I} & -k_I \mathbf{L}(\mathbf{w}) \\ k_I \mathbf{L}(\mathbf{w}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\pi}^{\mathbf{u}} \\ \boldsymbol{\zeta}^{\mathbf{u}} \end{bmatrix} + k_\gamma \begin{bmatrix} \mathbf{u} \\ \mathbf{0} \end{bmatrix} \tag{5.33}$$

It can be shown that the stable equilibrium manifolds (provided $k_\gamma, k_P, k_I > 0$) for these systems are:

$$\boldsymbol{\pi}^{\mathbf{u},*} = \frac{\mathbf{1}\mathbf{1}^\top \mathbf{u}}{n} \tag{5.34}$$

$$\boldsymbol{\zeta}^{\mathbf{u},*} = -\frac{k_\gamma}{k_I} \mathbf{L}^\dagger(\mathbf{w}) \mathbf{u} + \frac{\mathbf{1}\mathbf{1}^\top \boldsymbol{\zeta}^{\mathbf{u}}(0)}{n} \tag{5.35}$$

where $\mathbf{L}^\dagger(\mathbf{w})$ is the Moore-Penrose inverse of the graph Laplacian:

$$\mathbf{L}^\dagger(\mathbf{w}) \triangleq \left(\mathbf{L}(\mathbf{w}) - \frac{\mathbf{1}\mathbf{1}^\top}{n} \right)^{-1} - \frac{\mathbf{1}\mathbf{1}^\top}{n} \tag{5.36}$$

Considering as input $\mathbf{u}$ and as output $\boldsymbol{\pi}^{\mathbf{u}}$, there is an uncontrollable and unobservable mode in the integrator variables for each PI average consensus estimator associated with the consensus mode, $\mathbf{1}^\top \boldsymbol{\zeta}^{\mathbf{u}}(0)$, which does not have any effect on any other variables. Deflating the systems in this mode through simultaneous diagonalisation, and removal of the zero rows/columns, then results in a distinct equilibrium point. Let $\mathbf{V}(\mathbf{w})$ be the matrix of right eigenvectors of $\mathbf{L}(\mathbf{w})$ so that $\mathbf{L} = \mathbf{V}^\top(\mathbf{w}) \boldsymbol{\Lambda}(\mathbf{w}) \mathbf{V}(\mathbf{w})$. Because we know that the unit eigenvector associated with the consensus mode is simply $\mathbf{v_1} = \frac{\mathbf{1}}{\sqrt{n}}$, we can define an $n \times (n-1)$ matrix of the unknown unit eigenvectors $\mathbf{W}(\mathbf{w})$:

$$\mathbf{V}(\mathbf{w}) = \begin{bmatrix} \frac{\mathbf{1}}{\sqrt{n}} & \mathbf{W}(\mathbf{w}) \end{bmatrix} \tag{5.37}$$

and consequently perform the diagonalisation and truncation of the uncontollable/unobservable mode:

$$\boldsymbol{\pi}_d^{\mathbf{u}} = \mathbf{V}^\top(\mathbf{w}) \boldsymbol{\pi}^{\mathbf{u}}, \; \boldsymbol{\zeta}_d^{\mathbf{u}} = \mathbf{W}^\top(\mathbf{w}) \boldsymbol{\zeta}^{\mathbf{u}} \tag{5.38}$$

so that,

$$\begin{bmatrix} \dot{\boldsymbol{\pi}}_d^{\mathbf{u}} \\ \dot{\boldsymbol{\zeta}}_d^{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} -k_P \boldsymbol{\Lambda}(\mathbf{w}) - k_\gamma \mathbf{I} & k_I \boldsymbol{\Lambda}_r^\top(\mathbf{w}) \\ -k_I \boldsymbol{\Lambda}_r(\mathbf{w}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\pi}_d^{\mathbf{u}} \\ \boldsymbol{\zeta}_d^{\mathbf{u}} \end{bmatrix} + k_\gamma \begin{bmatrix} \mathbf{V}^\top(\mathbf{w})\mathbf{u} \\ \mathbf{0} \end{bmatrix} \tag{5.39}$$

with $\boldsymbol{\Lambda}_r(\mathbf{w})$ being the truncated matrix, with the first row removed:

$$\boldsymbol{\Lambda}_r(\mathbf{w}) = \begin{bmatrix} 0 & \lambda_2(\mathbf{w}) & 0 & 0 & \dots \\ 0 & 0 & \lambda_3(\mathbf{w}) & 0 & \dots \\ 0 & 0 & 0 & \ddots & \\ \vdots & \vdots & & & \lambda_n(\mathbf{w}) \end{bmatrix} \tag{5.40}$$

By Lemma 4.7, in Chapter 4, the system matrix of Equation (5.39) is Hurwitz. Thus, the distinct, stable equilibrium point in the transformed variables is:

$$\boldsymbol{\pi}_d^{\mathbf{u},*} = \begin{bmatrix} \frac{\mathbf{1}^\top \mathbf{u}}{\sqrt{n}} \\ \mathbf{0} \end{bmatrix} \tag{5.41}$$

$$\boldsymbol{\zeta}_d^{\mathbf{u},*} = -\frac{k_\gamma}{k_I} \boldsymbol{\Lambda}_{c,r}^{-1}(\mathbf{w}) \mathbf{W}^\top(\mathbf{w})\mathbf{u} \tag{5.42}$$

where the square matrix $\boldsymbol{\Lambda}_{c,r}(\mathbf{w})$ is the further truncated matrix (the first row and the first column are removed):

$$\boldsymbol{\Lambda}_{c,r}(\mathbf{w}) = \begin{bmatrix} \lambda_2(\mathbf{w}) & 0 & 0 & \dots \\ 0 & \lambda_3(\mathbf{w}) & 0 & \dots \\ 0 & 0 & \ddots & \\ \vdots & & & \lambda_n(\mathbf{w}) \end{bmatrix} \tag{5.43}$$

Futhermore, it is shown in [100] that this equilibrium is globally exponentially stable uniformly in the input vector $\mathbf{u}$.

Taking the ideal spectral radius estimator, as described in (5.1), and substituting into this, the two PI average consensus subsystems for estimating the functions $\langle \mathbf{b} \rangle$ and $\langle \mathbf{b}^{\circ 2} \rangle$ (5.31), we arrive at the fully decentralised system:

$$\begin{aligned} \dot{\mathbf{b}} &= -k_1 \mathbf{w} + k_2 \mathbf{L}\mathbf{b} + k_3 \mathbf{b} - k_3 \mathbf{b} \circ \mathbf{y} \\ \dot{\mathbf{w}} &= k_\gamma \mathbf{b} - k_\gamma \mathbf{w} - k_P \mathbf{L}\mathbf{w} + k_I \mathbf{L}\mathbf{x} \\ \dot{\mathbf{x}} &= -k_I \mathbf{L}\mathbf{w} \\ \dot{\mathbf{y}} &= k_\gamma \mathbf{b}^{\circ 2} - k_\gamma \mathbf{y} - k_P \mathbf{L}\mathbf{y} + k_I \mathbf{L}\mathbf{z} \\ \dot{\mathbf{z}} &= -k_I \mathbf{L}\mathbf{y} \end{aligned} \tag{5.44}$$

This set of ordinary differential equations describes the dynamics of the entire network, and so we call this set the "macroscopic estimator equation". To illustrate that this system is indeed fully decentralised, we may write the equations that the single node i follows, which we will call the "microscopic estimator equation":

$$\begin{aligned}
\dot{b}_i &= -k_1 w_i + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) - k_3(y_i - 1)b_i \\
\dot{w}_i &= k_\gamma(b_i - w_i) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(w_j - w_i) - k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(x_j - x_i) \\
\dot{x}_i &= k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(w_j - w_i) \\
\dot{y}_i &= k_\gamma(b_i^2 - y_i) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_j - y_i) - k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(z_j - z_i) \\
\dot{z}_i &= k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(y_j - y_i)
\end{aligned} \tag{5.45}$$

Note 5.2. *We are expanding the notation here so that:* $\mathbf{w} \triangleq \boldsymbol{\pi}^{\mathbf{b}}$, *is an estimate of the vector* $\langle\mathbf{b}\rangle\mathbf{1} = \frac{1}{n}\mathbf{1}\mathbf{1}^\top\mathbf{b}$*;* $\mathbf{x} \triangleq \boldsymbol{\zeta}^{\mathbf{b}}$ *are the integrator variables for the first PI average estimator;* $\mathbf{y} \triangleq \boldsymbol{\pi}^{\mathbf{b}^{\circ 2}}$, *is an estimate of the vector* $\langle\mathbf{b}^{\circ 2}\rangle\mathbf{1} = \frac{1}{n}\mathbf{1}\mathbf{b}^\top\mathbf{b}$*; and* $\mathbf{z} \triangleq \boldsymbol{\zeta}^{\mathbf{b}^{\circ 2}}$, *the integrator variables for the second PI average estimator. This change of notation is performed to remove the previously used superscripts, which will aid in the clarity of the following sections. However the reuse of the symbol* $\mathbf{w}$ *should not be confused with the vector of edge weights: dependence of the graph Laplacian matrix on the edge weights is supressed in the macroscopic equation, and* w_i *and* $w_{\{i,j\}}$ *are sufficiently dissimilar in the microscopic equation.*

5.3.1 Sufficient Time Scale Separation via Linearisation

Now that we have substituted estimates for the global functions $\langle\mathbf{b}\rangle$ and $\langle\mathbf{b}^{\circ 2}\rangle$ into the ideal spectral radius estimator, it is pertinent to ask how fast the estimates need to be so that the decentralised spectral radius estimator remains stable. To better phrase this question, we introduce the variable $\epsilon \triangleq \frac{1}{k_P}$ which controlls the relative speed of the spectral radius estimator and the PI average consensus subsystems:

$$\dot{\mathbf{b}} = -k_1\mathbf{w} + k_2\mathbf{L}\mathbf{b} + k_3\mathbf{b} - k_3\mathbf{b}\circ\mathbf{y} \tag{5.46}$$

$$\epsilon\dot{\mathbf{w}} = \frac{k_\gamma}{k_P}(\mathbf{b} - \mathbf{w}) - \mathbf{L}\mathbf{w} + \frac{k_I}{k_P}\mathbf{L}\mathbf{x} \tag{5.47}$$

$$\epsilon\dot{\mathbf{x}} = -\frac{k_I}{k_P}\mathbf{L}\mathbf{w} \tag{5.48}$$

$$\epsilon\dot{\mathbf{y}} = \frac{k_\gamma}{k_P}(\mathbf{b}^{\circ 2} - \mathbf{y}) - \mathbf{L}\mathbf{y} + \frac{k_I}{k_P}\mathbf{L}\mathbf{z} \tag{5.49}$$

$$\epsilon\dot{\mathbf{z}} = -\frac{k_I}{k_P}\mathbf{L}\mathbf{y} \tag{5.50}$$

Clearly if the estimators are infinitely fast ($\epsilon = 0$) then the estimates would converge onto the true averages instantly, and Equation (5.46) reduces to the ideal spectral radius estimator, Equation (5.1). As usual in singular perturbation theory [117], this is referred to as the *reduced system*, or the *exact slow model*.

Now, we know that the PI average consensus systems are exponentially stable on their equilibrium points (Equation (5.34) and Equation (5.35)) uniformly in their input **b**, the reduced system has an exponentially stable equilibrium point, the right-hand sides of Equations (5.46)-(5.50) are differentiable in **b**, **w**, **x**, **y**, **z**, and the equilibrium points are isolated and differentiable under the further assumption that the graph Laplacian has distinct eigenvalues. Then, due to the theorem of Tikhonov (Chapter 7, Corollary 2.3 in [117]), there exists a sufficiently small $\epsilon^* > 0$, such that for all $\epsilon \le \epsilon^*$, the equilibrium point of the reduced system is an exponentially stable equilibrium of the slow system.

We can employ linearisation analysis to find a sufficiently small ϵ in the vicinity of the equilibrium point. Firstly we perform a substitution to shift the equilibrium point of the PI average consensus estimators to the origin. The equilibrium points of the PI average consensus subsystems (5.46-5.50) are thus:

$$\begin{aligned}
\mathbf{w}^*(\mathbf{b}) &= \frac{\mathbf{1}\mathbf{1}^\top\mathbf{b}}{n}\\
\mathbf{x}^*(\mathbf{b}) &= -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger\mathbf{b}\\
\mathbf{y}^*(\mathbf{b}) &= \frac{\mathbf{1}\mathbf{b}^\top\mathbf{b}}{n}\\
\mathbf{z}^*(\mathbf{b}) &= -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger(\mathbf{b}\circ\mathbf{b})
\end{aligned} \tag{5.51}$$

and the equilibrium point of the reduced system is given by,

$$\mathbf{b}^* = \pm\mathbf{v}_n\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}} \tag{5.52}$$

so that,

$$\begin{aligned}
\mathbf{w}^*(\mathbf{b}^*) &= \mathbf{0}\\
\mathbf{x}^*(\mathbf{b}^*) &= -\frac{k_\gamma}{k_I\lambda_n}\mathbf{v}_n\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\\
\mathbf{y}^*(\mathbf{b}^*) &= \mathbf{1}\left(\frac{k_3+k_2\lambda_n}{k_3}\right)\\
\mathbf{z}^*(\mathbf{b}^*) &= -\frac{nk_\gamma(k_3+k_2\lambda_n)}{k_3}\mathbf{L}^\dagger(\mathbf{v}_n\circ\mathbf{v}_n)
\end{aligned} \tag{5.53}$$

Note 5.3. *We assume here that the initial condition of the integrator variables are zero,* $\mathbf{x}(0) = \mathbf{0}$*,* $\mathbf{z}(0) = \mathbf{0}$.

Taking small perturbations about these points,

$$\begin{aligned} \delta\mathbf{b} &= \mathbf{b} - \mathbf{b}^* \\ \delta\mathbf{w} &= \mathbf{w} - \mathbf{w}^* \\ \delta\mathbf{x} &= \mathbf{x} - \mathbf{x}^* \\ \delta\mathbf{y} &= \mathbf{y} - \mathbf{y}^* \\ \delta\mathbf{z} &= \mathbf{z} - \mathbf{z}^* \end{aligned} \tag{5.54}$$

results in the following linearised system:
(5.55)

$$\begin{bmatrix} \delta\dot{\mathbf{b}} \\ \epsilon\delta\dot{\mathbf{w}} \\ \epsilon\delta\dot{\mathbf{x}} \\ \epsilon\delta\dot{\mathbf{y}} \\ \epsilon\delta\dot{\mathbf{z}} \end{bmatrix} \approx \overbrace{\begin{bmatrix} k_2\mathbf{L} - k_2\lambda_n\mathbf{I} & -k_1\mathbf{I} & \mathbf{0} & -k_3\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\operatorname{diag}(\mathbf{v}_n) & \mathbf{0} \\ \frac{k_\gamma}{k_P}\mathbf{I} & -\frac{k_\gamma}{k_P}\mathbf{I} - \mathbf{L} & \frac{k_I}{k_P}\mathbf{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{k_I}{k_P}\mathbf{L} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 2\frac{k_\gamma}{k_P}\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\operatorname{diag}(\mathbf{v}_n) & \mathbf{0} & \mathbf{0} & -\frac{k_\gamma}{k_P}\mathbf{I} - \mathbf{L} & \frac{k_I}{k_P}\mathbf{L} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\frac{k_I}{k_P}\mathbf{L} & \mathbf{0} \end{bmatrix}}^{\mathbf{J}} \begin{bmatrix} \delta\mathbf{b} \\ \delta\mathbf{w} \\ \delta\mathbf{x} \\ \delta\mathbf{y} \\ \delta\mathbf{z} \end{bmatrix}$$

This linear dual time-scale system can be analysed using singular perturbation theory for linear time-invariant systems (Chapter 2, [117]). Specifically, using matrix norms of the blocks in the Jacobian $\mathbf{J}$ of (5.55), a sufficient lower bound can be determined for ϵ^*. We can transform the Jacobian into a more structured form using the transformation matrix which utilises the matrix $\mathbf{V}$ which diagonalises the graph Laplacian matrix: $\mathbf{\Lambda} = \mathbf{V}^\top\mathbf{L}\mathbf{V}$. Specifically, we transform the matrix $\mathbf{J}$ using the block diagonal matrix $\mathbf{Z}$:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{V} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{V} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{V} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{V} \end{bmatrix} \tag{5.56}$$

So that,

$$\mathbf{Z}^\top \mathbf{J}\mathbf{Z} = \left[\begin{array}{c|cccc} k_2\boldsymbol{\Lambda} - k_2\lambda_n\mathbf{I} & -k_1\mathbf{I} & \mathbf{0} & -k_3\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\mathbf{Q} & \mathbf{0} \\ \hline \frac{k_\gamma}{k_P}\mathbf{I} & -\frac{k_\gamma}{k_P}\mathbf{I} - \boldsymbol{\Lambda} & \frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 2\frac{k_\gamma}{k_P}\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\mathbf{Q} & \mathbf{0} & \mathbf{0} & -\frac{k_\gamma}{k_P}\mathbf{I} - \mathbf{L} & \frac{k_I}{k_P}\boldsymbol{\Lambda} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} \end{array}\right]$$

$$= \left[\begin{array}{c|c} \mathbf{A}_{1,1} & \mathbf{A}_{1,2} \\ \hline \mathbf{A}_{2,1} & \mathbf{A}_{2,2} \end{array}\right] \tag{5.57}$$

$$\mathbf{A}_{1,1} = k_2\boldsymbol{\Lambda} - k_2\lambda_n\mathbf{I}$$

$$\mathbf{A}_{1,2} = \begin{bmatrix} -k_1\mathbf{I} & \mathbf{0} & -k_3\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\mathbf{Q} & \mathbf{0} \end{bmatrix}$$

$$\mathbf{A}_{2,1} = \begin{bmatrix} \frac{k_\gamma}{k_P}\mathbf{I} & \mathbf{0} & 2\frac{k_\gamma}{k_P}\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}\mathbf{Q} & \mathbf{0} \end{bmatrix}^\top$$

$$\mathbf{A}_{2,2} = \begin{bmatrix} -\frac{k_\gamma}{k_P}\mathbf{I} - \boldsymbol{\Lambda} & \frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} & \mathbf{0} \\ -\frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\frac{k_\gamma}{k_P}\mathbf{I} - \mathbf{L} & \frac{k_I}{k_P}\boldsymbol{\Lambda} \\ \mathbf{0} & \mathbf{0} & -\frac{k_I}{k_P}\boldsymbol{\Lambda} & \mathbf{0} \end{bmatrix}$$

where $\mathbf{Q} = \mathbf{V}^\top \mathrm{diag}(\mathbf{v}_n)\mathbf{V}$. Using Lemma 2.2 from Chapter 2 of [117], we can find a sufficient bound on ϵ^*:

$$\epsilon^* = \frac{1}{||\mathbf{A}_{2,2}^{-1}||\left(||\mathbf{A}_0|| + ||\mathbf{A}_{1,2}||\,||\mathbf{A}_{2,2}^{-1}\mathbf{A}_{2,1}|| + 2\left(||\mathbf{A}_0||\,||\mathbf{A}_{1,2}||\,||\mathbf{A}_{2,2}^{-1}\mathbf{A}_{2,1}||\right)^{\frac{1}{2}}\right)} \tag{5.58}$$

where,

$$\mathbf{A}_0 = \mathbf{A}_{1,1} - \mathbf{A}_{1,2}\mathbf{A}_{2,2}^{-1}\mathbf{A}_{2,1}$$

$$= \begin{bmatrix} -k_1 - k_2\lambda_n & 0 & \dots & 0 & 0 \\ 0 & k_2(\lambda_2 - \lambda_n) & 0 & 0 & 0 \\ \vdots & 0 & \ddots & 0 & \vdots \\ 0 & 0 & 0 & k_2(\lambda_{n-1} - \lambda_n) & 0 \\ 0 & 0 & \dots & 0 & 2(k_3 + k_2\lambda_n) \end{bmatrix} \tag{5.59}$$

is the linearisation of the reduced system about its stable equilibrium associated with λ_n.

Upper bounds can be found for the matrix norms of the blocks $\mathbf{A}_{1,1}$, $\mathbf{A}_{1,2}$, $\mathbf{A}_{2,1}$, and $\mathbf{A}_{2,2}$ in terms of the control parameters and eigenvalues of the graph Laplacian matrix:

$$
\begin{aligned}
||\mathbf{A}_{2,2}^{-1}||_2 &\leq \max\left[\frac{k_P}{k_\gamma}, \frac{k_P^2}{k_I^2\lambda_2^2}\left(\lambda_2\left(1+\frac{k_I}{k_P}\right)+\frac{k_\gamma}{k_P}\right)\right] \\
||\mathbf{A}_0||_2 &= \max[k_1 + k_2\lambda_n, 2(k_3+k_2\lambda_n)] \\
||\mathbf{A}_{2,1}||_2 &\leq \sqrt{\frac{4k_\gamma^2 n(k_3+k_2\lambda_n)}{k_P^2 k_3} + \frac{k_\gamma^2}{k_P^2}} < \frac{k_\gamma}{k_P}\left(2\sqrt{\frac{n(k_3+k_2\lambda_n)}{k_3}}+1\right) \\
||\mathbf{A}_{1,2}||_2 &\leq \sqrt{k_1^2 + nk_3(k_3+k_2\lambda_n)} < k_1 + \sqrt{nk_3(k_3+k_2\lambda_n)}
\end{aligned} \tag{5.60}
$$

These upper bounds can be used in conjunction with the norm inequality $||\mathbf{AB}|| \leq ||\mathbf{A}||\,||\mathbf{B}||$ to find a lower bound on ϵ^* according to Equation (5.58).

5.4 Combination with Weight Adaptation

We are now ready to combine the algebraic connectivity estimator of [29], and the weight adaptation law presented in Chapter 4, with the spectral radius estimator presented in this Chapter. This allows us to, for instance, minimise the spectral radius of a weighted undirected graph whilst maintaining a minimum algebraic connectivity, ensuring that the network remains sufficiently well connected. This graph eigenvalue optimisation problem can be used for instance to guarantee a specified robustness to time-delay in simple linear consensus with delay. When the simple linear consensus protocol is affected by a homogeneous time delay of τ,

$$\dot{\mathbf{x}}(t) = -\mathbf{L}(\mathbf{w})\mathbf{x}(t-\tau) \tag{5.61}$$

the spectral radius of the graph Laplacian becomes critically important to the stability of the system. In a result from [45], it is shown that for the consensus solution to be stable, an upper bound must hold on the spectral radius λ_n of the graph Laplacian, and consequently that a network with lower spectral radius will be stable under longer time-delays. Specifically, it is found that for consensus to be robust to the time-delay, the spectral radius of the graph Laplacian must satisfy the following inequality:

$$\lambda_n < \frac{\pi}{2\tau} \tag{5.62}$$

The eigenvalue optimisation problem Equations (5.63) and (5.64) also has the effect of minimising the eigenratio $\frac{\lambda_n}{\lambda_2}$, also known as the synchronisability ratio, which has important implications for the ability of coupled oscillators on a network to synchronise [13, 43, 12].

Specifically, we want to find a decentralised and distributed weight adaptation algorithm to solve the following optimisation problem:

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}} && \lambda_n(\mathbf{L}(\mathbf{w})) && (5.63)\\ &\text{subject to} && \lambda_2(\mathbf{L}(\mathbf{w})) - \kappa \geq 0 && (5.64)\\ & && w_{\{i,j\}} \geq 0 \qquad \forall \{i,j\} \in \mathscr{E} \end{aligned}$$

where $\kappa > 0$ is some mimimum required bound on the algebraic connectivity, ensuring that the network remains connected.

According to the second weight adaptation law presented in Chapter 4, this convex optimisation problem can be solved through each edge weight obeying the following dynamics:

$$\dot{w}_{\{i,j\}} = -\frac{\partial(\lambda_n(\mathbf{w}))^2}{\partial w_{\{i,j\}}} + \frac{1}{2}\left(\mu^2_{\{i,j\}} + \frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}}\nu^2_{\{i,j\}}\right) \tag{5.65}$$

$$= -2\lambda_n(\mathbf{w})\frac{\partial \lambda_n(\mathbf{w})}{\partial w_{\{i,j\}}} + \frac{1}{2}\left(\mu^2_{\{i,j\}} + \frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}}\nu^2_{\{i,j\}}\right) \tag{5.66}$$

$$\dot{\mu}_{\{i,j\}} = -w_{\{i,j\}}\mu_{\{i,j\}}, \qquad \mu_{\{i,j\}}(0) > 0 \tag{5.67}$$

$$\dot{\nu}_{\{i,j\}} = (\kappa - \lambda_2(\mathbf{w}))\nu_{\{i,j\}}, \qquad \nu_{\{i,j\}}(0) > 0 \tag{5.68}$$

where the collection of all ODEs for each node and edge may be written in the more compact form:

$$\begin{aligned} \dot{\mathbf{w}} &= -2\lambda_n(\mathbf{w})\frac{\partial \lambda_n(\mathbf{w})}{\partial \mathbf{w}} + \frac{1}{2}\left(\boldsymbol{\mu}^{\circ 2} + \frac{\partial \lambda_2(\mathbf{w})}{\partial \mathbf{w}} \circ \boldsymbol{\nu}^{\circ 2}\right)\\ \dot{\boldsymbol{\mu}} &= -\mathbf{w} \circ \boldsymbol{\mu}\\ \dot{\boldsymbol{\nu}} &= (\kappa - \lambda_2(\mathbf{w}))\boldsymbol{\nu} \end{aligned} \tag{5.69}$$

Note 5.4. *The reason for why we are minimising the square of the spectral radius here is made clear is to enforce strict convexity on the objective function. This is explained in greater detail in Chapter 4, Section 4.4.1.*

The global functions $\lambda_2(\mathbf{w}), \lambda_n(\mathbf{w})$, and their partial derivatives $\frac{\partial \lambda_2}{\partial \mathbf{w}}$ and $\frac{\partial \lambda_n}{\partial \mathbf{w}}$, are estimated in a decentralised fashion using the algebraic connectivity estimator described previously in Chapter 3 [29] and the spectral radius estimator described earlier in this Chapter. These distributed estimators allow the following local estimates to be made at each node i:

$$\widehat{\lambda_2}^{(i)} = \frac{k_3}{k_2}\left(1 - \pi_i^{\mathbf{a}^{\circ 2}}\right) \tag{5.70}$$

$$\widehat{\frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i)} = \frac{(a_i - a_j)^2}{n\pi_i^{\mathbf{a}^{\circ 2}}} \tag{5.71}$$

$$\widehat{\lambda_n}^{(i)} = \frac{k_3}{k_2}\left(\pi_i^{\mathbf{b}^{\circ 2}} - 1\right) \tag{5.72}$$

$$\widehat{\frac{\partial \lambda_n(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i)} = \frac{(b_i - b_j)^2}{n\pi_i^{\mathbf{b}^{\circ 2}}} \tag{5.73}$$

using the earlier notation for the general PI average consensus estimators. Each edge $\{i,j\}$ has access to both the local variables at node i and at node j, thus it is possible to obtain local estimates of these functions. So as not to bias one parent node over the other, we choose to use the mean of the consensus variables from i and j when either would do, as we expect consensus to be reached. Specifically, we have:

$$\widehat{\lambda_2}^{(i,j)} = \frac{k_3}{k_2}\left(1 - \frac{\pi_i^{\mathbf{a}^{\circ 2}} + \pi_j^{\mathbf{a}^{\circ 2}}}{2}\right) \tag{5.74}$$

$$\widehat{\frac{\partial \lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i,j)} = \frac{2(a_i - a_j)^2}{n(\pi_i^{\mathbf{a}^{\circ 2}} + \pi_j^{\mathbf{a}^{\circ 2}})} \tag{5.75}$$

$$\widehat{\lambda_n}^{(i,j)} = \frac{k_3}{k_2}\left(\frac{\pi_i^{\mathbf{b}^{\circ 2}} + \pi_j^{\mathbf{b}^{\circ 2}}}{2} - 1\right) \tag{5.76}$$

$$\widehat{\frac{\partial \lambda_n(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i,j)} = \frac{2(b_i - b_j)^2}{n(\pi_i^{\mathbf{b}^{\circ 2}} + \pi_j^{\mathbf{b}^{\circ 2}})} \tag{5.77}$$

Substituting these local estimates into the weight optimisation layer (Equations (5.66) to (5.68)), we finally accomplish the goal of formulating an entirely distributed adaptive solution to the optimisation problem described in (5.63) and (5.64). Specifically, writing the equations in scalar form, we have:

$$\begin{aligned} \dot{w}_{\{i,j\}} &= -2\widehat{\lambda_n(\mathbf{w})}^{(i,j)} \widehat{\frac{\partial \lambda_n(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i,j)} + \frac{1}{2}\widehat{\frac{\lambda_2(\mathbf{w})}{\partial w_{\{i,j\}}}}^{(i,j)} \left(\frac{\nu_i + \nu_j}{2}\right)^2 + \frac{1}{2}\mu_{\{i,j\}}^2 \\ &= -\frac{4k_3}{nk_2}\left(\frac{\pi_i^{\mathbf{b}^{\circ 2}} + \pi_j^{\mathbf{b}^{\circ 2}}}{2} - 1\right)\frac{(b_i - b_j)^2}{(\pi_i^{\mathbf{b}^{\circ 2}} + \pi_j^{\mathbf{b}^{\circ 2}})} + \frac{(a_i - a_j)^2}{n(\pi_i^{\mathbf{a}^{\circ 2}} + \pi_j^{\mathbf{a}^{\circ 2}})}\left(\frac{\nu_i + \nu_j}{2}\right)^2 + \frac{1}{2}\mu_{\{i,j\}}^2 \end{aligned} \tag{5.78}$$

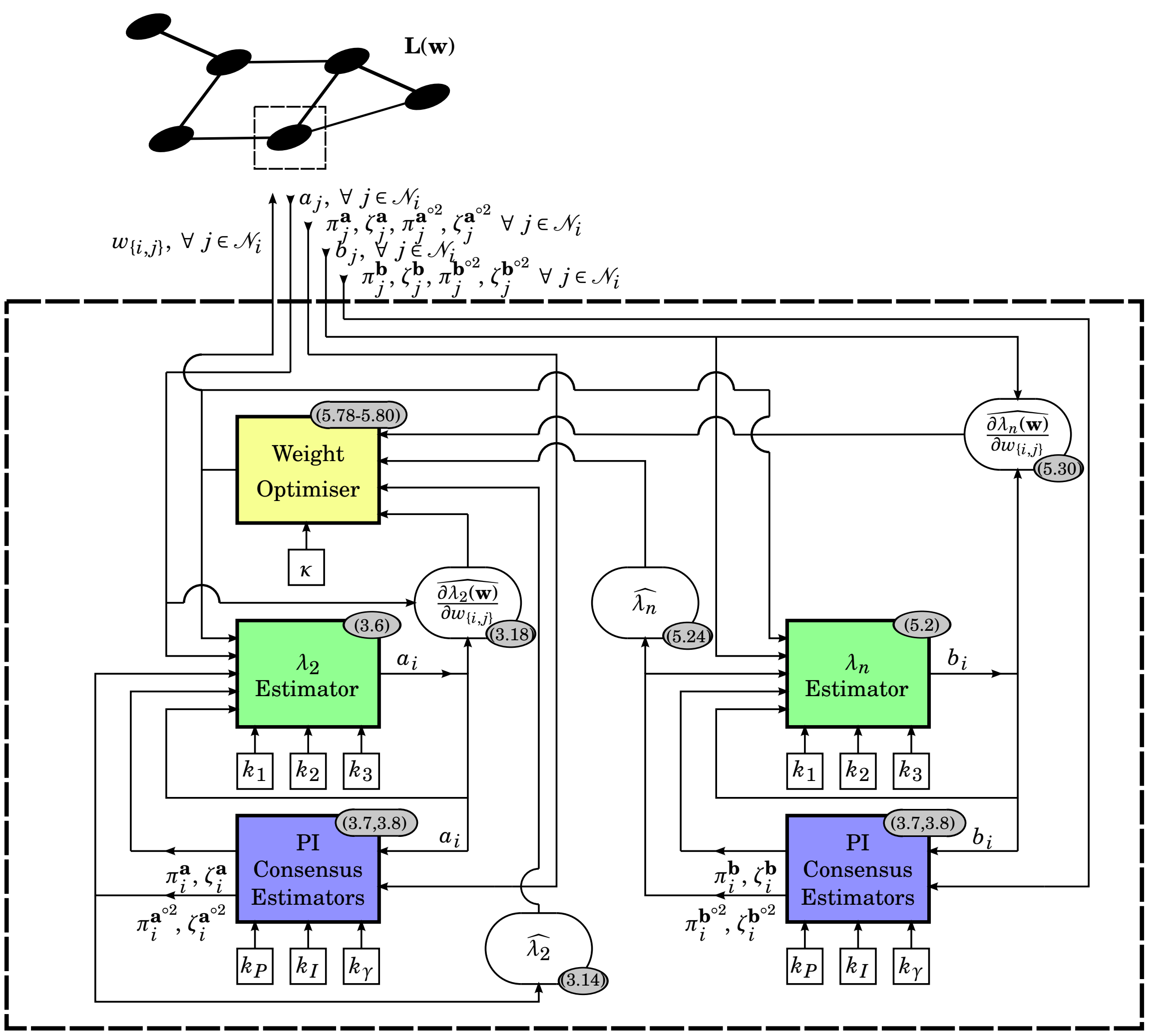
$$\dot{\mu}_{\{i,j\}} = -w_{\{i,j\}}\mu_{\{i,j\}}, \qquad \mu_{\{i,j\}}(0) > 0 \tag{5.79}$$

$$\begin{aligned} \dot{\nu}_i &= \left(\kappa - \widehat{\lambda_2(\mathbf{w})}^{(i)}\right)\nu_i \\ &= \left(\kappa - \frac{k_3}{k_2}\left(1 - \pi_i^{\mathbf{a}^{\circ 2}}\right)\right)\nu_i, \qquad \nu_i(0) > 0 \end{aligned} \tag{5.80}$$

The system is now entirely decentralised[4] and, for clarity, the interactions between the various subsystems within an individual node is graphically represented in Figure 5.1.

Note 5.5. *It can be seen that the parameter n, the number of nodes in the network, appears in Equation* (5.78)*, and is a global variable; an individual agent with only local information may not necessarily know the value of n. However, the variable n only acts to scale the estimates of the partial derivatives, and can therefore be omitted from Equation* (5.78) *without changing the direction of steepest descent, only the magnitude, and as such will not affect the location and*

[4]See Note 5.5, regarding knowledge of the global variable n.



L(w)
$w_{\{i,j\}}, \forall j \in \mathcal{N}_i$
$a_j, \forall j \in \mathcal{N}_i$
$\pi_j^{\mathbf{a}}, \zeta_j^{\mathbf{a}}, \pi_j^{\mathbf{a}^{\circ 2}}, \zeta_j^{\mathbf{a}^{\circ 2}} \forall j \in \mathcal{N}_i$
$b_j, \forall j \in \mathcal{N}_i$
$\pi_j^{\mathbf{b}}, \zeta_j^{\mathbf{b}}, \pi_j^{\mathbf{b}^{\circ 2}}, \zeta_j^{\mathbf{b}^{\circ 2}} \forall j \in \mathcal{N}_i$
(5.78-5.80)
Weight Optimiser
κ
$\frac{\partial \lambda_n(\mathbf{w})}{\partial w_{\{i,j\}}}$
(5.30)
$\frac{\widehat{\partial \lambda_2(\mathbf{w})}}{\partial w_{\{i,j\}}}$
(3.18)
$\widehat{\lambda_n}$
(5.24)
(3.6)
λ_2 Estimator
a_i
k_1 k_2 k_3
(5.2)
λ_n Estimator
b_i
(3.7,3.8)
PI Consensus Estimators
$\pi_i^{\mathbf{a}}, \zeta_i^{\mathbf{a}}$
$\pi_i^{\mathbf{a}^{\circ 2}}, \zeta_i^{\mathbf{a}^{\circ 2}}$
k_P k_I k_γ
$\widehat{\lambda_2}$
(3.14)
$\pi_i^{\mathbf{b}}, \zeta_i^{\mathbf{b}}$
$\pi_i^{\mathbf{b}^{\circ 2}}, \zeta_i^{\mathbf{b}^{\circ 2}}$

Figure 5.1: A simplified flowchart of information flow within a node. Each block is labeled with a number indicating which equation in the thesis the block represents. There are three time scales for the ordinary differential equations indicated by the colours: yellow (slow), green (faster), and blue (fastest). White rounded blocks are simply algebraic equations, and white squares signify control constants.

stability of the global minimum. To see that n can be omitted from the local update rule without changing the result of the optimisation, imagine instead we were performing the optimisation:

$$\begin{aligned}
&(5.81) \qquad && \underset{\mathbf{w}}{\text{minimize}} && n\lambda_n(\mathbf{L}(\mathbf{w})) \\
&(5.82) \qquad && \text{subject to} && n(\lambda_2(\mathbf{L}(\mathbf{w})) - \kappa) \geq 0 \\
& && && w_{\{i,j\}} \geq 0 \qquad \forall \{i,j\} \in \mathscr{E}
\end{aligned}$$

Clearly the same optimal value $\mathbf{w}_{\text{opt}}$ *solves both optimisation problems, but in this modification to the original optisation problem, the parameter n, premultiplying the functions Equation* (5.81) *and Equation* (5.82)*, will cancel with the reciprocal of n from the partial derivatives. In this case,* *the update of the the dual variable ν_i then becomes:*

$$\dot{\nu}_i = n\left(\kappa - \frac{k_3}{k_2}\left(1 - \pi_i^{\mathbf{a}^{\circ 2}}\right)\right)\nu_i, \qquad \nu_i(0) > 0 \tag{5.83}$$

If we omit the constant factor n from this equation, then the only effect is that ν_i will adapt proportionally slower; the stationary points and their stability remain the same. Hence, it is not necessary to know beforehand the value of n for this optimisation procedure to be fully decentralised.

In networks where the number of nodes in the network is unlikely to change over time, it would also be reasonable to run a decentralised algorithm for estimating the number of nodes in the network before the optimisation procedure, though this is not strictly necessary.

5.5 Singular Perturbation Analysis

Now that the estimation and optimisation system that solves the problem (5.63,5.64) has been fully decentralised, we seek to perform a singular perturbation analysis on the multi-layer system, proving that given sufficient separation in time-scale between the layers of estimation and optimisation, the joint system is stable on the solution, provided some assumptions hold.

To clarify the separation in time-scales between the different layers, we normalise the eigenvalue estimators by $\epsilon_1 = \frac{1}{k_2}$, and the PI average consensus layers by $\epsilon_2 = \frac{1}{k_P}$. We are interested in the behaviour of this system as $\epsilon_1 \to 0$ and $\frac{\epsilon_2}{\epsilon_1} \to 0$, when the control parameter ratios $\frac{k_1}{k_2}$, $\frac{k_3}{k_2}$, $\frac{k_\gamma}{k_P}$ and $\frac{k_I}{k_P}$ remain fixed. Again, the complete set of equations can be written in a more compact form, using the oriented and unoriented incidence matrices $\mathbf{P}$ and $\mathbf{M}$ defined previously in Chapter 2:

$$\begin{aligned}\dot{\mathbf{w}} = &-\frac{4k_3}{nk_2}\left(\frac{1}{2}\mathbf{M}\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}} - \mathbf{1}\right) \circ (\mathbf{Pb})^{\circ 2} \circ (\mathbf{M}\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}})^{\circ -1} \\ &+ \frac{1}{4n}(\mathbf{Pa})^{\circ 2}(\mathbf{M}\boldsymbol{\nu})^{\circ 2} \circ (\mathbf{M}\boldsymbol{\pi}^{\mathbf{a}^{\circ 2}})^{\circ -1} + \frac{1}{2}\boldsymbol{\mu}^{\circ 2}\end{aligned} \tag{5.84}$$

$$\dot{\boldsymbol{\mu}} = -\mathbf{w} \circ \boldsymbol{\mu} \tag{5.85}$$

$$\dot{\boldsymbol{\nu}} = \left(\kappa\mathbf{1} - \frac{k_3}{k_2}(\mathbf{1} - \boldsymbol{\pi}^{\mathbf{a}^{\circ 2}})\right) \circ \boldsymbol{\nu} \tag{5.86}$$

$$\epsilon_1\dot{\mathbf{a}} = -\frac{k_1}{k_2}\boldsymbol{\pi}^{\mathbf{a}} - \mathbf{L}(\mathbf{w})\mathbf{a} - \frac{k_3}{k_2}(\boldsymbol{\pi}^{\mathbf{a}^{\circ 2}} - \mathbf{1}) \circ \mathbf{a} \tag{5.87}$$

$$\epsilon_1\dot{\mathbf{b}} = -\frac{k_1}{k_2}\boldsymbol{\pi}^{\mathbf{b}} + \mathbf{L}(\mathbf{w})\mathbf{b} - \frac{k_3}{k_2}(\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}} - \mathbf{1}) \circ \mathbf{b} \tag{5.88}$$

$$\epsilon_2\dot{\boldsymbol{\pi}}^{\mathbf{a}} = \left(-\mathbf{L}(\mathbf{w}) - \frac{k_\gamma}{k_P}\mathbf{I}\right)\boldsymbol{\pi}^{\mathbf{a}} - \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\zeta}^{\mathbf{a}} + \frac{k_\gamma}{k_P}\mathbf{a} \tag{5.89}$$

$$\epsilon_2\dot{\boldsymbol{\zeta}}^{\mathbf{a}} = \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\pi}^{\mathbf{a}} \tag{5.90}$$

$$\epsilon_2\dot{\boldsymbol{\pi}}^{\mathbf{a}^{\circ 2}} = \left(-\mathbf{L}(\mathbf{w}) - \frac{k_\gamma}{k_P}\mathbf{I}\right)\boldsymbol{\pi}^{\mathbf{a}^{\circ 2}} - \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\zeta}^{\mathbf{a}^{\circ 2}} + \frac{k_\gamma}{k_P}\mathbf{a}^{\circ 2} \tag{5.91}$$

$$\epsilon_2\dot{\boldsymbol{\zeta}}^{\mathbf{a}^{\circ 2}} = \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\pi}^{\mathbf{a}^{\circ 2}} \tag{5.92}$$

$$\epsilon_2\dot{\boldsymbol{\pi}}^{\mathbf{b}} = \left(-\mathbf{L}(\mathbf{w}) - \frac{k_\gamma}{k_P}\mathbf{I}\right)\boldsymbol{\pi}^{\mathbf{b}} - \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\zeta}^{\mathbf{b}} + \frac{k_\gamma}{k_P}\mathbf{b} \tag{5.93}$$

$$\epsilon_2\dot{\boldsymbol{\zeta}}^{\mathbf{b}} = \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\pi}^{\mathbf{b}} \tag{5.94}$$

$$\epsilon_2\dot{\boldsymbol{\pi}}^{\mathbf{b}^{\circ 2}} = \left(-\mathbf{L}(\mathbf{w}) - \frac{k_\gamma}{k_P}\mathbf{I}\right)\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}} - \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\zeta}^{\mathbf{b}^{\circ 2}} + \frac{k_\gamma}{k_P}\mathbf{b}^{\circ 2} \tag{5.95}$$

$$\epsilon_2\dot{\boldsymbol{\zeta}}^{\mathbf{b}^{\circ 2}} = \frac{k_I}{k_P}\mathbf{L}(\mathbf{w})\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}} \tag{5.96}$$

The system exhibits a three time-scale structure of the form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \\ \epsilon_1\dot{\mathbf{y_1}} &= \mathbf{g_1}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \\ \epsilon_2\dot{\mathbf{y_2}} &= \mathbf{g_2}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \end{aligned} \tag{5.97}$$

with column vectors $\mathbf{x} \triangleq [\mathbf{w},\boldsymbol{\mu},\boldsymbol{\nu}]$, $\mathbf{y_1} \triangleq [\mathbf{a},\mathbf{b}]$, and $\mathbf{y_2} \triangleq [\boldsymbol{\pi}^{\mathbf{a}},\boldsymbol{\zeta}^{\mathbf{a}},\boldsymbol{\pi}^{\mathbf{a}^{\circ 2}},\boldsymbol{\zeta}^{\mathbf{a}^{\circ 2}},\boldsymbol{\pi}^{\mathbf{b}},\boldsymbol{\zeta}^{\mathbf{b}},\boldsymbol{\pi}^{\mathbf{b}^{\circ 2}},\boldsymbol{\zeta}^{\mathbf{b}^{\circ 2}}]$.

An important condition for the system to work as intended is the following assumption:

Assumption 5.1. *At the optimal edge weights* $\mathbf{w}_{\text{opt}}$ *which solve the optimisation problem stated in* (5.63),(5.64)*, we assume that the eigenvalue* $\lambda_2(\mathbf{w}_{\text{opt}}) \neq \lambda_3(\mathbf{w}_{\text{opt}})$ *and that* $\lambda_n(\mathbf{w}_{\text{opt}}) \neq \lambda_{n-1}(\mathbf{w}_{\text{opt}})$ *implying that the algebraic connectivity* λ_2 *and the spectral radius* λ_n *are distinct. Furthermore, any feasible algebraic connectivity is bounded away from zero when* $\kappa > 0$.

We make this assumption to facilitate the analysis of the subsystems, and argue that this assumption is required for exponential convergence of edge weights onto the solution in the entire distributed system. In Section 5.6.2, we illustrate via an example, what can potentially happen when this assumption does not hold, and demonstrate the performance of the system in this case.

Lemma 5.2. *[121]. Under Assumption 5.1, the algebraic connectivity* $\lambda_2(\mathbf{w})$*, the spectral radius* $\lambda_n(\mathbf{w})$ *and their associated unit eigenvectors,* $\mathbf{v_2}(\mathbf{w})$ *and* $\mathbf{v_n}(\mathbf{w})$*, are analytic functions of the edge weights in the vicinity of the solution,* $\mathbf{w}_{\text{opt}}$*, and hence their partial derivatives are well defined in this region.*

5.5.1 Existence of Sufficient Time Scale Separation

Theorem 1 from Hoppensteadt [119] deals specifically with singularly perturbed systems exhibiting a three time-scale structure, and conditions are given under which the solution of system (5.98) converges to that of the reduced system:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \\ \mathbf{0} &= \mathbf{g_1}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \\ \mathbf{0} &= \mathbf{g_2}(\mathbf{x},\mathbf{y_1},\mathbf{y_2}) \end{aligned} \tag{5.98}$$

as $\epsilon_1 \to 0$ and $\frac{\epsilon_2}{\epsilon_1} \to 0$. This corresponds to the weight adaptation law behaving similarly to when nodes and edges have perfect knowledge of the graph Laplacian eigenvalues and their derivatives. We now proceed to go through the conditions given in [119] and discuss under which assumptions they hold.

5.5.2 Transformation to place the stable equilibrium point at the origin

Condition 1 from [119] requires that (i) there exists an equilibrium point for each of the fast subsystems, (ii) that this equilibrium point is centred on the origin of the faster systems, and also (iii) that this equilibrium point is isolated. That is, it is required that:

$$\mathbf{g_1}(\mathbf{x},\mathbf{0},\mathbf{0}) = \mathbf{0} \tag{5.99}$$

$$\mathbf{g_2}(\mathbf{x},\mathbf{y_1},\mathbf{0}) = \mathbf{0} \tag{5.100}$$

To achieve this condition, a change of variables is firstly required. Observing the equilibrium point of the PI average consensus estimators (Equations (5.34) and (5.35)), we define:

$$\mathbf{Y_2}(\mathbf{x},\mathbf{y_1}) = \begin{bmatrix} \pi^{\mathbf{a},*}(\mathbf{y_1}) \\ \zeta^{\mathbf{a},*}(\mathbf{x},\mathbf{y_1}) \\ \pi^{\mathbf{a}^{\circ 2},*}(\mathbf{y_1}) \\ \zeta^{\mathbf{a}^{\circ 2},*}(\mathbf{x},\mathbf{y_1}) \\ \pi^{\mathbf{b},*}(\mathbf{y_1}) \\ \zeta^{\mathbf{b},*}(\mathbf{x},\mathbf{y_1}) \\ \pi^{\mathbf{b}^{\circ 2},*}(\mathbf{y_1}) \\ \zeta^{\mathbf{b}^{\circ 2},*}(\mathbf{x},\mathbf{y_1}) \end{bmatrix} = \begin{bmatrix} \frac{\mathbf{11}^\top \mathbf{a}}{n} \\ -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger(\mathbf{w})\mathbf{a} + \frac{\mathbf{11}^\top \zeta^{\mathbf{a}}(0)}{n} \\ \frac{\mathbf{1a}^\top \mathbf{a}}{n} \\ -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger(\mathbf{w})\mathbf{a}^{\circ 2} + \frac{\mathbf{11}^\top \zeta^{\mathbf{a}^{\circ 2}}(0)}{n} \\ \frac{\mathbf{11}^\top \mathbf{b}}{n} \\ -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger(\mathbf{w})\mathbf{b} + \frac{\mathbf{11}^\top \zeta^{\mathbf{b}}(0)}{n} \\ \frac{\mathbf{1b}^\top \mathbf{b}}{n} \\ -\frac{k_\gamma}{k_I}\mathbf{L}^\dagger(\mathbf{w})\mathbf{b}^{\circ 2} + \frac{\mathbf{11}^\top \zeta^{\mathbf{b}^{\circ 2}}(0)}{n} \end{bmatrix} \tag{5.101}$$

Lemma 5.3. *The equilibrium point of the PI average consensus estimators,* $\mathbf{Y_2}(\mathbf{x},\mathbf{y_1})$*, is an analytic function of* $\mathbf{x}$ *and* $\mathbf{y_1}$

Proof. It is evident that the equilibria of the proportional variables are analytic, as they are either linear or quadratic functions in $\mathbf{y_1}$. For analyticity of the integrator equilibria, notice that the Moore-Penrose inverse of the graph Laplacian $\mathbf{L}^\dagger(\mathbf{w}) = (\mathbf{L}(\mathbf{w}) - \frac{\mathbf{11}^\top}{n})^{-1} - \frac{\mathbf{11}^\top}{n}$ has elements which are ratios of polynomials in $\mathbf{w}$, with non-zero denominator in the feasible set of $\mathbf{w}$: numerators arise from the cofactors of $\mathbf{L}(\mathbf{w}) - \frac{\mathbf{11}^\top}{n}$, where elements of $\mathbf{L}(\mathbf{w})$ are linear in $\mathbf{w}$, and the denominator is $\det\{\mathbf{L}(\mathbf{w}) - \frac{\mathbf{11}^\top}{n}\} \neq 0$ if $\lambda_2 > 0$. □

Looking at the stable equilibria of the eigenvalue estimators, (3.12) and (5.21), we define the equilibrium of the eigenvalue estimation system as:

$$\mathbf{Y_1}(\mathbf{x}) = \begin{bmatrix} \mathbf{a}^*(\mathbf{x}) \\ \mathbf{b}^*(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} +\mathbf{v_2}(\mathbf{w})\sqrt{\frac{n(k_3 - k_2\lambda_2(\mathbf{w}))}{k_3}} \\ +\mathbf{v_n}(\mathbf{w})\sqrt{\frac{n(k_3 + k_2\lambda_n(\mathbf{w}))}{k_3}} \end{bmatrix} \tag{5.102}$$

which by Lemma 5.2 is a real analytic function in $\mathbf{x}$ in the vicinity of $\mathbf{x} = \mathbf{w}_{\text{opt}}$ under Assumption 5.1, provided that $k_3 \neq 0$ and $k_3 > k_2\lambda_2(\mathbf{w})$. Note that here we are restricting our view to just one equilibrium point of the four possible combinations of stable equilibria, in accordance with Condition 1 of [119]. However, the same analysis could be performed on each of the distinct points, yielding the same result for each combination of equilibria.

The equilibrium point of the reduced system is also known and is the constant point

$$\mathbf{x}^* = \begin{bmatrix} \mathbf{w}_{\text{opt}} \\ \boldsymbol{\mu}^* = +\sqrt{2}\boldsymbol{q}^{\circ\frac{1}{2}} \\ \boldsymbol{\nu}^* = +\mathbf{1}\sqrt{2r} \end{bmatrix} \tag{5.103}$$

where $\boldsymbol{q}$ and r are, respectively, the optimal dual variables corresponding to the inequalities $\mathbf{w} \geq 0$, and the optimal dual variable corresponding to the inequality constraint $\lambda_2(\mathbf{w}) \geq \kappa$, which solve the first order necessary KKT conditions of the optimisation problem, given in (5.63) and (5.64).

As such, we can apply the following change of variables to move the equilibrium points to the origins of the faster systems. Let $\tilde{\mathbf{x}}$ be the deviation of the slow variables from their stable equilibrium point, and likewise let $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{y}}_2$ be the deviation of the fast and fastest variables, resectively, from their stable equilibria also. We have:

$$\tilde{\mathbf{x}} \triangleq \mathbf{x} - \mathbf{x}^* \tag{5.104}$$

$$\tilde{\mathbf{y}_1} \triangleq \mathbf{y_1} - \mathbf{y_1}^* \tag{5.105}$$

$$\tilde{\mathbf{y}_2} \triangleq \mathbf{y_2} - \mathbf{y_2}^* \tag{5.106}$$

We then must redefine the right-hand-side functions too, to account for these change of variables:

$$\mathbf{F}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) = \mathbf{f}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}), \tilde{\mathbf{y_2}} + \mathbf{Y_2}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}))) \tag{5.107}$$

$$\begin{aligned}\mathbf{G_1}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) = {} & \mathbf{g_1}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}), \tilde{\mathbf{y_2}} + \mathbf{Y_2}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}))) \\ & - \epsilon_1 \left(\frac{\partial \mathbf{Y_1}}{\partial \mathbf{x}} \cdot \mathbf{F}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) \right)\end{aligned} \tag{5.108}$$

$$\begin{aligned}\mathbf{G_2}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) = {} & \mathbf{g_2}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}), \tilde{\mathbf{y_2}} + \mathbf{Y_2}(\mathbf{x}^* + \tilde{\mathbf{x}}, \tilde{\mathbf{y_1}} + \mathbf{Y_1}(\mathbf{x}^* + \tilde{\mathbf{x}}))) \\ & - \epsilon_2 \left(\frac{\partial \mathbf{Y_2}}{\partial \mathbf{x}} \cdot \mathbf{F}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) + \frac{\partial \mathbf{Y_2}}{\partial \mathbf{y_1}} \cdot \mathbf{G_1}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\tilde{\mathbf{y_2}}) \right)\end{aligned} \tag{5.109}$$

So that,

$$\mathbf{G_1}(\tilde{\mathbf{x}},\mathbf{0},\mathbf{0}) = \mathbf{0} \tag{5.110}$$

$$\mathbf{G_2}(\tilde{\mathbf{x}},\tilde{\mathbf{y_1}},\mathbf{0}) = \mathbf{0} \tag{5.111}$$

and also that Condition 2 in [119], that the equilibrium point of the slow system lies at the origin, is satisfied:

$$\mathbf{F}(\mathbf{0},\mathbf{0},\mathbf{0}) = \mathbf{0} \tag{5.112}$$

5.5.3 Analyticity of the right hand sides

- Condition 3 in [119] requires that the functions $\mathbf{F}(.)$, $\mathbf{G_1}(.)$, and $\mathbf{G_2}(.)$ and their derivatives with respect to the components of $\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}}$ are continuous in a ball of radius R around the origin.

- Condition 4 is a requirement that the elements of $\mathbf{F}(.)$, $\mathbf{G_1}(.)$, and $\mathbf{G_2}$, and the elements of $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}, \frac{\partial \mathbf{G}_i}{\partial \mathbf{x}}, \frac{\partial \mathbf{G}_i}{\partial \mathbf{y_j}}$ are bounded within the ball around the origin.

- Further to continuity and boundedness, Conditions 5 and 6 specify uniform smoothness conditions on $\mathbf{F}(.)$, and $\mathbf{G_2}(.)$. Specifically, it is required that there is a continuous nonnegative function $v(|\tilde{\mathbf{y}}|)$ with $v(0) = 0$, defining $\tilde{\mathbf{y}} = [\tilde{\mathbf{y_1}}^\top, \tilde{\mathbf{y_2}}^\top]^\top$ such that,

$$|\mathbf{F}(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) - \mathbf{F}(\tilde{\mathbf{x}}, \mathbf{0})| \leq v(|\tilde{\mathbf{y}}|) \tag{5.113}$$

  Likewise, there exists a continuous nonnegative function $u(|\tilde{\mathbf{y_2}}|)$ with $u(0) = 0$ such that

$$|\mathbf{G_1}(\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}}) - \mathbf{G_1}(\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \mathbf{0})| \leq u(|\tilde{\mathbf{y_2}}|) \tag{5.114}$$

All of these conditions are immediately satisfied if $\mathbf{F}(.)$, $\mathbf{G_1}(.)$, and $\mathbf{G_2}(.)$ are analytic functions of $\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}}$.

Lemma 5.4. $\mathbf{F}(\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}})$ *is an analytic function in the vicinity of the equilibrium point, provided Assumption 5.1 holds.*

Proof. Looking at Equations (5.84, 5.85, 5.86), it can be seen that $\mathbf{f}(\mathbf{x}, \mathbf{y_1}, \mathbf{y_2})$ is an analytic function of its variables, provided $\pi_i^{\mathbf{a}^{\circ 2}} \neq -\pi_j^{\mathbf{a}^{\circ 2}}, \forall i,j$ and $\pi_i^{\mathbf{b}^{\circ 2}} \neq -\pi_j^{\mathbf{b}^{\circ 2}}, \forall i,j$. This is apparent as $\mathbf{f}(.)$ is a sum of ratios of polynomials in its variables, and the denominators may only be zero if $\pi_i^{\mathbf{a}^{\circ 2}} = -\pi_j^{\mathbf{a}^{\circ 2}}$ or $\pi_i^{\mathbf{b}^{\circ 2}} = -\pi_j^{\mathbf{b}^{\circ 2}}$. At the equilibrium point, $\boldsymbol{\pi}^{\mathbf{a}^{\circ 2},*} = \frac{\mathbf{1}\mathbf{a}^{*\top}\mathbf{a}^*}{n}, \mathbf{a}^* \neq \mathbf{0}$, so $\pi_i^{\mathbf{a}^{\circ 2},*} = \pi_j^{\mathbf{a}^{\circ 2},*} \neq 0$. Likewise for the spectral radius estimator $\pi_i^{\mathbf{b}^{\circ 2},*} = \pi_j^{\mathbf{b}^{\circ 2},*} \neq 0$

Further to this, observing Equations (5.101) and (5.102), both $\mathbf{Y_1}(\mathbf{x})$ and $\mathbf{Y_2}(\mathbf{x}, \mathbf{y_1})$ are analytic functions in the vicinity of the equilibrium point under Assumption 5.1. Thus $\mathbf{F}(.)$, which is a composition of analytic functions, is analytic. □

Lemma 5.5. $\mathbf{G_1}(\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}})$ *and* $\mathbf{G_2}(\tilde{\mathbf{x}}, \tilde{\mathbf{y_1}}, \tilde{\mathbf{y_2}})$ *are analytic functions in the vicinity of the equilibrium point, provided Assumption 5.1 holds.*

Proof. It can be seen in Equations (5.87-5.96) that both $\mathbf{g_1}(.)$ and $\mathbf{g_2}(.)$ are quadratic functions of their variables, and are thus analytic. Further to this, observing Equations (5.101) and (5.102) both stationary manifolds $\mathbf{Y_1}(.)$ and $\mathbf{Y_2}(.)$ are analytic provided Assumption 5.1 holds. Hence, the function $\mathbf{G_1}(.)$ which is a composition of the analytic functions $\mathbf{g_1}(.)$, $\mathbf{Y_1}(.)$, $\mathbf{Y_2}(.)$ and $\mathbf{F}(.)$ is analytic. The same argument can be made for $\mathbf{G_2}(.)$, which is a composition of the aforementioned analytic functions, $\mathbf{g_2}(.)$, and $\mathbf{G_1}(.)$. □

5.5.4 Stability of the boundary layer systems and reduced system

The final two conditions in [119], Conditions 7 and 8, concern the uniform asymptotic stability of the reduced system and the boundary layer systems. That is, it is required that the origin of the reduced system:

$$\dot{\tilde{\mathbf{x}}} = \mathbf{F}(\tilde{\mathbf{x}}, \mathbf{0}, \mathbf{0}) \tag{5.115}$$

is uniformly asymptotically stable. And it is also required that the boundary layer systems:

$$\frac{\partial \tilde{\mathbf{y_1}}}{\partial \tau_1} = \mathbf{G_1}(\mathbf{x}^*, \tilde{\mathbf{y_1}}, \mathbf{0}) \tag{5.116}$$

and

$$\frac{\partial \tilde{\mathbf{y_2}}}{\partial \tau_2} = \mathbf{G_1}(\mathbf{x}^*, \mathbf{y_1}^*, \tilde{\mathbf{y_2}}) \tag{5.117}$$

when taking $\mathbf{x}^*$ and $\mathbf{y_1}^*$ as any constant value (in a ball around the origin), and defining $\tau_i \triangleq \frac{t}{\epsilon_i}$ are also uniformly asymptotically stable.

The fully distributed system has been designed so that the reduced system Equation (5.69) is locally exponentially stable (Theorem 4.4 in Chapter 4) implying uniform asymptotic stability. Likewise, each boundary layer system, found by taking slower variables as constant and faster variables as having already equilibriated, are simply the ideal eigenvalue estimators (3.7), (5.1), and (5.33), and are thus exponentially stable as proven in Theorem 5.1 and elsewhere [29, 100].

All the conditions from Theorem 1 in [119] hold, and so we may draw the conclusion that there exists a sufficiently small $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2]^\top$ such that $\epsilon_1 \to 0$ and $\frac{\epsilon_2}{\epsilon_1} \to 0$ as $|\boldsymbol{\epsilon}| \to 0$, so that the equilibrium point corresponding to the solution of the optimisation problem is locally exponentially stable. A suitable choice for $\boldsymbol{\epsilon}$ is $[\frac{1}{k_2}, \frac{1}{k_P}]^\top$ where $k_P = k_2^2$. There then exists a $k_P > \frac{1}{\epsilon^*}$ sufficiently large so that the equilibrium point of the entire distributed system is locally exponentially stable.

5.6 Examples

5.6.1 Linear consensus with delay

In this example, we start with a small network of $n = 8$ nodes and $m = 11$ edges, see Figure 5.2. Each edge is assigned an initial unitary weight, $w_{\{i,j\}}(0) = 1$, so that the initial algebraic connectivity $\lambda_2(\mathbf{w}(0)) \approx 0.7611$ and the initial spectral radius is $\lambda_n(\mathbf{w}(0)) \approx 5.8635$. Edge weights are then controlled according to the distributed multi-layer system in (5.84)-(5.96), with the objective of minimising λ_n, whilst holding the constraints that all weights are non-negative, and a mininimum algebraic connectivity of $\lambda_2 \geq 0.5$ is maintained. Discretisation of the continuous time differential equations in this simulation is achieved using the forward Euler method with a time step of 1×10^{-2}. As the simulation progresses, edge weights evolve over time and eventually settle to a stationary value, Figure 5.3(a). The edge weights in the network directly affect the graph Laplacian, resulting in the eigenvalues changing over time, Figure 5.3(b), and for this particular example eigenvalues at the solution $\mathbf{w} = \mathbf{w}_{\text{opt}}$ are distinct, so that edge weights converge as expected. At the end of the simulation, $t = 2000$, the algebraic connectivity has settled to its lower bound as expected $\lambda_2(\mathbf{w}(2000)) \approx 0.5000$, and the spectral radius has decreased to $\lambda_n(\mathbf{w}(2000)) \approx 3.2211$, shrinking to approximately 55% of its initial value. This result agrees with the optimal value found using a centralised SDP solver, with the relative error in the minimal spectral radius being less than 0.001%.

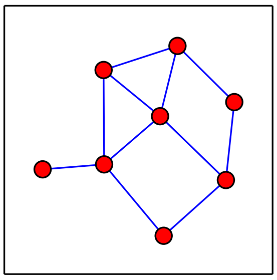
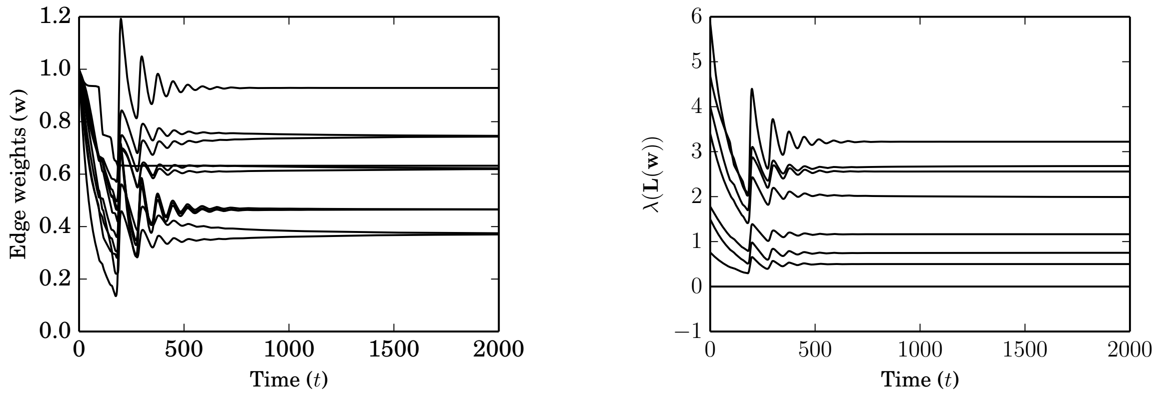


Figure 5.2: The initial network, at time zero, with all edge weights of $w_{\{i,j\}}(0) = 1$.



(a) Edge weights change over time from their initial value of $\mathbf{w}(0) = \mathbf{1}$, and converge onto the stationary point $\mathbf{w}_{\text{opt}}$.

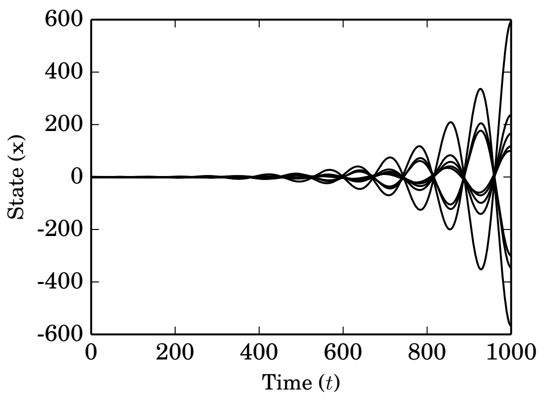
(b) As edge weights in the network change, the spectrum of the graph Laplacian also changes. As required by the optimisation problem, the algebraic connectivity maintains the bound $\lambda_2 \geq 0.5$, whilst the spectral radius λ_n is minimised.

Figure 5.3: Trajectories of the edge weights and the eigenvalues of the graph Laplacian, as the simulation progresses. Control parameters in this simulation are: $k_w = 1.5 \times 10^{-3}$,

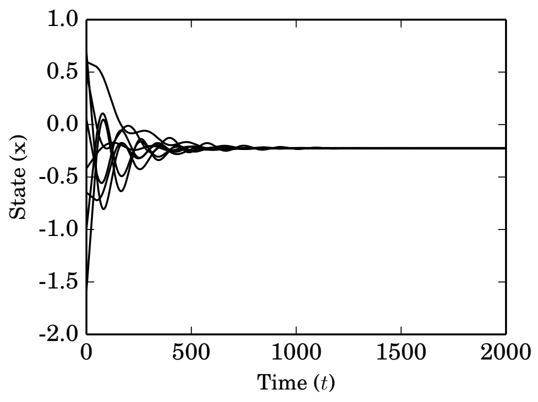
The simple linear consensus protocol $\dot{\mathbf{x}}(t) = -0.01 \times \mathbf{L}(\mathbf{w})\mathbf{x}(t-\tau)$ with homogeneous time delay $\tau = 40$ is run both on the intial and final states of the network, Figure 5.4. From the bound given in [45], we know that the system is only stable on the consensus mode if $\lambda_n \leq 100 \times \frac{\pi}{2\times 40} \approx 3.9270$. We see that the in the intial network (at time $t = 0$), the spectral radius exceeds this bound, and as a consequence trajectories in Figure 5.4(a) diverge. When the same system is realised on the near-optimal network, Figure 5.4(b), trajectories now converge as the consensus mode is stable.

5.6.2 Ramifications of Assumption 5.1 not holding

To illustrate the scenario where Assumption 5.1 does not hold, we use a larger network of $n = 50$, $m = 118$, so that there are $n - 1 = 49$ non-trivial eigenvalues in the connected graph. As $\lambda_n(\mathbf{w})$ is decreased the spectrum of the graph Laplacian is compressed into a smaller region, increasing the likelihood that at the optimal edge weights $\lambda_2(\mathbf{w})$ and $\lambda_n(\mathbf{w})$ will not be distinct, Figure 5.6(b).



(a) The spectral radius of the initial network exceeds the calculated bound $\lambda_n \approx 5.8635 > 3.9270$, and so the consensus mode is unstable; trajectories diverge.



(b) At time $t = 2000$, edge weights in the network have converged onto values reducing the spectral radius of the graph Laplacian below the bound $\lambda_n \approx 3.2211 < 3.9270$; consensus is stable.

Figure 5.4: The simple linear consensus protocol with homogeneous delay of $\tau = 40$ is run, both on the initial network with homogenous weights $\mathbf{L}(\mathbf{w}(0) = \mathbf{1})$, and on the final state of the network $\mathbf{L}(\mathbf{w}(2000) \approx \mathbf{w}_{\text{opt}})$.

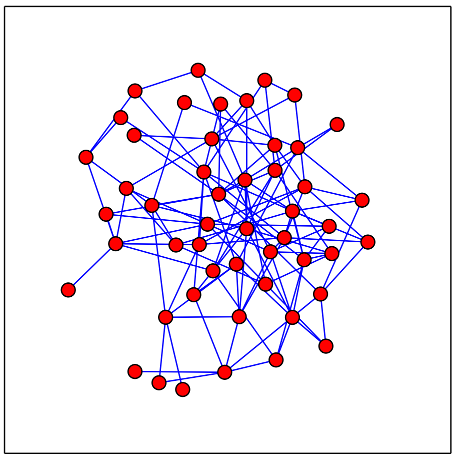
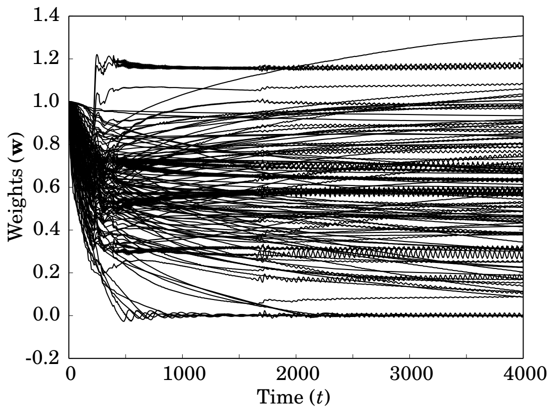
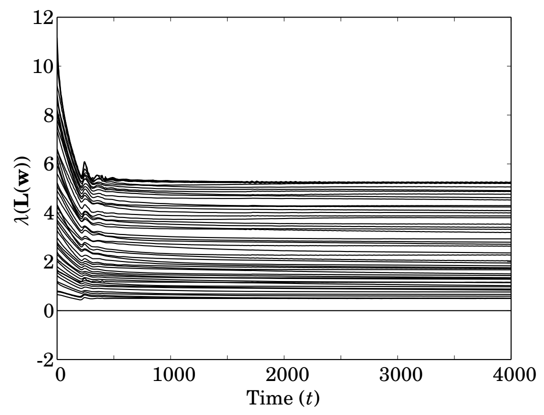


Figure 5.5: A larger network, $n = 50$, $m = 118$, at time zero, with edge weights each of $w_{\{i,j\}}(0) = 1$.

In this example, the spectral radius of the graph Laplacian is again decreased over time, from an initial value of $\lambda_n(\mathbf{w}(0)) \approx 11.1450$ to $\lambda_n(\mathbf{w}(4000)) \approx 5.2514$, whilst the algebraic connectivity falls from $\lambda_2(\mathbf{w}(0)) \approx 0.6589$ to $\lambda_2(\mathbf{w}(4000)) \approx 0.4988$. Note that this time at $t = 4000$, the algebraic connectivity lies slightly outside of its constraint. This is a result of Assumption 5.1 failing. At the optimal edge weights $\mathbf{w}_{\text{opt}}$, the algebraic connectivity has a multiplicity greater than unity. Therefore, the algebraic connectivity estimator subsystem does not converge to a specific



(a) Edge weight trajectories. In this example a number of edge weights are tending towards a value of zero.



(b) Eigenvalue trajectories. Even though at the optimal edge weights λ_n and λ_2 are not disitnct, the system still peforms as intended, minimising λ_n and maintaining $\lambda_2 \geq 0.5$.

Figure 5.6: The edge weight adaptation strategy is performed on the larger example network. The time step for this simulation is again 1×10^{-2}.

eigenvector associated with λ_2 but converges instead onto its associated eigenspace. The effect of this is that the algebraic connectivity derivative estimates, $\frac{\partial \lambda_2}{\partial \mathbf{w}}$, will be incorrect as the value is not well defined for non-distinct eigenvalues. This error forces the edge weights off of the optimal values, and a persistent oscillation in the neighbourhood of the optimum is set up. The magnitude and frequency of these oscillations can be controlled by the size of the separation in time-scales between the weight dynamics and the eigenvalue estimation, as this controls how far the edge weights can overshoot before the eigenvalue estimator re-converges on the correct value for $\frac{\partial \lambda_2}{\partial \mathbf{w}}$. We would however like to compare the values of the extremal eigenvalues to those found using a centralised solver, specifically using an SDP formulation presented in [77]. The centralised solver finds optimal eigenvalues of $\lambda_{2,opt} \approx 0.5000$ and $\lambda_{n,opt} \approx 5.2178$. Comparing these values to the extremal eigenvalues at $t = 4000$, of $\lambda_2(\mathbf{w}(4000)) \approx 0.4988$ and $\lambda_n(\mathbf{w}(4000)) \approx 5.2514$, we can see that the decentralised method performs admirably despite Assumption 5.1 not holding, finding edge weights that result in a cost just 0.64% greater than those found using a centralised solver. In contrast, the decentralised method presented in [98] can only adapt edge weights up to the point when the extremal eigenvalues become indistinct and the algorithm breaks down, resulting in a greater disparity between the solution to the optimisation problem and the result from the decentralised method. It is foreseen that this issue of non-distinct extremal eigenvalues will grow with increasing network size.

5.7 Summary

In this Chapter, the decentralised algebraic connectivity estimator from [29] was modified so that, instead, the largest eigenvalue of the Graph Laplacian matrix, its spectral radius, and its partial derivatives with respect to the edge weights, could be estimated in a wholly decentralised fashion. The local exponential stability of this idealised sytem was proven through a linearisation analysis, and by noticing that the system is a gradient system, both in cases when the eigenvalues are distinct and when there exist eigenvalues with higher multiplicities. The stable equilibrium manifold was also characterised by its shape according to the algebraic multiplicity of the spectral radius.

We then proceeded to implement a fully decentralised implementation of this estimator system, using two Proportional-Integral (PI) average consensus layers [100]. Stability of the fully decentralised system was then investigated and proven, and a conservative bound (5.58) given on the required separation in time-scale between the eigenvalue estimator system, and the two PI average consensus subsystems on which it relies.

This fully decentralised spectral radius estimator was then utilised in conjuction with one of the weight adaptation laws presented in Chapter 4, and the decentralised algebraic connectivity estimator presented in Chapter 3, to solve a network optimisation problem to maximise robustness to time-delay in a linear consensus system with uniform time-delay. It was then proven using methods from singular perturbation theory that there exists a sufficient separation in time-scale for local exponential stability between the three layers of this fully decentralised system (illustrated in Figure 5.1): the weight adaptation layer, the eigenvalue estimation layer, and the PI average consensus layer, under the assumption that both the algebraic connectivity and the spectral radius are distinct in the optimal network. The effects of this assumption not holding were investigated numerically, showing instead that the system converges to the vicinity of the solution, with persistent oscillations about the optimal value.

In the following Chapter, we will suggest two further modifications to the eigenvalue estimator presented in this Chapter, so that two important functions of all the non-zero eigenvalues may be estimated in a decentralised manner. Specifically, these two functions are the total effective graph resistance and the reduced Laplacian determinant, which are related to the harmonic and geometric means of the spectrum of graph Laplacian matrix, respectively.

CHAPTER 6

Functions of All Non-trivial Laplacian Eigenvalues

Previously we have focussed on optimising the extremal non-trivial eigenvalues of the graph Laplacian matrix: the smallest non-zero eigenvalue in the Laplacium spectrum, the algebraic connectivity λ_2, and the largest eigenvalue in the Laplacian spectrum, which is the spectral radius of the graph Laplacian λ_n. These extremal eigenvalues, and the ratio between them, have important ramifications on the consensus or synchronisation dynamics that occur on the network. However, we now turn towards some other spectral functions of the graph Laplacian matrix, namely the total effective graph resistance[122][1], which is related to the commute times of random walkers in networks among other problems [123, 124], and the reduced Laplacian determinant, which counts the number of spanning trees in a graph[125], and so can be used a measure of network robustness[126, 127].

Unlike the previous spectral functions, the algebraic connectivity and the spectral radius, the total effective graph resistance and the reduced graph Laplacian determinant are functions of all $n-1$ non-trivial eigenvalues of the graph Laplacian. Thus, if we were to use a method to make decentralised estimates of all non-trivial eigenvalues (this could be achieved, for example, by successive deflation on the graph Laplacian matrix in the same manner as the consensus mode is deflated), the amount of calculations and memory that each agent in the network would be required to undertake would grow linearly with the size of the network; this is clearly not desirable in a decentralised control application where we assume each individual agent in the network has access to some small finite amount of memory and computational power. As we will see, by introducing stochasticity into the network, and utilising the effects of this, we can excite all non-trivial modes in the network to make estimates of these spectral functions using a relatively few and constant number of variables in each node.

[1]Also known as the Kirchoff Index.

We begin by defining and providing some background on the total effective graph resistance[2], before motivitating the importance of this function by looking at linear consensus under diffusive coupling when subject to white noise. We then go on to see how we can use the introduction of white noise to allow the network to make a decentralised estimate of its graph resistance, before we proceed to find a method for the estimation of the partial derivatives of this function for use with the weight adaptation strategies presented in Chapter 4. During this process we will also touch on the reduced Laplacian determinant and find an unexpected connection between this quantity and the graph resistance.

6.1 Graph Resistance

Arising from an electrical interpretation of a network [122], the total effective graph resistance is another useful metric for networks [128, 76], particularly in the areas of network robustness [127, 129], commute times of random walks [123], and power dissipation in electrical circuits. In particular, it can be shown that if each edge of weight $w_{i,j}$ in a network is replaced with a resistor of $\frac{1}{w_{i,j}}$ Ohms [3], then the resistance distance between two nodes k and l is exactly the voltage difference measured when the resistance load between k and l is driven by a current source of 1 Ampere [123]. An equivalent definition of the resistance distance is given in [122, 128]:

Definition 6.1 (Resistance Distance, $\rho_{i,j}$ [128])**.** Applying a potential of v_i Volts to node i and a potential of v_j Volts to node j, the resistance distance between i and j, $\rho_{i,j}$ is defined,

$$\rho_{i,j} = \frac{v_i - v_j}{I} \tag{6.1}$$

where I is the resultant current between nodes i and j. △

Example 6.1. *Take for example the weighted graph as depicted in Figure 6.1, replacing each edge of weight $w_{\{i,j\}}$ with a resitor of $\frac{1}{w_{i,j}}$ Ohms. The resistance distance between nodes 1 and 4 can be computed simply using the rules for series and parallel combination of resistors: It can be seen that the path $(2,3,4)$ consists of two resistors in series, so this path can be simplified using the series addition of resistors to an effective resistance of $3\Omega + 1\Omega = 4\Omega$; this leaves two resistances in parallel between nodes 2 and 4, which can be simplified using parallel combination of resistors, resulting in an effective resistance of $\left(\frac{1}{4}+\frac{1}{4}\right)^{-1} = 2\Omega$ between nodes 2 and 4; finally the resistance distance between nodes 1 and 4 can be computed, again using the series addition of resistors, to find that the resistance distance $\rho_{1,4} = 2\Omega + 2\Omega = 4\Omega$.*

This method for calculating resistance distances will become increasingly laborious for larger networks with multiple paths between nodes. However, the resistance distance is strongly linked

[2]We also refer to this quantity simply as the graph resistance, and denote it using the symbol Ω for brevity.

[3]That is a conductance of $w_{i,j}$ Siemens, agreeing with the notion that a larger edge weight between two nodes indicates that the nodes are better connected.

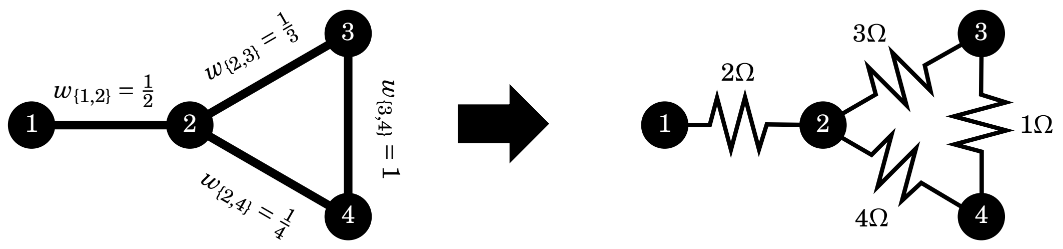

$w_{\{1,2\}} = \frac{1}{2}$
$w_{\{2,3\}} = \frac{1}{3}$
$w_{\{3,4\}} = 1$
$w_{\{2,4\}} = \frac{1}{4}$
1
2
3
4
2Ω
3Ω
1Ω
4Ω

Figure 6.1: Transforming edge weights to resistors, such that each resistor is $\frac{1}{w_{i,j}}$ Ohms.

to the graph Laplacian through the following result, also found in [128], which provides a far more amenable method for computing resistance distances:

$$\rho_{i,j} = (\mathbf{e}_i - \mathbf{e}_j)^\top \mathbf{L}^\dagger (\mathbf{e}_i - \mathbf{e}_j) \tag{6.2}$$

Again, $\mathbf{L}^\dagger$ is the Moore-Penrose pseudoinverse of the graph Laplacian, and $\mathbf{e}_i$ is the vector of zeros, save for the value of the i^{th} element being equal to one. Specifically, we have,

$$\mathbf{L}^\dagger = \left(\mathbf{L} + \frac{\mathbf{1}\mathbf{1}^\top}{n}\right)^{-1} - \frac{\mathbf{1}\mathbf{1}^\top}{n} \tag{6.3}$$

Example 6.2. *Again using the simple graph illustrated in Figure 6.1, we find its graph Laplacian,*

$$\mathbf{L} = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & \frac{13}{12} & -\frac{1}{3} & -\frac{1}{4} \\ 0 & -\frac{1}{3} & \frac{4}{3} & -1 \\ 0 & -\frac{1}{4} & -1 & \frac{5}{4} \end{bmatrix} \tag{6.4}$$

so that the Moore-Penrose pseudoinverse $\mathbf{L}^\dagger$ can be computed using Equation (6.3)*,*

$$\mathbf{L}^\dagger = \begin{bmatrix} 1.5546875 & 0.0546875 & -0.7890625 & -0.8203125 \\ 0.0546875 & 0.5546875 & -0.2890625 & -0.3203125 \\ -0.7890625 & -0.2890625 & 0.7421875 & 0.3359375 \\ -0.8203125 & -0.3203125 & 0.3359375 & 0.8046875 \end{bmatrix} \tag{6.5}$$

It is then straightforward to calculate any resistance distance with Equation (6.2)*. For example,*

$$\begin{aligned} \rho_{1,4} &= (\mathbf{e}_1 - \mathbf{e}_4)^\top \mathbf{L}^\dagger (\mathbf{e}_1 - \mathbf{e}_4) \\ &= \mathbf{L}^\dagger_{1,1} - \mathbf{L}^\dagger_{1,4} - \mathbf{L}^\dagger_{4,1} + \mathbf{L}^\dagger_{4,4} \\ &= 1.5546875 - (2 \times -0.8203125) + 0.8046875 = 4 \end{aligned} \tag{6.6}$$

Definition 6.2 (Total effective graph resistance, Ω [122]). For undirected graphs, the total effective graph resistance Ω is defined as the sum of all distinct pairwise resistance distances:

$$\Omega = \sum_{i<j} \rho_{i,j} = \frac{1}{2}\sum_{i}\sum_{j} \rho_{i,j} \tag{6.7}$$

△

In this manner the total effective resistance quantifies the size of a network in terms of the sum over all resistance distances. However, unlike the common notion of distance between two nodes in a graph (the length of the shortest path), the resistance distance also takes into account the number and quality of back-up paths between nodes, and hence is appropriate for quantifying the robustness of connection between two nodes.

A surprising result is that the total effective graph resistance can be exactly calculated from the eigenvalues of the graph Laplacian:

Theorem 6.1 (Relationship between the Laplacian spectrum and Ω [122, 76]). *The total effective graph resistance Ω of a connected undirected graph is equal to n times the sum of the reciprocals of the non-trivial eigenvalues of the Laplacian spectrum of that graph:*

$$\Omega = n\sum_{i=2}^{n} \frac{1}{\lambda_i} \tag{6.8}$$

Proof. Starting from the result Equation (6.2), and expanding,

$$\begin{aligned} \rho_{i,j} &= (\mathbf{e}_i - \mathbf{e}_j)^\top \mathbf{L}^\dagger (\mathbf{e}_i - \mathbf{e}_j) \\ &= L^\dagger_{i,i} + L^\dagger_{j,j} - L^\dagger_{i,j} - L^\dagger_{j,i} \end{aligned} \tag{6.9}$$

Then taking the sum over all distinct pairs,

$$\Omega = \sum_{i<j} \rho_{i,j} = (n-1)\sum_{i} L^\dagger_{i,i} - 2\sum_{i<j} L^\dagger_{i,j} \tag{6.10}$$

$$= (n-1)\sum_{i} L^\dagger_{i,i} - \overbrace{\left(2\sum_{i<j} L^\dagger_{i,j} + \sum_{i} L^\dagger_{i,i}\right)}^{\text{Equal to 0}} + \sum_{i} L^\dagger_{i,i} \tag{6.11}$$

$$= n\sum_{i} L^\dagger_{i,i} \tag{6.12}$$

$$= n\text{Trace}\{\mathbf{L}^\dagger\} \tag{6.13}$$

$$= n\sum_{i=2}^{n} \frac{1}{\lambda_i} \tag{6.14}$$

Using the fact that $\mathbf{L}^\dagger$ also has row sums of zero, and that the non-zero eigenvalues of $\mathbf{L}^\dagger$ are simply the reciprocals of the non-zero eigenvalues of $\mathbf{L}$. □

6.2 Linear Consensus with Additive White Noise

To demonstrate the practical importance of the total effect graph resistance, we can consider the case of simple linear consensus when all nodes are perturbed by identical and independently distributed white noise of intensity σ. This system can be represented using the stochastic differential equation (SDE):

$$d\mathbf{x}(t) = -\mathbf{L}\mathbf{x}(t)dt + \sigma\mathbf{I}_n d\mathbf{W} \tag{6.15}$$

where $\mathbf{W}(t)$ is a standard n dimensional Wiener process. Defining the expected value $\boldsymbol{\mu}(t) \triangleq E[\mathbf{x}(t)]$ and the covariance matrix $\Sigma_x(t) \triangleq E[(\mathbf{x}(t)-\boldsymbol{\mu}(t))(\mathbf{x}(t)-\boldsymbol{\mu}(t))^\top]$, it can be shown [129] that:

$$\dot{\boldsymbol{\mu}}(t) = -\mathbf{L}\boldsymbol{\mu}(t) \tag{6.16}$$

and

$$\dot{\Sigma}_x(t) = -\mathbf{L}\Sigma_x(t) - \Sigma_x(t)\mathbf{L} + \sigma^2\mathbf{I}_n \tag{6.17}$$

From the covariance matrix dynamics and the fact that $\mathbf{L}$ has a zero eigenvalue, it can be seen that states will grow without bound over time; the additive noise induces a drift in the consensus, as seen in Figure 6.2. Thus, although the expected value of the states will tend to the arithmetic

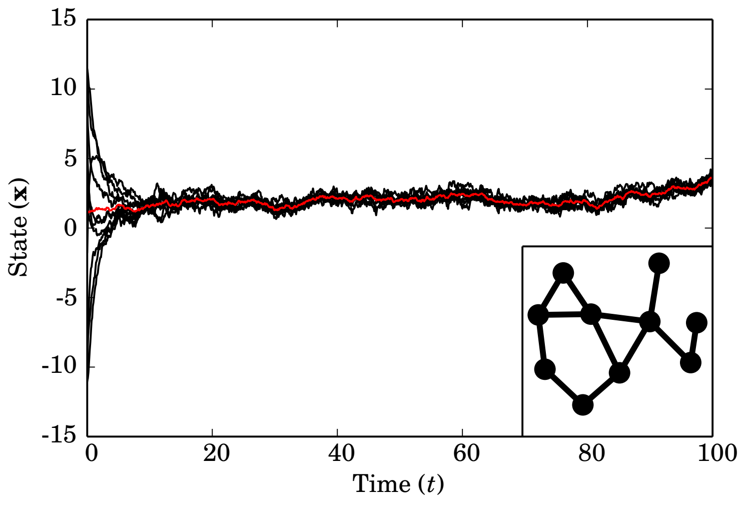


Figure 6.2: Starting from an initial state $\mathbf{x}(0)$ sampled from the normal distribution with mean of zero and variance of 10, agents come together under the action of Equation 6.15, with $\sigma = \frac{1}{2}$. The arbitrarily chosen network is illustrated inset, and all edge weights are chosen to be constant and of value 1. It can be seen that although the dispersion decreases from the inital value, the mean of the group (highlighted in red) drifts over time. Numerical solution of the stochastic differential equation is achieved using the Euler-Maruyama method with a time step of 1×10^{-3}.

mean of the initial expected values, their variance will grow without bound. Restricting the view solely to the disagreement subspace $\mathbf{y} = \mathbf{Q}\mathbf{x}$ where $\mathbf{Q}\mathbf{1} = \mathbf{0}$, $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}_{n-1}$ and $\mathbf{Q}^\top\mathbf{Q} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, the same analysis can be performed. Introducing the reduced Laplacian $\overline{\mathbf{L}} \triangleq \mathbf{Q}^\top\mathbf{L}\mathbf{Q}$, which has the same eigenvalues as the Laplacian matrix, save for the single zero eigenvalue associated with the consensus mode, which is removed, we now have:

$$d\mathbf{y}(t) = -\overline{\mathbf{L}}\mathbf{y}(t)dt + \sigma\mathbf{Q}d\mathbf{W} \tag{6.18}$$

so that

$$\dot{\Sigma}_y(t) = -\overline{\mathbf{L}}\Sigma(t) - \Sigma(t)\overline{\mathbf{L}} + \sigma^2\mathbf{I}_{n-1} \tag{6.19}$$

Thus the solution for the stationary covariance matrix can be found by solving the Lyapunov equation:

$$\overline{\mathbf{L}}^\top\Sigma_y + \Sigma_y\overline{\mathbf{L}} = -\sigma^2\mathbf{I} \tag{6.20}$$

which has a positive definite solution for Σ_y so long as $\overline{\mathbf{L}}$ is Hurwitz, which is guaranteed if the graph is connected and weights are non-negative. Thus the variance in the disagreement subspace no longer grows without bound. It can further be shown [129] that the expected two-norm of the disagreement vector will tend to a finite value:

$$\lim_{t\to\infty} E\left[||\mathbf{y}||_2^2\right]^{\frac{1}{2}} = \sqrt{\frac{\sigma^2\Omega}{2n}} \tag{6.21}$$

Alternatively, the Lyapunov equation becomes trivial to solve if we assume we can diagonalise the reduced Laplacian using some orthonormal basis $\overline{\mathbf{V}}$ so that $\overline{\mathbf{V}}^\top\overline{\mathbf{L}}\overline{\mathbf{V}} = \overline{\mathbf{\Lambda}} = \text{diag}[\lambda_2, \lambda_3, \ldots, \lambda_n]$. Let $\mathbf{y} = \overline{\mathbf{V}}\mathbf{z}$ so that:

$$d\mathbf{z}(t) = -\overline{\mathbf{\Lambda}}\mathbf{z}(t)dt + \sigma\overline{\mathbf{V}}\mathbf{Q}d\mathbf{W} \tag{6.22}$$

$$\dot{\Sigma}_z(t) = -\overline{\mathbf{\Lambda}}\Sigma_z(t) - \Sigma_z(t)\overline{\mathbf{\Lambda}} + \sigma^2\mathbf{I}_{n-1} \tag{6.23}$$

And thus, the stationary covariance matrix satisfies the Lyapunov equation:

$$\overline{\mathbf{\Lambda}}\Sigma_y + \Sigma_y\overline{\mathbf{\Lambda}} = -\sigma^2\mathbf{I} \tag{6.24}$$

$$\overline{\mathbf{\Lambda}}\Sigma_y + (\overline{\mathbf{\Lambda}}\Sigma_y)^\top = -\sigma^2\mathbf{I} \tag{6.25}$$

Solving this Lyapunov equation for the stationary covariance matrix yields,

$$\text{Var}[z_i] = E[(z_i - E[z_i])^2] = \frac{\sigma^2}{2\lambda_i} \tag{6.26}$$

$$\text{Cov}[z_i, z_j] = 0 \tag{6.27}$$

but then $E[z_i] = 0$ for all z_i so that:

$$E[(z_i)^2] = \frac{\sigma^2}{2\lambda_i} \tag{6.28}$$

$$E[\mathbf{z}^\top\mathbf{z}] = E[\mathbf{y}^\top\mathbf{y}] = \frac{\sigma^2}{2}\sum_{i=2}^{n}\frac{1}{\lambda_i} \tag{6.29}$$

$$= \frac{\sigma^2\Omega}{2n} \tag{6.30}$$

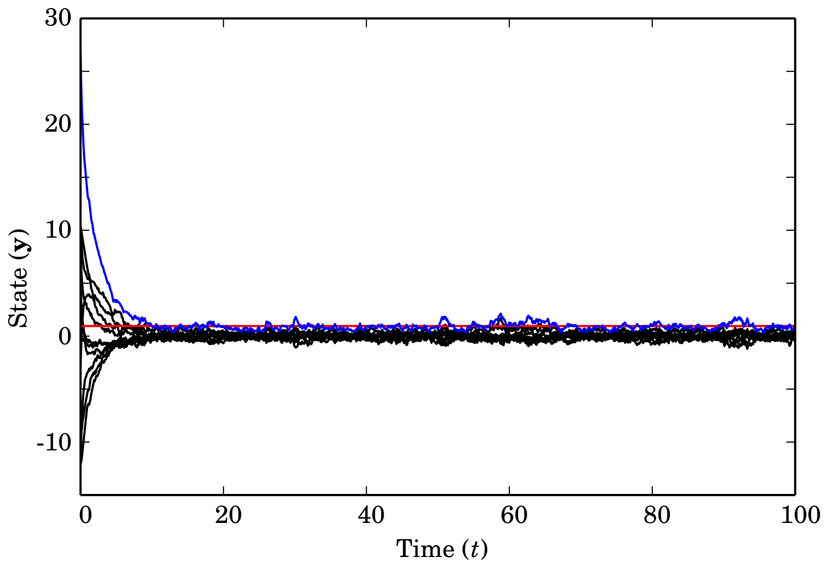


Figure 6.3: Using the simulation depicted in Figure 6.2, we plot the disagreement vector $\mathbf{y}$ in black, and the two-norm $||\mathbf{y}||_2$ in blue. The quantity $\sqrt{\frac{\sigma^2\Omega}{2n}}$ for the chosen graph with $\sigma = \frac{1}{2}$, $n = 10$, and $\Omega = 72.7436\ldots$ is marked by the red line. Note that $||\mathbf{y}||_2$ tends towards this mark, but never settles due to the stochastic nature of the process. Again numerical solution of the stochastic differential equation is achieved using the Euler-Maruyama method with a time step of 1×10^{-3}.

This result, which is illustrated in Figure 6.3, implies that we can formulate a decentralised estimator for the total effective graph resistance, by tracking an estimated value for the mean of $\mathbf{y}^\top\mathbf{y}$, similar to the previous method used to estimate λ_2 and λ_n. Specifically from Equation 6.21, we can estimate the total graph resistance by using the equation:

$$\Omega = \frac{2n \lim_{t\to\infty} E\left[||\mathbf{y}||_2^2\right]}{\sigma^2} \tag{6.31}$$

As in previous chapters, the problem now is to decentralise the computation of all the quantities needed to carry out the estimation of Ω.

6.3 Perturbed linear consensus with mean-shifting

Rather than simply estimating the disagreement vector $\mathbf{y}$ by tracking the mean of $\mathbf{x}$ (which can be accomplished with a PI consensus estimator) and taking the difference, we can instead introduce a negative feedback into the system so that the variance of $\mathbf{x}$ will no longer grow without bound.

We introduce the first system:

$$d\mathbf{x} = (-k_1\mathbf{b} - k_2\mathbf{L}\mathbf{x})dt + d\mathbf{W} \tag{6.32}$$

$$\dot{\mathbf{b}} = k_\gamma(\mathbf{x} - \mathbf{b}) - k_P\mathbf{L}\mathbf{b} + k_I\mathbf{L}\mathbf{c} \tag{6.33}$$

$$\dot{\mathbf{c}} = -k_I\mathbf{L}\mathbf{b} \tag{6.34}$$

Comparing this system to (6.15), we have chosen to set $\sigma = 1$, we scale the strength of the consensus term by the control parameter k_2 as in the λ_2 and λ_n estimators, and have added a PI average consensus estimator to track the mean of $\mathbf{x}$, $\mathbf{b} \to \mathbf{1}\langle\mathbf{x}\rangle$. Here $\mathbf{b}$ is the vector of proportional variables and $\mathbf{c}$ is the vector of integral variables in this PI average consensus estimator, with k_γ, k_P, and k_I, being as usual the control parameters: respectively, the input gain, the proportional gain, and the integral gain. The local estimates of the mean of $\mathbf{x}$ are fed back into (6.32), scaled by the control parameter k_1 to drive the mean towards zero, and constitutes the mean-shifting portion of the estimator.

To track the quantity $\mathbf{x}^\top\mathbf{x}$, a second PI consensus estimator is utilised in much the same way as the second PI estimator is used in the λ_2 and λ_n estimators (Equation (3.15) and Equation (5.45)) to track the norm of the vector. Specifically, we can use the additional equations,

$$\dot{\mathbf{d}} = k_\gamma(\mathbf{x}^{\circ 2} - \mathbf{d}) - k_P\mathbf{L}\mathbf{d} + k_I\mathbf{L}\mathbf{e} \tag{6.35}$$

$$\dot{\mathbf{e}} = -k_I\mathbf{L}\mathbf{e} \tag{6.36}$$

so that $\mathbf{d}$ is the vector of proportional variables that estimates $\frac{\mathbf{1}\mathbf{x}^\top\mathbf{x}}{n}$ and $\mathbf{e}$ is a vector of integrator variables. Again k_γ, k_P and k_I, are the control parameters for the input gain, the proportional gain, and the integrator gain.

It is worth pointing out that these two sets of local variables are decoupled from the previous set of equations, unlike the previous decentralised estimators. Finally, as the total effective graph resistance is related to $E[\mathbf{y}^\top\mathbf{y}]$, and typically the PI consensus estimators converge much faster than (6.32), so that $\mathbf{d} \to \mathbf{1}\langle\mathbf{x}^{\circ 2}\rangle = \frac{\mathbf{1}\mathbf{x}^\top\mathbf{x}}{n}$, it is sensible to use a low-pass filter on $\mathbf{d}$. For simplicity, we simply use a first order low-pass filter with gain of k_L on $\mathbf{d}$, although it may prove beneficial to implement a higher order low-pass filter. In particular we choose a simple linear filter dynamics of the form:

$$\dot{\mathbf{p}} = k_L(\mathbf{d} - \mathbf{p}) \tag{6.37}$$

Theorem 6.2 (Decentralised estimation of the total effective graph resistance)**.** *Using the decentralised system of SDEs described in* (6.32) *to* (6.37)*, and provided that* $k_1, k_2 > 0$*, with* $k_\gamma, k_P, k_I >> k_1, k_2$ *and* k_L *suitably small, each node can make a decentralised estimate of the total effective graph resistance by using the equation:*

$$\widetilde{\Omega}_{(i)} = 2k_2 n\left(np_i - \frac{1}{2k_1}\right) \tag{6.38}$$

Proof. This result is derived again using the diagonalised equation. We assume that the PI consensus estimators converge much faster than Equation (6.32) (i.e. $k_\gamma, k_P, k_I >> k_1, k_2$) so that we can study the reduced system (substituting the stationary value of $\mathbf{b} \to \frac{1}{n}\mathbf{1}\mathbf{1}^\top\mathbf{x}$):

$$d\mathbf{x} = \left(-\frac{k_1\mathbf{1}\mathbf{1}^\top}{n} - k_2\mathbf{L}\right)\mathbf{x}dt + d\mathbf{W} \tag{6.39}$$

Transforming to a diagonal form using the orthonormal modal matrix $\mathbf{V}$ so that $\mathbf{V}^\top\mathbf{L}\mathbf{V} = \mathbf{\Lambda} = \text{diag}[0, \lambda_2, \lambda_3, \ldots, \lambda_n]$ and using the substitution $\mathbf{x} = \mathbf{V}\boldsymbol{\xi}$ we arrive at:

$$d\boldsymbol{\xi} = \underbrace{\left(-k_1\begin{bmatrix}1 & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} - k_2\mathbf{\Lambda}\right)}_{\mathbf{A}}\boldsymbol{\xi}dt + \mathbf{V}^\top d\mathbf{W} \tag{6.40}$$

where $\mathbf{A}$ is Hurwitz, so that $\boldsymbol{\mu}_\xi \to \mathbf{0}$ and Σ_ξ is bounded. The covariance matrix follows a deterministic trajectory

$$\dot{\Sigma}_\xi = \mathbf{A}\Sigma_\xi + \Sigma_\xi\mathbf{A} + \mathbf{I} \tag{6.41}$$

implying that the stationary covariance matrix has:

$$E[\xi_1^2] = \frac{1}{2k_1} \tag{6.42}$$

$$E[\xi_i^2] = \frac{1}{2k_2\lambda_i}\ \forall i = 2, \ldots, n \tag{6.43}$$

$$E[\xi_i\xi_j] = 0\ \forall i \neq j \tag{6.44}$$

Therefore, in the second PI consensus estimator, given in Equations (6.35) and (6.36):

$$E[\mathbf{d}(t)] \to \frac{\mathbf{1}}{n}\left(\frac{1}{2k_1} + \sum_{i=2}^{n}\frac{1}{2k_2\lambda_i}\right), \text{ as } t \to \infty \tag{6.45}$$

The variable $\mathbf{d}(t)$ is passed to a first order low pass filter, so that $\mathbf{p}(t) \to E[\mathbf{d}(t)]$. Substituting and rearranging, each node can make a decentralised estimate of the total effective graph resistance, $\widetilde{\Omega}_{(i)}$, by computing:

$$\widetilde{\Omega}_{(i)} = k_2 n\left(2np_i - \frac{1}{k_1}\right) \tag{6.46}$$

and $\widetilde{\Omega}_{(i)} \to \Omega$ as $t \to \infty$ if $k_L << 1$. It is required that the low pass filter gain k_L in Equation (6.37) is suitably small so that p_i tends to $E[\mathbf{d}(t)]$, and the high frequency effects of the introduced noise are filtered out. □

Example 6.3. *To demonstrate the efficacy of this decentralised estimator, we run the equations* (6.32) *to* (6.37) *on the example graph used in Figures 6.2 and 6.3. The control parameters used are: $k_1 = 5$, $k_2 = 1$, $k_\gamma = 20$, $k_P = 50$, $k_I = 50$, and $k_L = 5 \times 10^{-4}$. The numerical solution to the* *stochastic differential equations is found using the Euler-Maruyama method with a time step of* 1×10^{-3}.

By direct computation of the eigenvalues of the graph Laplacian, it is known that the total effective graph resistance of this network is $\Omega = 72.7436\ldots$ *when all edge weights are equal in value to one. The decentralised estimates of the total effective graph resistance as computed through Equation* (6.38) *are plotted in black in Figure 6.4, and the ground truth is marked in red.*

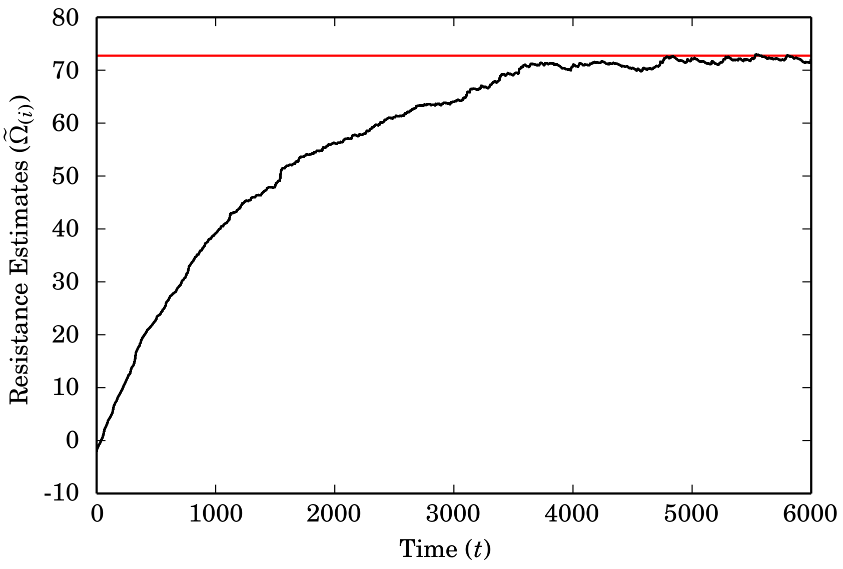


Figure 6.4: Plot showing the trajectories of the decentralised estimates of the total effective graph resistance, as computed through Equations (6.32) to (6.37) and (6.38). The estimates approach the region of the solution but never settle down due to the stochastic nature of the estimator.

6.3.1 The quantity $E[(x_i - x_j)^2]$

As the estimator, decribed in the set of SDEs from (6.32) to (6.37), looks superficially quite similar to the previous decentralised estimators, the quantity $E[(x_i - x_j)^2]$ may be interesting, as this was previously related to the partial derivatives (sub-gradients) of the estimated eigenvalues. As such we can introduce one more low-pass filtered variable which is decoupled from the main system:

$$\dot{\mathbf{r}} = k_L \left((\mathbf{P}\mathbf{x})^{\circ 2} - \mathbf{r} \right) \tag{6.47}$$

The matrix $\mathbf{P}$ here is again the standard oriented incidence matrix, so that $\mathbf{r}_{q(\{i,j\})} \to E[(x_i - x_j)^2]$, and k_L is a control parameter to act as a first order low pass filter.

Again, in analysing the whole system, it is convenient to use the diagonalised equations using the modal matrix of the graph Laplacian $\mathbf{V}$. It has been previously shown that at long times:

$$E[\xi_i] = 0 \; \forall i \tag{6.48}$$

$$E[\xi_1^2] = \frac{1}{2k_1} \tag{6.49}$$

$$E[\xi_i^2] = \frac{1}{2k_2\lambda_i} \; \forall i = 2,\ldots,n \tag{6.50}$$

$$E[\xi_i\xi_j] = 0 \; \forall i \neq j \tag{6.51}$$

That is:

$$\text{Var}[\xi_1] = \frac{1}{2k_1} \tag{6.52}$$

$$\text{Var}[\xi_i] = \frac{1}{2k_2\lambda_i} \; \forall i = 2,\ldots,n \tag{6.53}$$

$$\text{Cov}[\xi_i,\xi_j] = 0 \; \forall i \neq j \tag{6.54}$$

Using this knowledge we can calculate:

$$E[(x_i - x_j)^2] = E[x_i^2 - 2x_ix_j + x_j^2] \tag{6.55}$$

$$= E[x_i^2] - 2E[x_ix_j] + E[x_j^2] \tag{6.56}$$

$$= E[(\mathbf{v}_{i,:}\boldsymbol{\xi})^2] - 2E[(\mathbf{v}_{i,:}\boldsymbol{\xi})(\mathbf{v}_{j,:}\boldsymbol{\xi})] + E[(\mathbf{v}_{j,:}\boldsymbol{\xi})^2] \tag{6.57}$$

where $\mathbf{v}_{i,:}$ represents the i^{th} row of the matrix $\mathbf{V}$. Taking each of these terms in turn:

$$E[(\mathbf{v}_{i,:}\boldsymbol{\xi})^2] = \text{Var}[\sum_{k=1}^{n} v_{i,k}\xi_k] \tag{6.58}$$

$$= \sum_{k=1}^{n} v_{i,k}^2 \text{Var}[\xi_k] \tag{6.59}$$

$$= \left(\frac{1}{\sqrt{n}}\right)^2 \frac{1}{2k_1} + \sum_{k=2}^{n} v_{i,k}^2 \frac{1}{2k_2\lambda_k} \tag{6.60}$$

$$= \frac{1}{2nk_1} + \sum_{k=2}^{n} v_{i,k}^2 \frac{1}{2k_2\lambda_k} \tag{6.61}$$

Note that the initial 'Variance of Sums' to 'Sum of Variances' swap from (6.58) to (6.59) is valid as the variables ξ_k are uncorrelated. Likewise it can be shown that:

$$E[(\mathbf{v}_{j,:}\boldsymbol{\xi})^2] = \frac{1}{2nk_1} + \sum_{k=2}^{n} v_{j,k}^2 \frac{1}{2k_2\lambda_k} \tag{6.62}$$

Therefore, we have:

$$
\begin{aligned}
-2E[(\mathbf{v}_{i,:}\boldsymbol{\xi})(\mathbf{v}_{j,:}\boldsymbol{\xi})] &= -2E\left[\left(\sum_{k=1}^{n} v_{i,k}\xi_k\right)\left(\sum_{k=1}^{n} v_{j,k}\xi_k\right)\right] \tag{6.63}\\
&= -2E\left[\sum_{k=1}^{n}\sum_{l=1}^{n} v_{i,k}v_{j,l}\xi_k\xi_l\right] \tag{6.64}\\
&= -2E\left[\sum_{k=1}^{n} v_{i,k}v_{j,k}\xi_k^2 + \sum_{k\neq l} v_{i,k}v_{j,l}\xi_k\xi_l\right] \tag{6.65}\\
&= -2E\left[\sum_{k=1}^{n} v_{i,k}v_{j,k}\xi_k^2\right] \underbrace{-2E\left[\sum_{k\neq l} v_{i,k}v_{j,l}\xi_k\xi_l\right]}_{=0 \text{ as uncorrelated}} \tag{6.66}\\
&= -2\sum_{k=1}^{n} v_{i,k}v_{j,k}E[\xi_k^2] \tag{6.67}\\
&= -\frac{1}{nk_1} - 2\sum_{k=2}^{n} v_{i,k}v_{j,k}\frac{1}{2k_2\lambda_k} \tag{6.68}
\end{aligned}
$$

Summing these terms together yields:

$$
\begin{aligned}
E[(x_i-x_j)^2] &= \cancel{\frac{1}{2nk_1}} + \sum_{k=2}^{n} v_{i,k}^2\frac{1}{2k_2\lambda_k} - \cancel{\frac{1}{nk_1}} - 2\sum_{k=2}^{n} v_{i,k}v_{j,k}\frac{1}{2k_2\lambda_k} + \cancel{\frac{1}{2nk_1}} + \sum_{k=2}^{n} v_{j,k}^2\frac{1}{2k_2\lambda_k} \tag{6.69}\\
&= \sum_{k=2}^{n}\frac{1}{2k_2\lambda_k}(v_{i,k}^2 - 2v_{i,k}v_{j,k} + v_{j,k}^2) \tag{6.70}\\
&= \sum_{k=2}^{n}\frac{1}{2k_2\lambda_k}(v_{i,k}-v_{j,k})^2 \tag{6.71}\\
&= \frac{1}{2k_2}\sum_{k=2}^{n}\frac{1}{\lambda_k}\frac{\partial\lambda_k}{\partial w_{\{i,j\}}} \tag{6.72}
\end{aligned}
$$

However, if we want to find the partial derivatives of the total effective graph resistance so that we can adapt edge weights using steepest descent, we want to find the quantities:

$$
\begin{aligned}
\frac{\partial\Omega}{\partial w_{\{i,j\}}} &= \frac{\partial}{\partial w_{\{i,j\}}} n\sum_{k=2}^{n}\frac{1}{\lambda_k} \tag{6.73}\\
&= -n\sum_{k=2}^{n}\frac{1}{\lambda_k^2}\frac{\partial\lambda_k}{\partial w_{\{i,j\}}} \tag{6.74}
\end{aligned}
$$

As can be seen, each partial derivative in the sum is scaled incorrectly; in (6.72) we see we have the reciprocals of the Laplacian eigenvalues scaling the partial derivatives, whereas in (6.74) it can be seen that we require the squares of these reciprocals to scale the partial derivatives. All is not lost, however, as we shall find a solution to this problem in Section 6.6. But firstly, we can realise that we have inadvertently found the derivative of another interesting spectral function (up to a constant multiplier):

$$\int \frac{1}{2k_2} \sum_{k=2}^{n} \frac{1}{\lambda_k} \frac{\partial \lambda_k}{\partial w_{\{i,j\}}} dw_{\{i,j\}} = \frac{1}{2k_2} \sum_{k=2}^{n} \ln(\lambda_k) \tag{6.75}$$

$$= \frac{1}{2k_2} \ln\left(\prod_{k=2}^{n} \lambda_k \right) \tag{6.76}$$

$$= \frac{1}{2k_2} \ln\left(\det\left[\overline{\mathbf{L}} \right] \right) \tag{6.77}$$

That is, we have estimated the partial derivative of the log determinant of the reduced Laplacian, see Figure 6.5. This is a concave function of edge weights, and if we maximise the log determinant, we also maximise the determinant of the reduced Laplacian. This is an interesting function thanks to a famous theorem by Kirchoff, which we explore next.

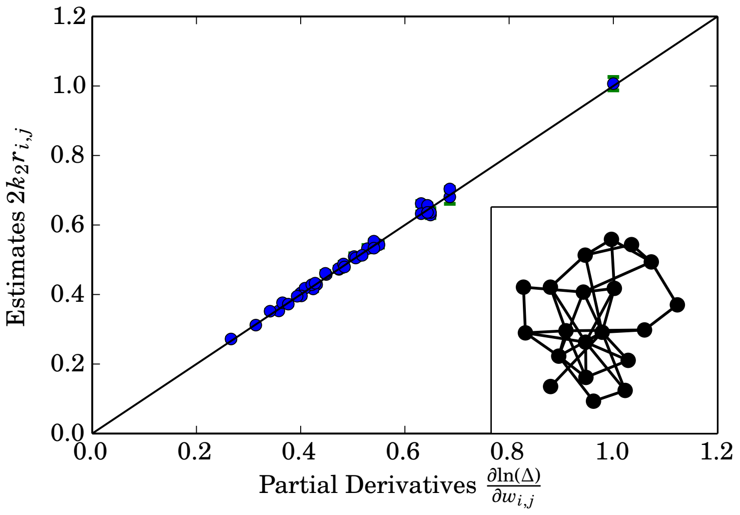


Figure 6.5: Plot comparing the estimates of the partial derivatives of the log determinant of the reduced Laplacian to the true values. Estimates are plotted as the mean of the last 1000 seconds of the simulation run, once estimates have converged, with error bars of 2 standard deviations marked in green.

6.4 Kirchoff's Matrix Tree Theorem

Kirchoff's matrix tree theorem shows that the number of spanning trees in unweighted, undirected graph $\mathscr{G}$ can be deduced from the spectrum of the graph Laplacian matrix $\mathbf{L}$. Specifically, if the eigenvalues of $\mathbf{L}$ are ordered $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, and the set of spanning trees of $\mathscr{G}$ is $\mathscr{T}$ then the number of labelled spanning trees $|\mathscr{T}| = \frac{1}{n}\lambda_2\lambda_3 \ldots \lambda_n$. From this it is evident that if the graph is disconnected, and there are no spanning trees, then $\lambda_2 = 0$. Alternatively, the number of spanning trees is equal to any cofactor of $\mathbf{L}$. It has been suggested in previous literature that the number of spanning trees could be used a metric for network robustness [126, 127] in networks with probabilistic edge losses, because the property of a network being connected is equivalent to the existence of a spanning tree.

Example 6.4. *Consider the simple unweighted graph depicted in Figure 6.6 :*

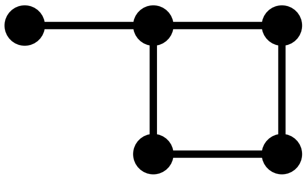


Figure 6.6: A very simple unweighted and undirected graph, $\mathscr{G}$

The graph Laplacian is then:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{6.78}$$

$$= \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & -1 & 0 & -1 & 2 \end{bmatrix} \tag{6.79}$$

which has eigenvalues:

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0.8299\ldots \\ 2 \\ 2.6889\ldots \\ 4.4812\ldots \end{bmatrix} \tag{6.80}$$

Thus, by Kirchoff's matrix tree theorem, the number of spanning trees can be found:

$$|\mathscr{T}| = \frac{1}{n}\lambda_2\lambda_3\lambda_4\lambda_5 = 4 \tag{6.81}$$

We can confirm for this simple example by inspection of the set of spanning trees, $\mathscr{T}$:

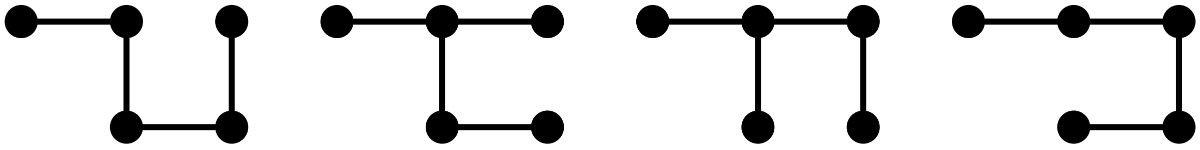


Figure 6.7: The set of spanning trees $\mathscr{T}$ of the graph $\mathscr{G}$

6.4.1 Extension to weighted undirected graphs

When weights are assigned to each edge in the network, the determinant of the reduced Laplacian corresponds to n times the weighted sum of each spanning tree in the network, where the weight of a spanning tree is defined as the product of all edge weights in that tree [130]. As such, the determinant of the reduced Laplacian still provides a useful metric for the robustness of a network, as it quantifies both the number and strength of spanning trees in the network.

$$\det(\overline{\mathbf{L}}) = \lambda_2\lambda_3\ldots\lambda_n \tag{6.82}$$

$$= n \sum_{\mathcal{T}_k\in\mathcal{T}} \prod_{\{i,j\}\in\mathcal{T}_k} w_{\{i,j\}} \tag{6.83}$$

As a particular case, one can consider a weighted graph with integer weights as an unweighted multigraph, where the number of edges between two nodes is given by the weight. Then, the determinant of the reduced Laplacian still determines the number of spanning trees in the multigraph.

Example 6.5. *Consider again the small network used in the previous example, but now let us assign weights to each edge:*

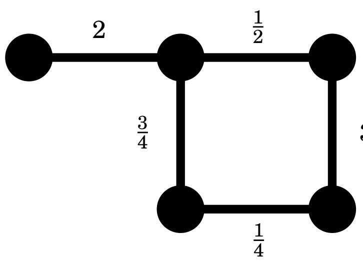


Figure 6.8: A small weighted graph, used in this example.

The set of spanning trees remains the same as in Figure 6.7, but now each tree has a weight defined by the product of edge weights in that tree.

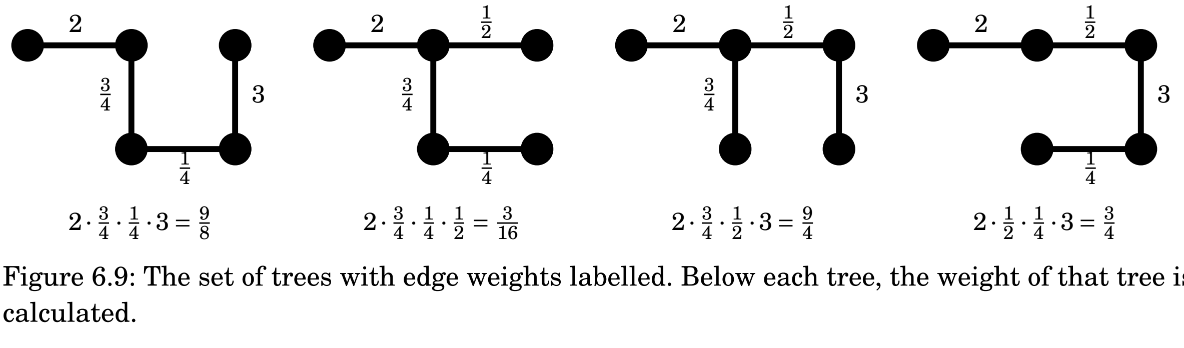

$2\cdot\frac{3}{4}\cdot\frac{1}{4}\cdot 3 = \frac{9}{8}$
$2\cdot\frac{3}{4}\cdot\frac{1}{4}\cdot\frac{1}{2} = \frac{3}{16}$
$2\cdot\frac{3}{4}\cdot\frac{1}{2}\cdot 3 = \frac{9}{4}$
$2\cdot\frac{1}{2}\cdot\frac{1}{4}\cdot 3 = \frac{3}{4}$

Figure 6.9: The set of trees with edge weights labelled. Below each tree, the weight of that tree is calculated.

From the weights of each tree, we can calculate the determinant of the reduced Laplacian in accordance with Equation (6.83)*:* $\det(\overline{\mathbf{L}}) = 5\times(\frac{9}{8}+\frac{3}{16}+\frac{9}{4}+\frac{3}{4}) = \frac{345}{16} = 21.5625$. *To confirm this* *result, we can find the non-trivial eigenvalues of the graph Laplacian and compute the product:*

$$\begin{aligned}\sigma(\overline{\mathbf{L}}) &= \{0.5880\ldots,\ 1.1920\ldots,\ 4.7692\ldots,\ 6.4508\ldots\}\\ \det(\overline{\mathbf{L}}) &= 21.5625\end{aligned}$$

6.5 Optimisation of the Determinant of the Reduced Laplacian

Now that we have a decentralised estimate for the partial derivatives of the log determinant of the reduced Laplacian, these derivative estimates can be substituted into the framework of the existing weight adaptation law. Without an estimate for the value of the of the reduced Laplacian determinant, we are more limited in what optimisation problems can be solved. For example, maximisation of the reduced determinant is readily achieved, but we cannot for instance control the determinant to a specific value, nor constrain another optimisation or control problem by requiring that the reduced determinant be less than, or greater than, a specific value.

Nevertheless, we demonstrate the constrained maximisation of the reduced Laplacian determinant:

$$\begin{aligned}&\underset{\mathbf{w}}{\text{maximise}} && \det(\overline{\mathbf{L}}(\mathbf{w})) &\\ &\text{subject to} && w_{\{i,j\}} \geq 0 && \forall\{i,j\} \in \mathcal{E}\\ &&& \sum_j w_{\{i,j\}} \leq \kappa_i && \forall i \in \mathcal{V}\end{aligned} \tag{6.84}$$

where the constraints are non-negativity of the edge weights, and that the weighted degree of each node may not be greater than a constant value for each node, κ_i. We choose κ_i to be equal to the degree, $\kappa_i = |\mathcal{N}_i|$ so that this constraint is local to each node.

First we notice that as the logarithmic function over the positive reals is a monotonically increasing function, the previous optimisation problem is equivalent to:

$$\begin{aligned}&\underset{\mathbf{w}}{\text{maximise}} && \log\left(\det(\overline{\mathbf{L}}(\mathbf{w})\right) &\\ &\text{subject to} && w_{\{i,j\}} \geq 0 && \forall\{i,j\} \in \mathcal{E}\\ &&& \sum_j w_{\{i,j\}} \leq \kappa_i && \forall i \in \mathcal{V}\end{aligned} \tag{6.85}$$

The log-determinant of a matrix is concave over the set of positive definite matrices, and all constraints are affine so this is a concave maximisation problem.

Applying the decentralised estimator for the partial derivatives of the log determinant of the reduced Laplacian, with the decentralised weight adaptation law, we formulate a stochastic differential equation for the maximisation of the reduced Laplacian determinant:

$$
\begin{aligned}
dx_i &= \left(-k_1 b_i + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(x_j - x_i)\right) dt + dW_i \\
\dot{b_i} &= k_\gamma (x_i - b_i) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) - k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(c_j - c_i) \\
\dot{c_i} &= k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) \\
\dot{r}_{\{i,j\}} &= k_L \left((x_i - x_j)^2 - r_{\{i,j\}}\right) \\
\dot{w}_{\{i,j\}} &= k_w \left(2k_2 r_{\{i,j\}} - \nu_i - \nu_j + \mu_{\{i,j\}}\right) \\
\dot{\mu}_{\{i,j\}} &= -w_{\{i,j\}}\mu_{\{i,j\}} \\
\dot{\nu}_i &= \left(-\kappa_i + \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}\right) \nu_i
\end{aligned}
\tag{6.86}
$$

Example 6.6 (Maximisation of the Determinant of the Reduced Laplacian)**.** *To illustrate the effectiveness of this method, we seek to maximise the determinant of the reduced Laplacian of the random network used in Figure 6.10. Numerical solution of the stochastic differential equations is achieved using the Euler-Maruyama method with a time step of* 1×10^{-3}*, and the control parameters used in this simulation are as follows:* $k_1 = 5$*,* $k_2 = 1$*,* $k_\gamma = 20$*,* $k_P = 40$*,* $k_I = 40$*,* $k_L = 1 \times 10^{-3}$ *and* $k_w = 1 \times 10^{-4}$*. It can be seen in Figure 6.12 that the determinant of the reduced Laplacian (as calculated centrally) rises over time as the edge weights adapt in Figure 6.11. No node knows the current value of the determinant, and each may only communicate with its neighbours. Regardless, the edge weights adapt over time to maximise the determinant of the reduced graph Laplacian, whilst observing the constraint that the weighted degree of each node may not be greater than its degree. This optimisation problem is equivalent to maximising the weighted sum of spanning trees in the network.*

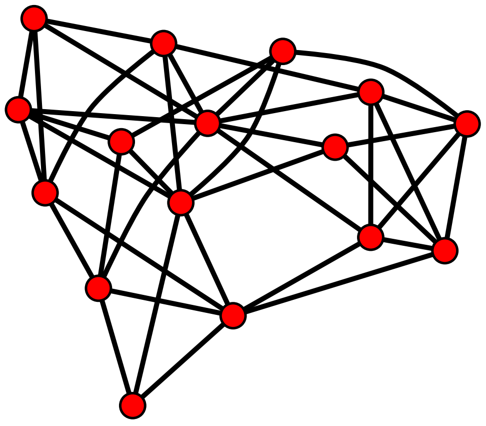


Figure 6.10: A random graph with 16 nodes and 40 edges, used for the reduced Laplacian determinant maximisation example. The original value of the reduced Laplacian determinant with all edge weights of unit value is equal to 16,536,969,984, implying that there are exactly 1,033,560,624 spanning trees in this graph.

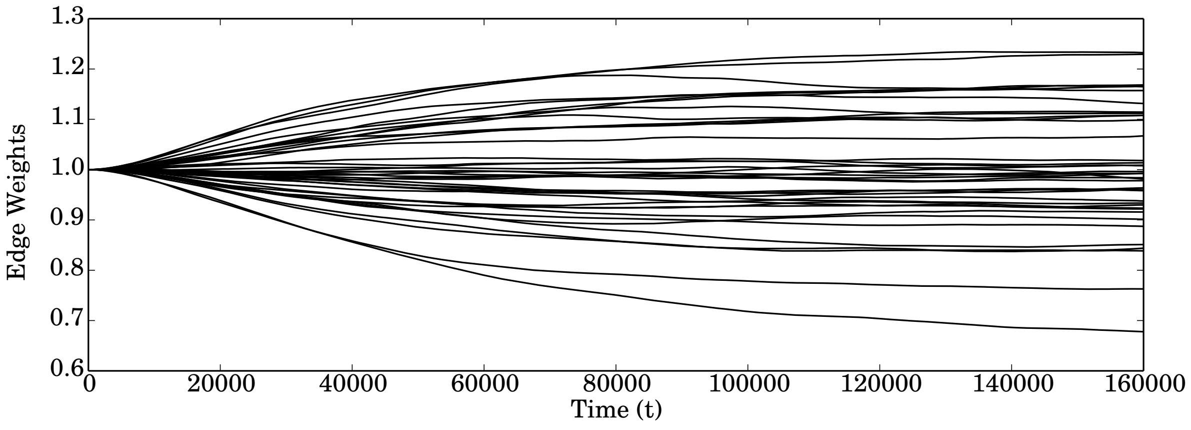


Figure 6.11: Edge weights are adapted in time according to the decentralised adaptive update law as presented in Equation (6.86). Note that due to the constraints and initial conditions, total edge weight in the network does not increase over time.

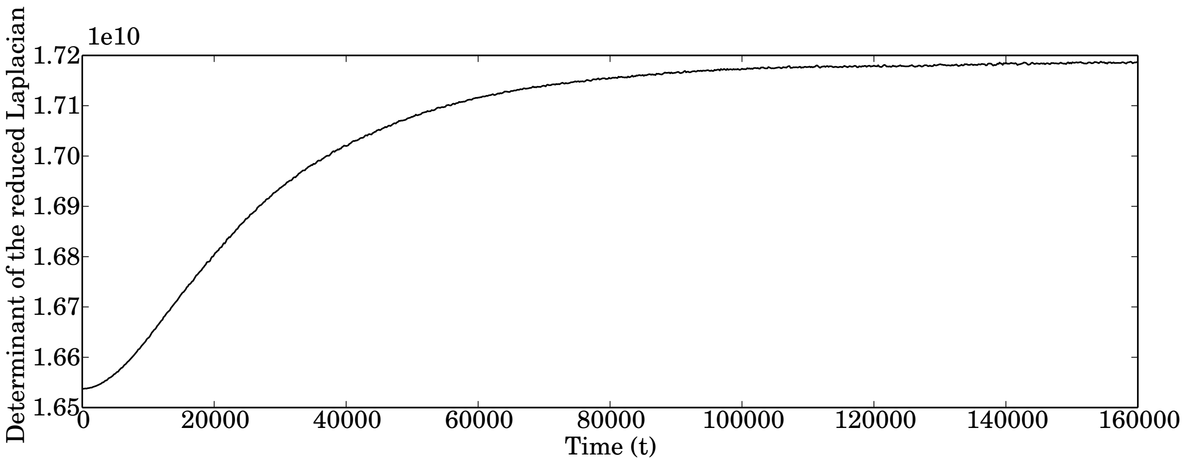


Figure 6.12: As edge weights adapt, the determinant of the reduced Laplacian is suitably improved as we expect.

6.6 Estimating the Partial Derivatives of the Total Effective Graph Resistance

Consider again the partial derivatives of the total effective graph resistance that we wish to estimate:

$$\frac{\partial \Omega}{\partial w_{\{i,j\}}} = \frac{\partial}{\partial w_{\{i,j\}}} n \sum_{k=2}^{n} \frac{1}{\lambda_k} \tag{6.87}$$

$$= -n \sum_{k=2}^{n} \frac{1}{\lambda_k^2} \frac{\partial \lambda_k}{\partial w_{\{i,j\}}} \tag{6.88}$$

Compare this to the result we have obtained for $E[(x_i - x_j)^2]$ in the previous decentralised estimator, (6.72):

$$E[(x_i - x_j)^2] = \frac{1}{2k_2} \sum_{k=2}^{n} \frac{1}{\lambda_k} \frac{\partial \lambda_k}{\partial w_{\{i,j\}}} \tag{6.89}$$

We can see that this result would agree (up to a constant factor) if we used a different matrix in place of the Graph Laplacian in the stochastic differential equation (6.32), with eigenvalues of $\{0, \lambda_2^2, \lambda_3^2, \ldots, \lambda_n^2\}$, and the same eigenvectors as the original Laplacian matrix. Fortunately, such a matrix is easily found in $\mathbf{L}^2$:

$$\begin{aligned} \mathbf{L}^2 &= \mathbf{V\Lambda V}^\top \mathbf{V\Lambda V}^\top \\ &= \mathbf{V\Lambda}^2 \mathbf{V}^\top \end{aligned} \tag{6.90}$$

Unfortunately though, $\mathbf{L}^2$ is not a decentralised matrix, in the sense that for an agent i to compute x_i in the operation $\mathbf{x} = \mathbf{L}^2\mathbf{y}$, not only is information required from neighbours, but also the neighbours of neighbours. However, we can circumnavigate this problem by introducing a fast estimator $\mathbf{f}$ for $\mathbf{Lx}$:

$$d\mathbf{x} = (-k_1\mathbf{b} - k_2\mathbf{Lf})dt + d\mathbf{W} \tag{6.91}$$

$$\dot{\mathbf{b}} = k_\gamma(\mathbf{x} - \mathbf{b}) - k_P\mathbf{Lb} + k_I\mathbf{Lc} \tag{6.92}$$

$$\dot{\mathbf{c}} = -k_I\mathbf{Lb} \tag{6.93}$$

$$\dot{\mathbf{f}} = k_\gamma(\mathbf{Lx} - \mathbf{f}) \tag{6.94}$$

$$\dot{\mathbf{r}} = k_L\left((\mathbf{Px})^{\circ 2} - \mathbf{r}\right) \tag{6.95}$$

Now, the reduced equation for Equation (6.91) is given by:

$$d\mathbf{x} = \left(-k_1\frac{\mathbf{1}\mathbf{1}^\top}{n} - k_2\mathbf{L}^2\right)\mathbf{x}dt + d\mathbf{W} \tag{6.96}$$

The system Equation (6.95) is a first order low pass filter of $(x_i - x_j)^2$ such that we would expect:

$$r_{\{i,j\}} \to E[(x_i - x_j)^2] \to \frac{1}{2k_2} \sum_{k=2}^{n} \frac{1}{\lambda_k^2} \frac{\partial \lambda_k}{\partial w_{\{i,j\}}} = -\frac{1}{2nk_2} \frac{\partial \Omega}{\partial w_{\{i,j\}}} \tag{6.97}$$

Each edge can thus make the estimate:

$$\widetilde{\frac{\partial \Omega}{\partial w_{\{i,j\}}}} = -2nk_2 r_{\{i,j\}} \tag{6.98}$$

and $\widetilde{\frac{\partial \Omega}{\partial w_{\{i,j\}}}} \to \frac{\partial \Omega}{\partial w_{\{i,j\}}}$.

Example 6.7. *To illustrate this result, we again plot the estimates of the partial derivatives against the true values found by direct computation, for a randomly chosen graph, Figure 6.13. The randomly chosen graph is depicted inset in Figure 6.13.*

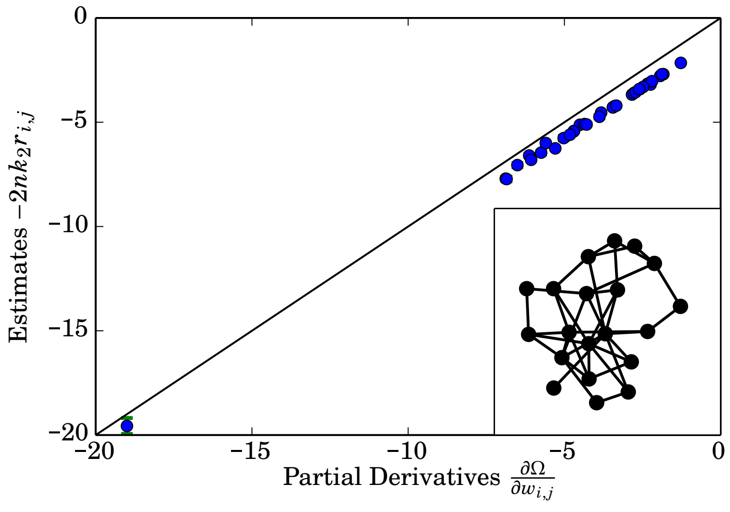


Figure 6.13: Estimates of the partial derivatives of the total effective graph resistance are made using Equations (6.91) to (6.95) and (6.98), and these are plotted against the true values found by direct computation. Note that there is a constant offset present between the estimates and the true values.

It can be seen that there is a constant offset between the estimated gradients and the ground truth. Upon further investigation, it appears that this offset is equal to $\frac{2nk_2}{k_\gamma}$, see Figure 6.14. In the previous analysis, we assumed that k_γ would be large so that this offset would tend to zero. However in simulation, this is not practical, and we may need to relax this assumption to perform a better estimate.

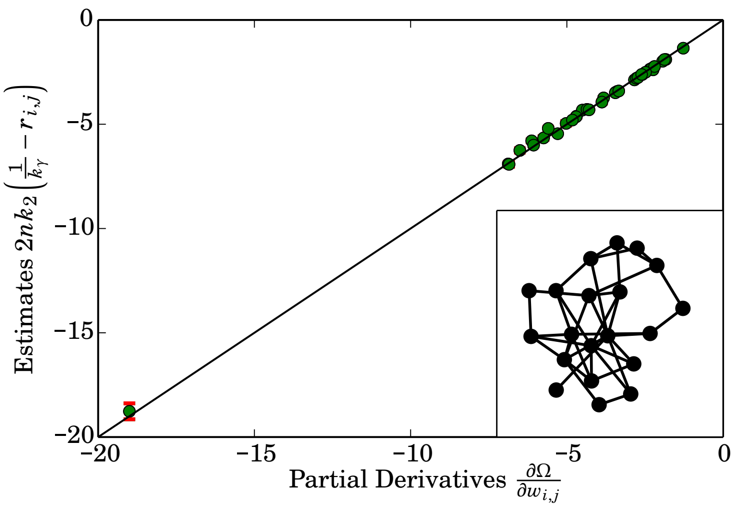


Figure 6.14: Using a modified estimate $\widetilde{\frac{\partial\Omega}{\partial w_{\{i,j\}}}} = 2nk_2\left(\frac{1}{k_\gamma} - r_{\{i,j\}}\right)$, we see that the estimators perform much better. Note that in the derivation we assumed that k_γ was large, so the $\frac{1}{k_\gamma}$ term would be negligible. However, in simulation, it is impractical to make k_γ excessively large, and so further analysis may be required to provide an explanation for the existence of this term.

Finally, we now have a method for making decentralised estimates of the partial derivatives of the total effective graph resistance. This system fits into the existing decentralised weight adaptation law, and can be implemented to solve a number of decentralised optimisation and control problems.

6.7 Optimisation of the Total Effective Graph Resistance

We apply our decentralised edge adaptation algorithm to the problem of minimising the total effective graph resistance of a weighted undirected network, over the edge weights as decision variables. The problem is constrained to an existing graph topology, and admissible edge weights are bounded below by a non-negativity condition, and above by a constraint on the maximum weighted degree of all nodes. As discussed in Section 6.2, solving this problem corresponds to maximising robustness of consensus in the presence of noise, as the bounds given in Equation (6.21) becomes minimal.

Stating this problem in the standard form of a constrained optimisation problem, we have:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimise}} \quad & \Omega(\mathbf{L}(\mathbf{w})) \\ \text{subject to} \quad & w_{\{i,j\}} \geq 0 \qquad \forall \{i,j\} \in \mathscr{E} \\ & \sum_j w_{\{i,j\}} \leq \kappa_i \qquad \forall i \in \mathscr{V} \end{aligned} \tag{6.99}$$

where κ_i is the degree of node i, i.e. $\kappa_i = |\mathscr{N}_i|$.

To solve this optimisation in a decentralised manner, we first implement the following decentralised SDE on the network used in Example 6.7:

$$
\begin{aligned}
dx_i &= \left(-k_1 b_i + k_2 \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(f_j - f_i)\right) dt + dW_i \\
\dot{b_i} &= k_\gamma (x_i - b_i) + k_P \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) - k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(c_j - c_i) \\
\dot{c_i} &= k_I \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(b_j - b_i) \\
\dot{f_i} &= k_\gamma \left(-f_i + \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}(x_i - x_j)\right) \\
\dot{r}_{\{i,j\}} &= k_L \left((x_i - x_j)^2 - r_{\{i,j\}}\right) \\
\dot{w}_{\{i,j\}} &= k_w \left(2nk_2 \left(r_{\{i,j\}} - \frac{1}{k_\gamma}\right) - \nu_i - \nu_j + \mu_{\{i,j\}}\right) \\
\dot{\mu}_{\{i,j\}} &= -w_{\{i,j\}}\mu_{\{i,j\}} \\
\dot{\nu}_i &= \left(-\kappa_i + \sum_{j\in\mathcal{N}_i} w_{\{i,j\}}\right)\nu_i
\end{aligned}
\tag{6.100}
$$

We have chosen to present this SDE from the local view to make it clear that the method is fully decentralised, and to show that 5 variables are updated on each node, and 3 variables are updated on each edge. Note that all W_i are uncorreleated and identical sources of Gaussian white noise with unit variance.

Example 6.8 (Minimisation of the Total Effective Graph Resistance)**.** *Initialising the same network as used in Example 6.6, Figure 6.10, with unit edge weights, $w_{\{i,j\}}(t=0)=1$, so that the boundary conditions are satisfied, we implement the set of SDEs described in* (6.100)*. Control parameters used in this simulation are: $k_1 = 5$, $k_2 = 1$, $k_\gamma = 40$, $k_P = 100$, $k_I = 100$, $k_L = 1\times 10^{-3}$ and $k_w = 4\times 10^{-4}$, and numerical solution is accomplished using the Euler-Maruyama method with a time step of 1×10^{-3}. Edge weights then adapt in time, Figure 6.15, to minimise the total effective graph resistance, Figure 6.16, and the final state of the adapted network is illustrated in Figure 6.17.*

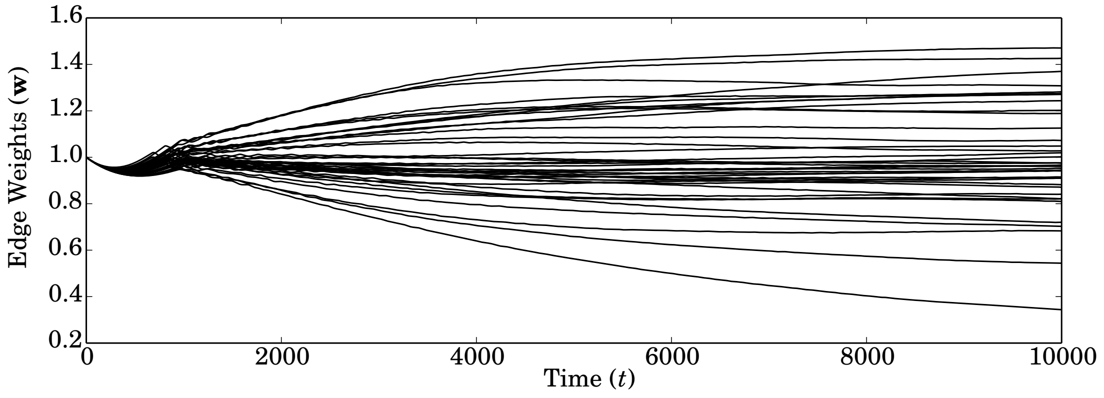


Figure 6.15: Edge weights are adapted in time according to the decentralised adaptive update law (6.100), where the refined sensitivities for the effective resistance, $\widetilde{\frac{\partial \Omega}{\partial w_{\{i,j\}}}} = 2nk_2\left(\frac{1}{k_\gamma} - r_{\{i,j\}}\right)$,have been substituted.

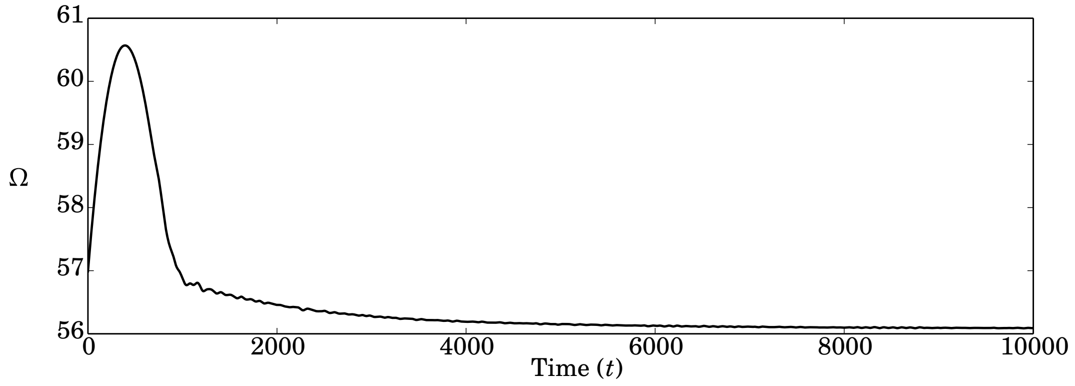


Figure 6.16: As edge weights adapt, the total effective graph resistance Ω is reduced. It can be seen that Ω drops from an initial value of approximately 56.99, to a value that oscillates around ~ 56.09.

It is apparent that graph resistance initially increases in the period before the partial derivative estimates have converged. This problem is exacerbated by the constant offset in the partial derivative estimates of $\frac{2nk_2}{k_\gamma}$*, causing estimates for the partial derivatives to be positive in the early part of the simulation, when it is known that the total effective graph resistance is a strictly decreasing function of edge weight. In practice, this problem could be avoided by delaying the start of the edge adaptation update until the partial derivative estimators have converged sufficiently. Another option to avoid this problem would be simply to threshold the estimates for the partial derivatives, ensuring that they are less than 0: i.e.* $\widetilde{\frac{\partial \Omega}{\partial w_{\{i,j\}}}} = -2nk_2\mathrm{ramp}\left(r_{\{i,j\}} - \frac{1}{k_\gamma}\right)$*, where* ramp() *is*

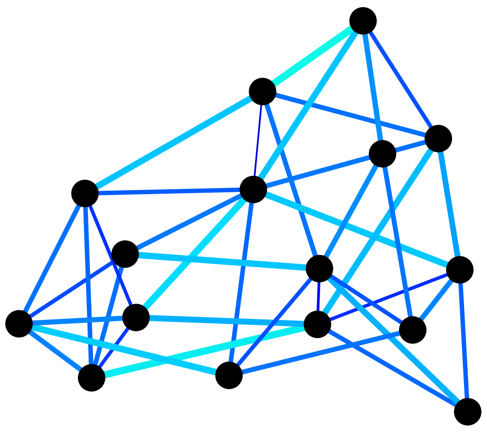


Figure 6.17: The final state of the network, with edge thickness and brightness proportional to the edge weights.

the ramp function:

$$\text{ramp}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \le 0 \end{cases} \tag{6.101}$$

6.8 Summary

In this Chapter, we presented two decentralised estimators so that agents in a network can make estimates of the total effective graph resistance (Equations (6.32) to (6.37)), and the partial derivatives with respect to their incident edge weights (Equations (6.91) to (6.95)). This allows us to minimise the total effective graph resistance of a network, Example 6.8, resulting in lower expected commute times of random walkers on the network and improved robustness to noise in linear consensus applications.

We also found a decentralised method for estimating the partial derivatives of the natural log of the reduced Laplacian determinant (6.72, 6.77) so that we can also develop a decentralised optimisation strategy for maximising the reduced Laplacian determinant, Example 6.6. This also maximises the weighted sum of spanning trees in a network in accordance with Kirchoff's matrix tree theorem.

These two spectral functions of the graph Laplacian matrix are, unlike the algebraic connectivity and spectral radius, functions of all $n-1$ non-trivial eigenvalues. To make these estimates, white noise was introduced into the estimators in order to excite every mode associated with a non-trivial eigenvalue, rather than just reinforcing the effects of the extremal eigenvalues, and resulting in a set of stochastic differential equations. The introduction of noise itself into the estimators then necessitated the use of filtering to reduce the amount of noise in the output estimates, before the estimated functons and derivatives could be used in conjunction with the previously described weight adaptation laws.

Chapter 7

Conclusions

In this Thesis we have explored how a network of interconnected and individual agents can cooperate with their neighbours, exchanging and updating solely local information, so that the collective may make a joint estimate of various global observables of the network as a whole. Then using these decentalised estimates, we have proposed a set of decentralised weight adaptation techniques to achieve constrained optimisation and control of these global observables.

We have focussed specifically on the estimation and optimisation of spectral functions of the graph Laplacian matrix, as these variables can often serve as metrics of the performance of important network applications, such as speed of consensus, synchronisation, and connectivity. Based upon the decentralised algebraic connectivity estimator presented in [29], we have proposed estimators that yield estimates for the spectral radius (and hence the synchronisability ratio may also be estimated), and the effective graph resistance. We have also sought methods for the estimation of the partial derivatives of these functions with respect to edge weights, which are required for optimisation by gradient descent.

In Chapter 3, we reviewed a number of both centralised and decentralised methods for the optimisation and control of spectral functions, paying special attention to the decentralised algebraic connectivity estimator presented in [29], as we develop this method further for more spectral functions in subsequent Chapters. Following from this, in Chapter 4, we propose two decentralised methods for the constrained optimisation of network observables. The first is an interior point method utilising logarithmic barrier functions to ensure feasibility, and the second is a set of ordinary differential equations designed so that the stable equilibrium satisfies the Karush-Kuhn-Tucker (KKT) conditions. Each method has its own advantages and disadvantages. The logarithmic barrier method ensures that at all times edge weights are guaranteed to remain

Spectral Function	Symbol	Constrained Optimisation	Control to a Desired Value
Algebraic Connectivity	λ_2	Chapter 4	Chapter 4
Spectral Radius	λ_n	Chapter 5	Chapter 5
Effective Graph Resistance	Ω	Chapter 6.2	Chapter 6.5
Reduced Graph Laplacian Determinant	Δ	Chapter 6.2.1	✗

Table 7.1: A guide to where the constrained optimisation problems that have been explored in this thesis may be found.

feasible so long as they are initially feasible, but fails if a perturbation forces edge weights into an infeasible regime. Contrariwise, the KKT method remains stable even when edge weights leave the feasible set, but does not guarantee that edge weights will maintain feasibility in the transient to the stable equilibrium. Further to this, the KKT method is proven to be exponentially stable, whereas asymptotic stablity is only ensured for the logarithmic barrier method. Both these methods can be used flexibly to achieve a number of common and relevant graph optimisation problems, so long as the partial derivatives of the objective function and the contstraints can be estimated in an entirely decentralised manner.

In the remaining Chapters, we sought to extend the number of spectral functions of the graph Laplacian that can be estimated using an entirely decentralised approach. In Chapter 5 we modified the algebraic connectivity estimator presented in [29] to estimate the largest eigenvalue of the graph Laplacian matrix, its spectral radius λ_n. This allowed us to optimise both the spectral radius λ_n and the ratio of the algebraic connectivity and the spectral radius, the oft-called synchronisability ratio, thanks to its importance in synchronisation applications. Special attention was placed on proving the exponential stability of the multi-layer system, based upon the separation of time-scales, using singular perturbation theory. Finally in Chapter 6, we modified the decentralised estimator once more to estimate the total effective graph resistance, a function of all non-zero eigenvalues. This was achieved through the addition of white noise to excite all modes with equal energy. When following a similar approach to that used previously to find an estimate for the partial derivative of the effective graph resistance, we instead, surprisingly, found that we had inadvertantly found a decentralised estimate of the partial derivative of the reduced Laplacian determinant. It was found to be necessary once more to modify the estimator for the estimation of the partial derivative of the effective graph resistance. Interestingly, it was not immediately obvious how to recover an estimator for the reduced graph Laplacian from the estimator for its partial derivatives, and we leave this problem to further work. A summary of the decentralised constrained optimisation problems that have been looked at in this Thesis can be seen in Table 7.1.

The most important result from this Thesis is that we have proven that agents in a network are able to, through only local interactions, find globally optimal solutions for a number of common network optimisation problems. There are however a number of caveats. The weight adaptation strategies we have presented rely on a number of layers of successive estimation of global variables. This makes these local strategies far slower than global strategies, as we must wait for estimates to diffuse across the network. Furthermore, convergence of the top layers relies on convergence of all those below, and in cases when the extremal eigenvalues of the optimal network are not distinct, convergence is not guaranteed, and a limit cycle in the vicinity of the optimal network might instead emerge.

7.1 Future Work

There remains a great many further avenues that can be explored. Looking at Table 7.1, we are currently missing a single piece in the control and optimisation of the graph Laplacian spectral functions that have been investigated. Specifically, though we have presented a method for the decentralised estimation of the gradient of the reduced Laplacian determinant in Chapter 6.2.1, we have yet been able to design a method for the value of the reduced Laplacian determinant itself. This means that we are able to perform decentralised constrained maximisation as in Example 6.1, but are currently unable to control the Reduced Determinant to a desired value.

As stated previously, using the current proposed methods, we cannot guarantee exponential convergence onto the solution when the optimal network has non-distinct extremal eigenvalues. A direction for further work would be to devise methods that can circumvent or mitigate this problem, for example by damping the excitations induced by the rapid switching of the equilibrium of the eigenvector estimator when there is not sufficient separation of the eigenvalues.

7.1.1 Discrete Time Stability Analysis

So far, the methods proposed are continuous-time in nature, and the proofs given are also set in the continuous-time framework. However, in practice, numerical integration of the set of ordinary differential equations and stochastic differential equations for the simulations has been achieved using the Runge-Kutta method, Forward Euler method, and Euler-Maruyama method. In digital applications, such as controlling the formation of a swarm of mobile robots, it may prove beneficial to perform a more complete stability and time-scale separation analysis on the difference equations that are suggested by the Forward Euler and Euler-Maruyama methods. This would allow us to determine control gains sufficent to ensure stability for a given time-step, which would significantly lower the number of computations performed before convergence to the vicinity of the solution.

7.1.2 Discrete Edge Weights

An interesting extension would be to move from the realm of optimising networks with continuous edge weights, to those with integer edge weights. This has been achieved for instance in centralised optimisation with the movement from semidefinite programming [77, 76] to mixed-integer semidefinite programming techniques [82, 83]. The paramount in this class of networks are those with 0-1 edge weights, as these are equivalent to unweighted graphs, and this would be a reasonable problem to start with, before generalisation to general integer weights (multigraphs).

A first step towards integer edge weights could be achieved through the sparsification of the continuous edge weight solutions. A simple method to achieve this would be to employ lasso regularisation through the addition of the one norm of the edge weight vector, $\sigma||\mathbf{w}||_1$ (suitably scaled by σ), to the objective function to be minimised. This one-norm, like the spectral function, is also a global function of all the edge-weights and could be estimated using consensus.

Of course some edge weights will still lie in the $(0,1)$ interval. A further method would be required to restrict edge weights to the allowable set of $\{0,1\}$, and if we are interested in finding the globally optimal network, some method to iterate over all feasible combinations would need to be implemented. For example, we could investigate whether a decentralised branch and bound algorithm could be implemented, using the convex relaxation as the bound. If we are instead interested in finding a good (but not necessarily optimal) solution, an initial option for achieving this would to be to use an edge snapping approach similar to that suggested in [131, 132], through modification of the potential function by application of a double potential well on each edge weight.

7.1.3 Generalised Means of the Graph Laplacian Spectrum

It may also be possible to extend the number of spectral functions that can be estimated in a decentralised manner. Though the algebraic connectivity, synchronisability ratio, effective graph resistance, and reduced Laplacian determinant are some of the most frequently studied and useful spectral functions of the graph Laplacian (and this is why we have focussed on these functions in this Thesis) it should be noted that all of these functions are related to each other in the following way: they are all intimately related to generalised means of the set of non-zero eigenvalues of the graph Laplacian matrix.

The generalised mean μ_p of a set of n values x_i is defined in the following manner:

$$\mu_p = \left(\frac{1}{n}\sum_{i=1}^{n} x_i^p\right)^{\frac{1}{p}} \tag{7.1}$$

And specifically there are a number of important generalised means: $\mu_{-\infty}$ is simply the minimum, $\mu_{+\infty}$ is likewise the maximum, μ_{-1} is the harmonic mean, μ_0 is the geometric mean, and μ_1 is the arithmetic mean. When we look at generalised means of the non-zero graph Laplacian eigenvalues $\{\lambda_2,\ldots,\lambda_n\}$ then we find that:

$$\mu_{-\infty} = \min\{\lambda_2, \ldots, \lambda_n\} = \lambda_2 \tag{7.2}$$

$$\mu_{-1} = \frac{n}{\frac{1}{\lambda_2} + \cdots + \frac{1}{\lambda_n}} = \frac{n^2}{\Omega} \tag{7.3}$$

$$\mu_0 = \sqrt[n]{\lambda_2 \ldots \lambda_n} = \sqrt[n]{\Delta} \tag{7.4}$$

$$\mu_1 = \frac{\lambda_2 + \cdots + \lambda_n}{n} = \frac{1}{n}\mathrm{Trace}(\mathbf{L}) = \frac{2\mathbf{1}^\top \mathbf{w}}{n} \tag{7.5}$$

$$\mu_{+\infty} = \max\{\lambda_2, \ldots, \lambda_n\} = \lambda_n \tag{7.6}$$

That is, the spectral functions we have been interested are intimately related to generalised means of the non-zero eigenvalues of the graph Laplacian matrix: the algebraic connectivity and spectral radius are minimum and maximum respectively, the harmonic mean μ_{-1} is inversely proportional to the effective graph resistance, and the geometric mean is the n^{th} root of the reduced graph Laplacian determinant. The arithmetic mean is proportional to the trace of the graph Laplacian; which is the arithmetic mean of the weighted degrees of all nodes; which is related to the the total edge weight in the network. This being a common constraint to use in network optimisation problems.

All generalised means are symmetric functions, and they are either convex or concave, so they will be convex/concave functions of the edge weights. Perhaps it may be possible to generalise the decentralised estimators proposed so as to estimate any generalised mean of the non-zero eigenvalues.

7.1.4 Other Matrices

Another extension would be to shift focus to other important graph matrices, for example the controlling the spectrum of the adjacency and normalised Laplacian matrices [35]. Bounds on the chromatic number of a graph can be determined by the extremal eigenvalues of the adjacency matrix, and it is reasonably straightforward to modify the decentralised estimators to estimate the extremal eigenvalues of the adjacency matrix. For example, to estimate the largest eigenvalue of the adjacency matrix, and its associated eigenvector, one could simply use the following system:

$$\dot{\mathbf{x}} = \kappa_2 \mathbf{A}\mathbf{x} - \kappa_3 \mathbf{x}\mathbf{x}^\top \mathbf{x} \tag{7.7}$$

and the estimator vector $\mathbf{x}$ will align with the eigenvector $\mathbf{v}_n$ associated with the largest eigenvalue of the adjacency matrix α_n, with the magnitude of this vector being related to the magnitude of the eigenvalue in a similar manner to seen with the graph Laplacian estimators. Again the arithmetic mean of the squared components of $\mathbf{x}$ will have to be estimated, with for example a PI consensus subsystem.

The partial derivative of the largest eigenvalue with respect to the edge weights can also be found from its associated eigenvector:

$$\frac{\partial \alpha_n}{\partial w_{\{i,j\}}} = \frac{2v_{n1}v_{n2}}{||\mathbf{v}_n||_2^2} \tag{7.8}$$

The derivation follows the same path as for the graph Laplacian matrix.

BIBLIOGRAPHY

[1] Frede Blaabjerg, Remus Teodorescu, Marco Liserre, and Adrian V Timbus.
"Overview of control and grid synchronization for distributed power generation systems".
In: *IEEE Transactions on Industrial Electronics* 53.5 (2006), pp. 1398–1409.

[2] Florian Dörfler, Michael Chertkov, and Francesco Bullo.
"Synchronization in complex oscillator networks and smart grids".
In: *Proceedings of the National Academy of Sciences* 110.6 (2013), pp. 2005–2010.

[3] Lewis GW Roberts, Alan R Champneys, Keith RW Bell, and Mario di Bernardo.
"Analytical approximations of critical clearing time for parametric analysis of power system transient stability".
In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 5.3 (2015), pp. 465–476.

[4] Réka Albert, Hawoong Jeong, and Albert-Lászlo Barabási.
"Internet: Diameter of the world-wide web".
In: *Nature* 401.6749 (1999), pp. 130–131.

[5] Mark E Crovella and Azer Bestavros.
"Self-similarity in World Wide Web traffic: evidence and possible causes".
In: *IEEE/ACM Transactions on Networking* 5.6 (1997), pp. 835–846.

[6] Wayne W Zachary.
"An information flow model for conflict and fission in small groups".
In: *Journal of Anthropological Research* 33.4 (1977), pp. 452–473.

[7] Raymond T Sparrowe, Robert C Liden, Sandy J Wayne, and Maria L Kraimer.
"Social networks and the performance of individuals and groups".
In: *Academy of Management Journal* 44.2 (2001), pp. 316–325.

[8] Jennifer A Dunne, Richard J Williams, and Neo D Martinez.
"Network structure and biodiversity loss in food webs: robustness increases with connectance".
In: *Ecology Letters* 5.4 (2002), pp. 558–567.

[9] 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.

[10] Olaf Sporns, Giulio Tononi, and Rolf Kötter.
"The human connectome: a structural description of the human brain".
In: *PLoS Comput Biol* 1.4 (2005), e42.

[11] Edward T Bullmore and Danielle S Bassett.
"Brain graphs: graphical models of the human brain connectome".

In: *Annual Review of Clinical Psychology* 7 (2011), pp. 113–140.

[12] Francesc Comellas and Silvia Gago.
"Synchronizability of complex networks".
In: *Journal of Physics A: Mathematical and Theoretical* 40.17 (2007), p. 4483.

[13] Liang Huang, Qingfei Chen, Ying-Cheng Lai, and Louis M Pecora.
"Generic behavior of master-stability functions in coupled nonlinear dynamical systems".
In: *Physical Review E* 80.3 (2009), p. 036204.

[14] Duncan J Watts and Steven H Strogatz.
"Collective dynamics of ‚Äòsmall-world‚Äônetworks".
In: *Nature* 393.6684 (1998), pp. 440–442.

[15] Hyunsuk Hong, Moo-Young Choi, and Beom Jun Kim.
"Synchronization on small-world networks".
In: *Physical Review E* 65.2 (2002), p. 026139.

[16] Dean V Buonomano and Michael M Merzenich.
"Cortical plasticity: from synapses to maps".
In: *Annu. Rev. Neurosci.* 21.1 (1998), pp. 149–186.

[17] Dmitri B Chklovskii, BW Mel, and K Svoboda.
"Cortical rewiring and information storage".
In: *Nature* 431.7010 (2004), pp. 782–788.

[18] Ryutaro Kawamura, K-I Sato, and Ikuo Tokizawa.
"Self-healing ATM networks based on virtual path concept".
In: *IEEE Journal on Selected Areas in Communications* 12.1 (1994), pp. 120–127.

[19] Luca Donetti, Pablo I Hurtado, and Miguel A Munoz.
"Entangled networks, synchronization, and optimal network topology".
In: *Physical Review Letters* 95.18 (2005), p. 188701.

[20] Thilo Gross and Bernd Blasius.
"Adaptive coevolutionary networks: a review".
In: *Journal of the Royal Society Interface* 5.20 (2008), pp. 259–271.

[21] Thomas E Gorochowski, Mario Di Bernardo, and Claire S Grierson.
"Evolving dynamical networks: a formalism for describing complex systems".
In: *Complexity* 17.3 (2012), pp. 18–25.

[22] Dragoslav D Siljak.
Decentralized control of complex systems.
Courier Corporation, 2011.

[23] Herbert A Simon.
"The architecture of complexity".
In: *Proceedings of the American Philosophical Society* 106.6 (1962), pp. 467–482.

[24] Philip W Anderson et al.
"More is different".
In: *Science* 177.4047 (1972), pp. 393–396.

[25] Stuart Kauffman.
At home in the universe: The search for the laws of self-organization and complexity.
Oxford University Press, 1996.

[26] Melanie Mitchell.
Complexity: A guided tour.
Oxford University Press, 2009.

[27] James Ladyman, James Lambert, and Karoline Wiesner.
"What is a complex system?"
In: *European Journal for Philosophy of Science* 3.1 (2013), pp. 33–67.

[28] Yang-Yu Liu, Jean-Jacques Slotine, and Albert-László Barabási.
"Controllability of complex networks".
In: *Nature* 473.7346 (2011), pp. 167–173.

[29] Peng Yang, Randy A Freeman, Geoffrey J Gordon, Kevin M Lynch, Siddhartha S Srinivasa, and Rahul Sukthankar.
"Decentralized estimation and control of graph connectivity for mobile sensor networks".
In: *Automatica* 46.2 (2010), pp. 390–396.

[30] Alexander Bertrand and Marc Moonen.
"Distributed computation of the Fiedler vector with application to topology inference in ad hoc networks".
In: *Signal Processing* 93.5 (2013), pp. 1106–1117.

[31] Louis Kempton, Guido Herrmann, and Mario di Bernardo.
"Adaptive weight selection for optimal consensus performance".
In: *53rd Annual Conference on Decision and Control (CDC)*.
IEEE. 2014,
Pp. 2234–2239.
DOI: `10.1109/CDC.2014.7039730`.

[32] Louis Kempton, Guido Herrmann, and Mario di Bernardo.
"Distributed optimisation and control of graph Laplacian eigenvalues for robust consensus via an adaptive multi-layer strategy".
In: *International Journal of Robust and Nonlinear Control* 27.9 (2017), pp. 1499–1525.
DOI: `10.1002/rnc.3808`.

[33] Louis Kempton, Guido Herrmann, and Mario di Bernardo.
"Distributed adaptive optimization and control of network structures".
In: *55th Annual Conference on Decision and Control (CDC)*.
IEEE. 2016,
Pp. 5839–5844.
DOI: `10.1109/CDC.2016.7799167`.

[34] Louis Kempton, Guido Herrmann, and Mario di Bernardo.
"Self-organization of weighted networks for optimal synchronizability".
In: *Transactions on Control of Network Systems* (2017). (Accepted).
DOI: `10.1109/TCNS.2017.2732161`.

[35] Fan RK Chung.
Spectral graph theory.
Number 92 in CBMS Regional Conference Series in Mathematics.
American Mathematical Soc., 1997.

[36] Andries E Brouwer and Willem H Haemers.

Spectra of graphs.
Springer Science & Business Media, 2011.

[37] Zoran Stanić.
Inequalities for graph eigenvalues.
Number 423 in London Mathematical Society Lecture Note Series.
Cambridge University Press, 2015.

[38] Claude Shannon.
"The zero error capacity of a noisy channel".
In: *IRE Transactions on Information Theory* 2.3 (1956), pp. 8–19.

[39] Dénes König.
Theory of Finite and Infinite Graphs.
Springer, 1990.

[40] Bojan Mohar, Y Alavi, G Chartrand, and OR Oellermann.
"The Laplacian spectrum of graphs".
In: *Graph Theory, Combinatorics, and Applications* 2.871-898 (1991), p. 12.

[41] Nair Maria Maia De Abreu.
"Old and new results on algebraic connectivity of graphs".
In: *Linear Algebra and Its Applications* 423.1 (2007), pp. 53–73.

[42] Miroslav Fiedler.
"Algebraic connectivity of graphs".
In: *Czechoslovak Mathematical Journal* 23.2 (1973), pp. 298–305.

[43] Louis M Pecora and Thomas L Carroll.
"Master stability functions for synchronized coupled systems".
In: *Physical Review Letters* 80.10 (1998), p. 2109.

[44] Mauricio Barahona and Louis M Pecora.
"Synchronization in small-world systems".
In: *Physical Review Letters* 89.5 (2002), p. 054101.

[45] Reza Olfati-Saber and Richard M Murray.
"Consensus problems in networks of agents with switching topology and time-delays".
In: *IEEE Transactions on Automatic Control* 49.9 (2004), pp. 1520–1533.

[46] Wenwu Yu, Pietro DeLellis, Guanrong Chen, Mario Di Bernardo, and Jürgen Kurths.
"Distributed adaptive control of synchronization in complex networks".
In: *IEEE Transactions on Automatic Control* 57.8 (2012), pp. 2153–2158.

[47] Jonathan Richard Shewchuk.
"Allow Me to Introduce Spectral and Isoperimetric Graph Partitioning".
In: (2016). http://people.eecs.berkeley.edu/ jrs/papers/partnotes.pdf.

[48] Inderjit S Dhillon.
"Co-clustering documents and words using bipartite spectral graph partitioning".
In: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
ACM. 2001,
Pp. 269–274.

[49] Bojan Mohar and Svatopluk Poljak.
"Eigenvalues and the max-cut problem".
In: *Czechoslovak Mathematical Journal* 40.2 (1990), pp. 343–352.

[50] Ali Jadbabaie, Jie Lin, and A Stephen Morse.
"Coordination of groups of mobile autonomous agents using nearest neighbor rules".
In: *IEEE Transactions on Automatic Control* 48.6 (2003), pp. 988–1001.

[51] Wei Ren and Randal W Beard.
"Consensus seeking in multiagent systems under dynamically changing interaction topologies".
In: *IEEE Transactions on Automatic Control* 50.5 (2005), pp. 655–661.

[52] 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.

[53] Wenwu Yu, Guanrong Chen, and Ming Cao.
"Consensus in directed networks of agents with nonlinear dynamics".
In: *IEEE Transactions on Automatic Control* 56.6 (2011), pp. 1436–1441.

[54] Lin Xiao and Stephen Boyd.
"Fast linear iterations for distributed averaging".
In: *Systems & Control Letters* 53.1 (2004), pp. 65–78.

[55] Bharath Sundararaman, Ugo Buy, and Ajay D Kshemkalyani.
"Clock synchronization for wireless sensor networks: a survey".
In: *Ad Hoc Networks* 3.3 (2005), pp. 281–323.

[56] Jianping He, Peng Cheng, Ling Shi, and Jiming Chen.
"Sats: Secure average-consensus-based time synchronization in wireless sensor networks".
In: *IEEE Transactions on Signal Processing* 61.24 (2013), pp. 6387–6400.

[57] Dimitri P Bertsekas and John N Tsitsiklis.
Parallel and distributed computation: numerical methods.
Vol. 23.
Prentice hall Englewood Cliffs, NJ, 1989.

[58] Chengcheng Zhao, Jianping He, Peng Cheng, and Jiming Chen.
"Consensus-based energy management in smart grid with transmission losses and directed communication".
In: *IEEE Transactions on Smart Grid* PP.99 (2016), pp. 1–13.

[59] Vignesh Kumar and Ferat Sahin.
"Cognitive maps in swarm robots for the mine detection application".
In: *IEEE International Conference on Systems, Man and Cybernetics*.
Vol. 4.
IEEE. 2003,
Pp. 3364–3369.

[60] Cai Luo, Andre Possani Espinosa, Danu Pranantha, and Alessandro De Gloria.
"Multi-robot search and rescue team".
In: *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*.
IEEE. 2011,

Pp. 296–301.

[61] Guoxian Zhang, Gregory K Fricke, and Devendra P Garg.
"Spill detection and perimeter surveillance via distributed swarming agents".
In: *IEEE/ASME Transactions on Mechatronics* 18.1 (2013), pp. 121–129.

[62] Luca Scardovi and Rodolphe Sepulchre.
"Synchronization in networks of identical linear systems".
In: *Automatica* 45.11 (2009), pp. 2557–2562.

[63] Zhongkui Li, Zhisheng Duan, Guanrong Chen, and Lin Huang.
"Consensus of multiagent systems and synchronization of complex networks: A unified viewpoint".
In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 57.1 (2010), pp. 213–224.

[64] Pietro DeLellis, Mario di Bernardo, and Giovanni Russo.
"On QUAD, Lipschitz, and contracting vector fields for consensus and synchronization of networks".
In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 58.3 (2011), pp. 576–583.

[65] Louis M Pecora and Thomas L Carroll.
"Synchronization in chaotic systems".
In: *Physical Review Letters* 64.8 (1990), p. 821.

[66] Daniel A Spielman.
"Spectral graph theory and its applications".
In: *Foundations of Computer Science, 48th Annual IEEE Symposium on.*
IEEE. 2007,
Pp. 29–38.

[67] Leo Grady and Eric L Schwartz.
"Isoperimetric graph partitioning for image segmentation".
In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.3 (2006), pp. 469–475.

[68] Svatopluk Poljak and Franz Rendl.
"Solving the max-cut problem using eigenvalues".
In: *Discrete Applied Mathematics* 62.1-3 (1995), pp. 249–278.

[69] Tiberiu Rotaru and Hans-Heinrich Nägeli.
"Dynamic load balancing by diffusion in heterogeneous systems".
In: *Journal of Parallel and Distributed Computing* 64.4 (2004), pp. 481–497.

[70] William Thomas Tutte.
"On the algebraic theory of graph colorings".
In: *Journal of Combinatorial Theory* 1.1 (1966), pp. 15–50.

[71] Dragoš Cvetković, Mirjana Čangalović, and Vera Kovačević-Vujčić.
"Semidefinite programming methods for the symmetric traveling salesman problem".
In: *International Conference on Integer Programming and Combinatorial Optimization.*
Springer. 1999,
Pp. 126–136.

[72] Bojan Mohar.
"Graph laplacians".

In: *Topics in Algebraic Graph Theory*.
Vol. 102.
Cambridge University Press Cambridge, 2004,
Pp. 113–136.

[73] Luca Donetti, Franco Neri, and Miguel A Munoz.
"Optimal network topologies: expanders, cages, Ramanujan graphs, entangled networks and all that".
In: *Journal of Statistical Mechanics: Theory and Experiment* 2006.08 (2006), P08007.

[74] Dragoš Cvetković, Pierre Hansen, and Vera Kovačević-Vujčić.
"On some interconnections between combinatorial optimization and extremal graph theory".
In: *Yugoslav Journal of Operations Research* 14.2 (2004), pp. 147–154.

[75] Miroslav Fiedler.
"Absolute algebraic connectivity of trees".
In: *Linear and Multilinear Algebra* 26.1-2 (1990), pp. 85–106.

[76] Arpita Ghosh, Stephen Boyd, and Amin Saberi.
"Minimizing effective resistance of a graph".
In: *SIAM Review* 50.1 (2008), pp. 37–66.

[77] Stephen Boyd.
"Convex optimization of Graph Laplacian eigenvalues".
In: *Proceedings of the International Congress of Mathematicians* 3.1-3 (2006), pp. 1311–1319.

[78] Arpita Ghosh and Stephen Boyd.
"Upper bounds on algebraic connectivity via convex optimization".
In: *Linear Algebra and its Applications* 418.2-3 (2006), pp. 693–707.

[79] Yoonsoo Kim and Mehran Mesbahi.
"On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian".
In: *IEEE Transactions on Automatic Control* 51.1 (2006), pp. 116–120.

[80] Arpita Ghosh and Stephen Boyd.
"Growing well-connected graphs".
In: *45th Annual Conference on Decision and Control (CDC)*.
IEEE. 2006,
Pp. 6605–6611.

[81] Damon Mosk-Aoyama.
"Maximum algebraic connectivity augmentation is NP-hard".
In: *Operations Research Letters* 36.6 (2008), pp. 677–679.

[82] Ran Dai and Mehran Mesbahi.
"Optimal topology design for dynamic networks".
In: *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*.
IEEE. 2011,
Pp. 1280–1285.

[83] Mohammad Rafiee and Alexandre M Bayen.

"Optimal network topology design in multi-agent systems for efficient average consensus". In: *49th IEEE Conference on Decision and Control (CDC).* IEEE. 2010, Pp. 3877–3883.

[84] Ali Sydney, Caterina Scoglio, and Don Gruenbacher. "Optimizing algebraic connectivity by edge rewiring". In: *Applied Mathematics and Computation* 219.10 (2013), pp. 5465–5479.

[85] Xiangrong Wang, Evangelos Pournaras, Robert E Kooij, and Piet Van Mieghem. "Improving robustness of complex networks via the effective graph resistance". In: *The European Physical Journal B* 87.9 (2014), p. 221.

[86] Mohammed JF Alenazi, Egemen K Cetinkaya, and James PG Sterbenz. "Network design and optimisation based on cost and algebraic connectivity". In: *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2013 5th International Congress on.* IEEE. 2013, Pp. 193–200.

[87] Huijuan Wang and Piet Van Mieghem. "Algebraic connectivity optimization via link addition". In: *Proceedings of the 3rd International Conference on Bio-Inspired Models of Network, Information and Computing Sytems.* ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). 2008, P. 22.

[88] Ming Cao and Chai Wah Wu. "Topology design for fast convergence of network consensus algorithms". In: *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on.* IEEE. 2007, Pp. 1029–1032.

[89] Andrea Simonetto, Tamás Keviczky, and Robert Babuška. "On distributed maximization of algebraic connectivity in robotic networks". In: *American Control Conference (ACC), 2011.* IEEE. 2011, Pp. 2180–2185.

[90] Michael M Zavlanos, Ali Jadbabaie, and George J Pappas. "Flocking while preserving network connectivity". In: *46th IEEE Conference on Decision and Control.* IEEE. 2007, Pp. 2919–2924.

[91] Michael M Zavlanos and George J Pappas. "Distributed connectivity control of mobile networks". In: *IEEE Transactions on Robotics* 24.6 (2008), pp. 1416–1428.

[92] Michael M Zavlanos, Magnus B Egerstedt, and George J Pappas. "Graph-theoretic connectivity control of mobile robot networks". In: *Proceedings of the IEEE* 99.9 (2011), pp. 1525–1540.

[93] Dimos V Dimarogonas and Karl H Johansson.
"Decentralized connectivity maintenance in mobile networks with bounded inputs".
In: *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*.
IEEE. 2008,
Pp. 1507–1512.

[94] Dimos V Dimarogonas and Karl H Johansson.
"Bounded control of network connectivity in multi-agent systems".
In: *IET Control Theory & Applications* 4.8 (2010), pp. 1330–1338.

[95] Maria Carmela De Gennaro and Ali Jadbabaie.
"Decentralized control of connectivity for multi-agent systems".
In: *45th IEEE Conference on Decision and Control*.
IEEE. 2006,
Pp. 3628–3633.

[96] David Kempe and Frank McSherry.
"A decentralized algorithm for spectral analysis".
In: *Journal of Computer and System Sciences* 74.1 (2008), pp. 70–83.

[97] Mauro Franceschelli, Andrea Gasparri, Alessandro Giua, and Carla Seatzu.
"Decentralized estimation of Laplacian eigenvalues in multi-agent systems".
In: *Automatica* 49.4 (2013), pp. 1031–1036.

[98] Alexander Bertrand and Marc Moonen.
"Topology-aware distributed adaptation of Laplacian weights for in-network averaging".
In: *21st European Signal Processing Conference (EUSIPCO 2013)*.
IEEE. 2013,
Pp. 1–5.

[99] Xiaoli Li, Shanwei Zhang, and Yugeng Xi.
"Connected flocking of multi-agent system based on distributed eigenvalue estimation".
In: *30th Chinese Control Conference (CCC)*.
IEEE. 2011,
Pp. 6061–6066.

[100] Randy A Freeman, Peng Yang, and Kevin M Lynch.
"Stability and convergence properties of dynamic average consensus estimators".
In: *45th IEEE Conference on Decision and Control*.
IEEE. 2006,
Pp. 338–343.

[101] SJ Kirkland and M Neumann.
"On algebraic connectivity as a function of an edge weight".
In: *Linear and Multilinear Algebra* 52.1 (2004), pp. 17–33.

[102] Alexander Bertrand and Marc Moonen.
"Seeing the bigger picture: How nodes can learn their place within a complex ad hoc network topology".
In: *IEEE Signal Processing Magazine* 30.3 (2013), pp. 71–82.

[103] Lieven Vandenberghe and Stephen Boyd.
"Semidefinite programming".
In: *SIAM Review* 38.1 (1996), pp. 49–95.

[104] Stephen Boyd, Laurent El-Ghaoui, Eric Feron, and Venkataramanan Balakrishnan.
Linear matrix inequalities in system and control theory.
SIAM, 1994.

[105] Stephen Boyd and Lieven Vandenberghe.
Convex optimization.
Cambridge university press, 2004.

[106] Reha H Tütüncü, Kim-Chuan Toh, and Michael J Todd.
"Solving semidefinite-quadratic-linear programs using SDPT3".
In: *Mathematical Programming* 95.2 (2003), pp. 189–217.

[107] Johan Lofberg.
"YALMIP: A toolbox for modeling and optimization in MATLAB".
In: *IEEE International Symposium on Computer Aided Control Systems Design.*
IEEE. 2004,
Pp. 284–289.

[108] Nocedal Jorge and Stephen J Wright.
Numerical optimization.
Springer-Verlag New York, 1999.
Chap. 17.

[109] H. W. Kuhn and A. W. Tucker.
"Nonlinear programming".
In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability.*
University of California Press, 1951,
Pp. 481–492.

[110] Dimitri P Bertsekas.
Nonlinear programming.
Athena Scientific Belmont, 1999.

[111] Heinz H. Bauschke and Patrick L. Combettes.
"Fermat's Rule in Convex Optimization".
In: *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.*
CMS Books in Mathematics.
Springer New York, 2011,
Pp. 381–397.

[112] Boris T Polyak.
"Some methods of speeding up the convergence of iteration methods".
In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), pp. 1–17.

[113] Hedy Attouch, Xavier Goudou, and Patrick Redont.
"The heavy ball with friction method, I. The continuous dynamical system: global exploration of the local minima of a real-valued function by asymptotic analysis of a dissipative dynamical system".
In: *Communications in Contemporary Mathematics* 2.01 (2000), pp. 1–34.

[114] Euhanna Ghadimi, Hamid Reza Feyzmahdavian, and Mikael Johansson.
"Global convergence of the heavy-ball method for convex optimization".

In: *European Control Conference (ECC)*.
IEEE. 2015,
Pp. 310–315.

[115] Gabriel Goh.
"Why momentum really works".
In: *Distill* (2017).
URL: `http://distill.pub/2017/momentum`.

[116] J LaSalle.
"Some extensions of Liapunov's second method".
In: *Circuit Theory, IRE Transactions on* 7.4 (1960), pp. 520–527.

[117] Petar Kokotovic, Hassan K Khalil, and John O'reilly.
Singular perturbation methods in control: analysis and design.
SIAM, 1986.

[118] H. K. Khalil.
Nonlinear Systems.
2nd Edition.
Englewood Cliffs, NJ: Prentice-Hall, 1996.

[119] Frank Hoppensteadt.
"On systems of ordinary differential equations with several parameters multiplying the derivatives".
In: *Journal of Differential Equations* 5.1 (1969), pp. 106–116.

[120] John Guckenheimer and Philip Holmes.
Nonlinear oscillations, dynamical systems, and bifurcations of vector fields.
Vol. 42.
Springer Science & Business Media, 2013.

[121] Peter D Lax.
Linear Algebra and Its Applications, 2nd ed.
Wiley-Interscience, New York, 2007.

[122] Douglas J Klein and Milan Randić.
"Resistance distance".
In: *Journal of Mathematical Chemistry* 12.1 (1993), pp. 81–95.

[123] Prasad Tetali.
"Random walks and the effective resistance of networks".
In: *Journal of Theoretical Probability* 4.1 (1991), pp. 101–109.

[124] Ashok K Chandra, Prabhakar Raghavan, Walter L Ruzzo, Roman Smolensky, and Prasoon Tiwari.
"The electrical resistance of a graph captures its commute and cover times".
In: *Computational Complexity* 6.4 (1996), pp. 312–340.

[125] Seth Chaiken and Daniel J Kleitman.
"Matrix tree theorems".
In: *Journal of Combinatorial Theory, Series A* 24.3 (1978), pp. 377–381.

[126] John S Baras and Pedram Hovareshti.

"Efficient and robust communication topologies for distributed decision making in networked systems".
In: *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*.
IEEE. 2009,
Pp. 3751–3756.

[127] Wendy Ellens and Robert E Kooij.
"Graph measures and network robustness".
In: *arXiv preprint arXiv:1311.5064* (2013).

[128] W Ellens, FM Spieksma, P Van Mieghem, A Jamakovic, and RE Kooij.
"Effective graph resistance".
In: *Linear Algebra and Its Applications* 435.10 (2011), pp. 2491–2506.

[129] George Forrest Young, Luca Scardovi, and Naomi Ehrich Leonard.
"Robustness of noisy consensus dynamics with directed communication".
In: *Proceedings of the 2010 American Control Conference*.
IEEE. 2010,
Pp. 6312–6317.

[130] Paul Abraham Fuhrmann and Uwe Helmke.
The mathematics of networks of linear systems.
Springer, 2015.

[131] Pietro DeLellis, Franco di Bernardo Mario Garofalo, and Maurizio Porfiri.
"Evolution of complex networks via edge snapping".
In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 57.8 (2010), pp. 2132–2143.

[132] Pietro DeLellis, Mario di Bernardo, and Maurizio Porfiri.
"Pinning control of complex networks via edge snapping".
In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 21.3 (2011), p. 033119.