UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DISTRIBUTED COMPUTATION OF GRAPH SPECTRUM, EIGENVECTOR CENTRALITY, AND SOLUTION TO LINEAR EQUATIONS

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

MU YANG
Norman, Oklahoma
2017

DISTRIBUTED COMPUTATION OF GRAPH SPECTRUM,
EIGENVECTOR CENTRALITY,
AND SOLUTION TO LINEAR EQUATIONS

A DISSERTATION APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

To my family

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor, Dr. Choon Yik Tang, for providing tremendous mentoring and support throughout my Ph.D. studies. He gave me freedom to pursue my own research interests and at the same time provided me with valuable feedback and advice. Besides research, he also gave me insightful suggestions on my career and life.

I am grateful to Dr. J. R. Cruz, Dr. S. Lakshmivarahan, Dr. Thordur Runolfsson, and Dr. Krishnaiyan Thulasiraman for their interest in my research and for serving on my dissertation committee. I have greatly benefited from their involvement and recommendations.

I would like to thank my family. My wife, Jingjue Yi, has been extremely supportive of me throughout this entire process and has made countless sacrifices to help me get to this point. My daughter, Harper, has continually brought me happiness. My parents, Chengfeng Yang and Yafei Lin, always have faith in me and give me liberty to choose what I desired. Special thanks to my father- and mother-in-law, Yong Yi and Huiping Liu, who came to Norman twice and helped us take care of Harper.

Finally, financial support from the National Science Foundation is gratefully acknowledged.

Table of Contents

Acknowledgements **iv**

List of Figures **vii**

Abstract **viii**

Chapter 1. Introduction **1**

1.1 Background and Motivation 1

1.2 Literature Review 2

1.3 Original Contributions 4

1.4 Dissertation Outline 6

Chapter 2. Distributed Estimation of Graph Spectrum **7**

2.1 Introduction 7

2.2 Problem Formulation 9

2.3 Forming a Set of Linear Equations 11

2.4 Solving the Linear Equations 16

2.4.1 Scenario 1: Cyclic Case 16

2.4.2 Scenario 2: Acyclic Case 21

2.5 Simulation Results 24

2.5.1 Simulation of Algorithm 2.1 for Scenario 1 24

2.5.2 Simulation of Algorithm 2.2 for Scenario 2 26

2.6 Conclusion 28

Chapter 3. A Distributed Algorithm for Solving General Linear Equations **29**

3.1 Introduction 29

3.2 Problem Formulation 31

3.3 Algorithm Design 32

3.4 Algorithm Analysis 34
3.5 Convergence Rate Analysis 42
3.5.1 Proof of Theorem 3.4 45
3.5.2 Why Convergence may be Slow 52
3.6 Simulation Results 53
3.6.1 Two 15-Node Graphs 54
3.6.2 A 5-Node Graph 55
3.7 Conclusion 55

Chapter 4. Continuous-Time Distributed Computation of the Perron-Frobenius Eigenvector 62

4.1 Introduction 62
4.2 Problem Formulation 64
4.3 Design of Continuous-Time Algorithms 67
4.4 Analysis of Continuous-Time Algorithms 71
4.4.1 Convergence Analysis 71
4.4.2 Convergence Rate 77
4.4.3 Special Cases 88
4.5 Simulation Results 90
4.6 Conclusion 90

Chapter 5. Asynchronous Gossip Computation of the Perron-Frobenius Eigenvector 92

5.1 Introduction 92
5.2 Problem Formulation 93
5.3 Design of Gossip Algorithm 94
5.4 Analysis of Gossip Algorithm 99
5.5 Simulation Results 110
5.6 Conclusion 110

Chapter 6. Conclusions 113

6.1 Overall Summary 113
6.2 Future Work 115

Bibliography 118

List of Figures

2.1 Distribution of the eigenvalues of a graph's adjacency matrix, represented by red crosses, offers insights into the shapes and sizes of communities in the graph. 8

2.2 Flowchart illustrating our approach to estimating graph spectrum. 10

2.3 Performance of Algorithm 2.1 for Scenario 1. 25

2.4 Performance of Algorithm 2.2 for Scenario 2. 27

3.1 Illustration of the idea behind algorithm (3.5). 33

3.2 Contour plot showing how the lower bound λ^* on the convergence rate λ_1 of algorithm (3.7) depends on the smallest singular value σ of $\hat{A}$ and the algebraic connectivity $\lambda_{\min}(L)$ of $\mathcal{G}$, when $(N, \alpha, \beta) = (100, 100, 1)$. 45

3.3 A simple example explaining why algorithm (3.7) may converge slowly when A is nearly singular. 53

3.4 Performance of algorithm (3.6) when there is a unique solution. 57

3.5 Performance of algorithm (3.6) when there is no solution. 58

3.6 Performance of algorithm (3.6) when there is no solution. 59

3.7 Performance of algorithm (3.6) when there is a unique solution. 60

3.8 Performance of algorithm (3.6) when there are infinitely many solutions. 61

4.1 Examples illustrating the concept of eigenvector centrality. . . . 63

4.2 Vector field for a 3-node path graph. 73

4.3 Performance of continuous-time algorithm (4.12). 91

5.1 Performance of gossip algorithm. 112

Abstract

DISTRIBUTED COMPUTATION OF GRAPH SPECTRUM, EIGENVECTOR CENTRALITY, AND SOLUTION TO LINEAR EQUATIONS

Mu Yang, Ph.D.
The University of Oklahoma, 2017

Supervisor: Choon Yik Tang

This dissertation is devoted to the development of distributed algorithms, with which nodes in a large decentralized network can accomplish tasks that are seemingly difficult without an omniscient central node. The tasks include estimating the graph spectrum, from which each node can draw its own conclusion about the network structure, computing the eigenvector centrality, from which every node can judge its own importance in the network, and solving a system of linear equations whose data are scattered across the network or discovering that no solution exists. The ability to perform these tasks enhances the capability of existing and emerging networks such as smart power grids, social networks, and ad hoc sensor networks, potentially allowing them to function in ways that are not previously thought to be possible.

We begin with the design of a novel, two-stage distributed algorithm that enables nodes in an undirected and connected graph to jointly estimate the spectrum of a matrix associated with the graph, which includes the adjacency and Laplacian matrices as special cases. In the first stage, the algorithm uses a discrete-time linear iteration and the Cayley-Hamilton theorem to convert the problem into one of solving linear equations, where each equation is known to a node. In the second stage, if the nodes happen to know that said matrix is cyclic, the algorithm uses a Lyapunov approach to asymptotically solve the equations with an exponential rate of convergence. Otherwise, it uses a random perturbation approach and a structural controllability result to approximately solve the equations with an error that can be made small.

We then consider the fundamental problem of cooperatively solving a general system of linear equations over a network, for which a continuous-time distributed algorithm is devised. We show that the algorithm enables the nodes to asymptotically agree on a solution when there are infinitely many solutions, determine the solution when there is exactly one, and detect that no solution exists when there are none. We also establish that the algorithm is globally exponentially convergent, derive an explicit lower bound on its convergence rate that it can do no worse than, and prove that the larger the network's algebraic connectivity, or the further away from being singular the system of equations, the larger this lower bound.

Finally, we address the open question of whether it is possible to calculate eigenvector centrality over a network. We provide an affirmative answer by presenting a class of continuous-time distributed algorithms and an asynchronous gossip algorithm, which allow every node i in a graph to compute the ith entry of the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix induced by the graph, as well as the corresponding eigenvalue, when node i knows only row i of the matrix. We show that each continuous-time distributed algorithm is a nonlinear networked dynamical system with a skew-symmetric structure, whose state is guaranteed to stay on a sphere, remain nonnegative, and converge asymptotically to said eigenvector at an $O(\frac{1}{t})$

rate. We also show that under a mild assumption on the gossiping pattern, the gossip algorithm is able to do the same.

Chapter 1 Introduction

1.1 Background and Motivation

Recent years have witnessed a tremendous growth in the types and applications of networked systems. From a network of computers and cell phones that form the Internet and cellular networks in the last couple of decades, to a network of tiny wireless sensors and huge wind turbines that monitor activities and generate renewable energy in the past few years, networked systems continue to change the way we live. Indeed, new types and applications of such systems (e.g., social networks) continue to emerge, offering potential that captures the fascination of scientists and engineers.

In many emerging and future applications of networked systems, systems in the network—commonly referred to interchangeably as *nodes* or *agents*—often have to cooperatively accomplish sophisticated tasks that require extensive sharing and rapid processing of information, as well as optimal formulation and precise coordination of actions, under a variety of constraints and uncertainties. For instance, sensors in a wireless network typically have to communicate in multi-hop fashion over unreliable physical channels for as long as possible, despite facing severe bandwidth and battery constraints. Therefore, although such networked systems offer promising potential, their design and operation are very challenging, to say the least.

A key factor that adds significantly to the challenge is the fact that in many of these networked systems, for various practical reasons it is often not feasible, or not advisable, to have a powerful centralized node, who knows all about the network topology and makes all the necessary decisions. For example, with the aforementioned wireless sensor network, transmitting information from every node and relaying it back to the centralized node may be too costly from a bandwidth and battery standpoint. Such transmission and relay is also vulnerable to node mobility and single-point failures, making it necessary to frequently maintain an overlay tree rooted at the centralized node, which maybe costly. Likewise, for networks deployed in battlefields and for social networks, doing so may simply be impermissible for vulnerability, security, and privacy reasons. As a result, the nodes must interact locally—perhaps only with immediately neighboring nodes—autonomously and collaboratively performing the tasks as if a centralized node is present. It follows that *distributed* (or *decentralized*) *algorithms*, which define how the nodes should locally interact, are critical to effectively realizing a variety of networked systems.

1.2 Literature Review

Recognizing the pressing need for distributed algorithms, researchers from a number of scientific and engineering disciplines (e.g., systems and control, computer science, operation research) have invested a great deal of research efforts in designing and analyzing such algorithms. In the field of systems and control, the efforts may be roughly grouped into three overlapping areas, namely, distributed consensus, distributed computation, and distributed optimization.

In distributed consensus, nodes or agents in a network seek to achieve an agreement on what they individually observe or experience. The agreement may be completely arbitrary (e.g., a platoon of vehicles may want to agree on a direction along which they all move), or it may be constrained to be some form of weighted average (e.g., a set of temperature sensors may want to determine the average of their individual temperature measurements). Being able to achieve such an agreement is often the basis of cooperation in a distributed system. Due to its significance, the distributed consensus problem has been widely studied in the literature, resulting in a rich collection of distributed algorithms in continuous-time (e.g., [1–13]) and in discrete-time with synchronous (e.g., [1,3,6–9,11,14–31]) and asynchronous (e.g., [20,32–50]) time models. In addition, such algorithms have been tailored to a variety of engineering applications, including but not limited to motion coordination [51], vehicle formation [52,53], and flocking [43,54,55].

In distributed computation, nodes in a network seek to compute a global, non-trivial quantity of common interest, whose value depends on either the graph topology or the scattered node observations. In this area, a growing number of problems have been addressed to date. Over the past decade, for example, notable research efforts have been devoted to the distributed computation of maximum [34,37,56–58], sum/count [14,33,34,57], power mean [34,56,59], resource redistribution [15], Kalman filters gains [12,60–63], linear functions [28,64–66], average-max-min [2], log-sum-exp [58], and a class of general functions [56,59,67]. More recently, some attention has been given to distributed computation of betweenness and closeness centrality [68–72], the spectrum of a graph and its corresponding eigenvectors [73–79], and the solution to linear equations [31,62,80–87].

Finally, in distributed optimization, each node typically observes a local, often convex, objective function and some local constraints, and all of the nodes wish to find an optimizer that minimizes the sum of their local objective functions subject to satisfying all their local constraints. This problem has an emerging number of applications, including to power grids [88–90], smart buildings [91, 92], and sensor networks [93, 94]. Motivated by its potential, the problem has been gaining much attention, leading to a large collection of distributed algorithms such as the incremental subgradient algorithms [93, 95–102], non-incremental ones [31, 103–110], zero-gradient-sum algorithms [50, 111–116], and various other algorithms [117–120].

1.3 Original Contributions

In this dissertation, we add to the growing literature on distributed computation by focusing on three specific problems of considerable significance: distributed computation of the spectrum of a graph, the solution to general linear equations, and the Perron-Frobenius eigenvector. Our original contributions can be summarized as follows.

First, we construct a novel, two-stage distributed algorithm that enables nodes in an undirected and connected graph to jointly estimate the spectrum of a matrix associated with the graph, which includes its adjacency and Laplacian matrices as special cases. Knowledge of the spectrum allows the nodes to infer about the graph structure. In the first stage, the algorithm uses a discrete-time linear iteration and the Cayley-Hamilton theorem to convert the problem into one of solving a set of linear equations, where each equation is known to a node. In the second stage, if the nodes happen to know that said matrix is cyclic, the algorithm uses a Lyapunov approach to asymptotically solve the equations with an exponential rate of convergence. If they do not know whether said matrix is cyclic, the algorithm uses a random perturbation approach and a structural controllability result to approximately solve the equations with an error that can be made small.

Second, we design a continuous-time distributed algorithm that allows nodes in an undirected and connected graph to collaboratively solve a general system of linear equations, where the only assumption is that each equation is known to at least one node. We show that the algorithm enables the nodes to asymptotically agree on a solution when there are infinitely many solutions, determine the solution when there is exactly one, and discover that no solution exists when there are none. In addition, we prove that the algorithm is globally exponentially convergent, derive an explicit lower bound on its convergence rate, and show that under certain conditions, the larger the graph's algebraic connectivity, or the further away from being singular the system of equations, the larger this lower bound.

Third, we devise a class of continuous-time distributed algorithms, which enable each node i in an undirected and connected graph to compute the ith entry of the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue, when node i knows only row i of the matrix. Knowledge of such entries allows the nodes to determine their eigenvector centrality representing their relative importance in the graph. We show that each continuous-time distributed algorithm in the class is a nonlinear networked dynamical system with a skew-symmetric structure, whose state is guaranteed to stay on a sphere, remain nonnegative, and converge asymptotically to said eigenvector at an $O(\frac{1}{t})$ rate.

We also show that the same idea that yields the continuous-time algorithms can be extended to a discrete-time setting, leading to an asynchronous gossip algorithm for computing the Perron-Frobenius eigenvector, which is provably asymptotically convergent at an $O(\frac{1}{k})$ rate under a mild assumption on the gossiping pattern.

1.4 Dissertation Outline

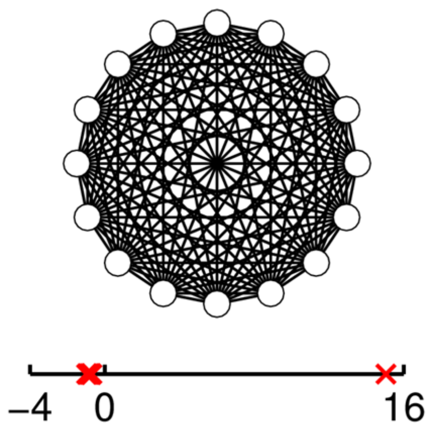
The outline of this dissertation is as follows: Chapter 2 studies distributed estimation of graph spectrum, in which a two-stage algorithm is developed. Chapter 3 constructs a continuous-time distributed algorithm for solving general linear equations over networks. Chapters 4 and 5 address the problem of distributed computation of the Perron-Frobenius eigenvector. In particular, Chapter 4 presents a class of continuous-time solutions, while Chapter 5 presents an asynchronous gossip counterpart. Finally, Chapter 6 concludes the dissertation and suggests a number of possible extensions as future work.

Chapter 2 Distributed Estimation of Graph Spectrum

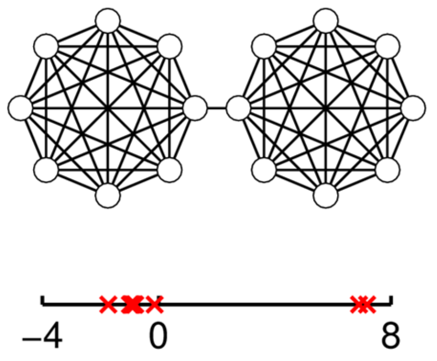
2.1 Introduction

The spectrum of a graph, defined as the set of eigenvalues of either its adjacency or Laplacian matrix, provide a useful characterization of the properties of the graph. For instance, as illustrated in Figure 2.1, the distribution of such eigenvalues offers insights into the shapes and sizes of communities in a network [121]. Indeed, for the complete graph depicted in Figure 2.1(a), its eigenvalues form two distinct clusters, with the first cluster having one dominant, positive eigenvalue and the second cluster having the rest of the eigenvalues concentrated around -1. For the barely connected graphs with two communities in Figures 2.1(b) and 2.1(c), their eigenvalues also form two clusters, but the clusters are much closer to each other, and there may be either one or two dominant, positive eigenvalues in the first cluster. For the cycle graph in Figure 2.1(d), its eigenvalues are more or less uniformly distributed over an interval centered at zero. As another example, the largest and smallest of such eigenvalues provides bounds on the maximum, minimum, and average node degrees [122]. The spectrum of a graph has also been used, for example, in chemistry, where it is associated with the stability of molecules [122], and in quantum mechanics, where it is related to the energy of Hamiltonian systems [122].

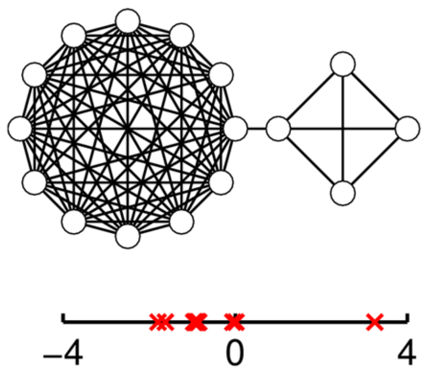
With the continued advances in technology that enable humans to build



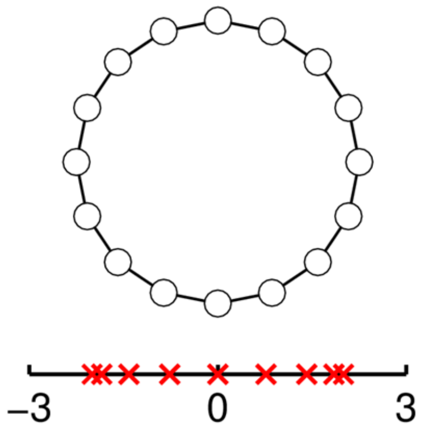
(a) A complete graph and its spectrum.



(b) A graph composed of two *identical* complete subgraphs connected by an edge, and its spectrum.



(c) A graph composed of two *different* complete subgraphs connected by an edge, and its spectrum.



(d) A cycle graph and its spectrum.

Figure 2.1: Distribution of the eigenvalues of a graph's adjacency matrix, represented by red crosses, offers insights into the shapes and sizes of communities in the graph.

increasingly complex networks, it is becoming desirable that nodes in a network have the ability to analyze the network themselves, such as decentralizedly computing the spectrum of the network, so that valuable understanding about, say, the network structure may be gained. Motivated by this, a number of distributed algorithms have been proposed in the literature, including [123–125] that consider estimation of the entire spectrum of the Laplacian matrix, and

[77–79] that focus on estimation of its second smallest eigenvalue (i.e., the algebraic connectivity).

In this chapter, we add to the literature by developing a two-stage distributed algorithm, which enables nodes in a graph to cooperatively estimate the spectrum of a matrix W associated with the graph. Unlike in [77–79, 123–125], the matrix W can be the adjacency or Laplacian matrix of the graph, a weighted version of these matrices, or any other matrix induced by the graph (see Chapter 2.2). To construct the algorithm, we first use a discrete-time linear iteration and the Cayley-Hamilton theorem to convert the original problem into an equivalent problem of solving a set of linear equations of the form $Ax = b$, where every row of A and b is known to a particular node (Chapter 2.3). We then show that the matrix A can be made almost surely nonsingular if the nodes happen to know that W is cyclic, but not necessarily so if they do not (Chapter 2.3). In the case of the former, we use a Lyapunov approach to asymptotically solve the equations with an exponential rate of convergence (Chapter 2.4.1). In the case of the latter, we use a random perturbation approach and a structural controllability result to approximately solve the equations with an error that can be made small (Chapter 2.4.2). A flowchart illustrating the aforementioned approach is depicted in Figure 2.2. Finally, we provide simulation results that demonstrate the effectiveness of our distributed algorithm (Chapter 2.5).

2.2 Problem Formulation

Consider a network modeled as an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ denotes the set of $N \geq 2$ nodes and $\mathcal{E} \subset$

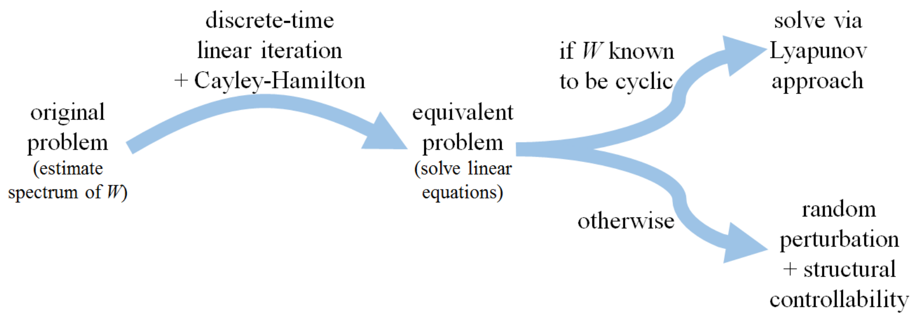

original problem (estimate spectrum of W)
discrete-time linear iteration + Cayley-Hamilton
equivalent problem (solve linear equations)
if W known to be cyclic
solve via Lyapunov approach
otherwise
random perturbation + structural controllability

Figure 2.2: Flowchart illustrating our approach to estimating graph spectrum.

$\{\{i, j\} : i, j \in \mathcal{V}, i \neq j\}$ denotes the set of edges. Any two nodes $i, j \in \mathcal{V}$ are neighbors and can communicate if and only if $\{i, j\} \in \mathcal{E}$. The set of neighbors of each node $i \in \mathcal{V}$ is denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$, and the communications are assumed to be delay- and error-free, with no quantization.

Suppose associated with the graph $\mathcal{G}$ is a square matrix $W = [w_{ij}] \in \mathbb{R}^{N\times N}$ satisfying the following assumption:

Assumption 2.1. The matrix W is such that for each $i, j \in \mathcal{V}$ with $i \neq j$, if $\{i, j\} \notin \mathcal{E}$, then $w_{ij} = w_{ji} = 0$.

Note that Assumption 2.1 allows w_{ii} $\forall i \in \mathcal{V}$ to be arbitrary. It also allows w_{ij} and w_{ji} $\forall \{i, j\} \in \mathcal{E}$ to be arbitrary and different. Thus, W can be the adjacency or Laplacian matrix of graph $\mathcal{G}$, a weighted version of these matrices, or any other matrix associated with $\mathcal{G}$ as long as Assumption 2.1 holds.

Suppose each node $i \in \mathcal{V}$ knows only $\mathcal{N}_i$, w_{ii}, and w_{ij} $\forall j \in \mathcal{N}_i$, which it prefers to not share with any of its neighbors due perhaps to security and privacy reasons. Yet, despite having only such local information about the graph $\mathcal{G}$ and matrix W, suppose every node $i \in \mathcal{V}$ wants to determine the spectrum of W, i.e., all the N eigenvalues of W, denoted as

$$\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(N)} \in \mathbb{C}, \tag{2.1}$$

where complex eigenvalues must be in the form of conjugate pairs. Finally, suppose each node $i \in \mathcal{V}$ knows the value of N, which is not an unreasonable assumption since each of them wants to determine the values of N objects.

Given the above, the goal of this chapter is to devise a distributed algorithm that enables every node $i \in \mathcal{V}$ to estimate the spectrum (2.1) of W with a guaranteed accuracy.

2.3 Forming a Set of Linear Equations

In this section, we show that by having the nodes execute a discrete-time linear iteration N times, the problem of finding the spectrum (2.1) of W may be converted into one of solving a set of linear equations with appealing properties.

Observe that although none of the nodes has complete information about $\mathcal{G}$ and W, each node $i \in \mathcal{V}$ knows the entire row i of W (since it knows w_{ii} and w_{ij} $\forall j \in \mathcal{N}_i$, and since $w_{ij} = 0$ $\forall j \notin \{i\} \cup \mathcal{N}_i$ by Assumption 2.1). This makes the nodes well-suited to carry out the discrete-time linear iteration

$$y_i(t+1) = w_{ii} y_i(t) + \sum_{j \in \mathcal{N}_i} w_{ij} y_j(t), \quad \forall i \in \mathcal{V},\ \forall t \in \mathbb{Z}_+, \tag{2.2}$$

which in matrix form may be written as

$$y(t+1) = W y(t), \quad \forall t \in \mathbb{Z}_+, \tag{2.3}$$

where $\mathbb{Z}_+ = \{0, 1, 2, \ldots\}$, $y_i(t) \in \mathbb{R}$ is maintained in node i's local memory, and

$$y(t) = \begin{bmatrix} y_1(t) & y_2(t) & \cdots & y_N(t) \end{bmatrix}^T \in \mathbb{R}^N. \tag{2.4}$$

Indeed, (2.2) or (2.3) can be implemented by having each node $i \in \mathcal{V}$ repeatedly send its $y_i(t)$ to every neighbor $j \in \mathcal{N}_i$.

Since (2.3) is a discrete-time linear system, we can write

$$y(t) = W^t y(0), \quad \forall t \in \mathbb{Z}_+, \tag{2.5}$$

so that

$$y(N) = W^N y(0). \tag{2.6}$$

By the Cayley-Hamilton theorem, W^N in (2.6) may be expressed as

$$W^N = -x^{(0)} I_N - x^{(1)} W - \cdots - x^{(N-1)} W^{N-1}, \tag{2.7}$$

where $I_n \in \mathbb{R}^{n \times n}$ is the identity matrix and the scalars $x^{(0)}, x^{(1)}, \ldots, x^{(N-1)} \in \mathbb{R}$ are the N coefficients of the characteristic polynomial of W, i.e.,

$$\begin{aligned} \det(\lambda I_N - W) &= (\lambda - \lambda^{(1)})(\lambda - \lambda^{(2)}) \cdots (\lambda - \lambda^{(N)}) \\ &= \lambda^N + x^{(N-1)} \lambda^{N-1} + \cdots + x^{(1)} \lambda + x^{(0)}. \end{aligned} \tag{2.8}$$

Substituting (2.7) into (2.6) and using (2.5), we obtain

$$\begin{aligned} y(N) &= (-x^{(0)} I_N - x^{(1)} W - \cdots - x^{(N-1)} W^{N-1}) y(0) \\ &= -x^{(0)} y(0) - x^{(1)} y(1) - \cdots - x^{(N-1)} y(N-1). \end{aligned} \tag{2.9}$$

By using (2.4), we can rewrite (2.9) as

$$\underbrace{\begin{bmatrix} y_1(0) & y_1(1) & \cdots & y_1(N-1) \\ y_2(0) & y_2(1) & \cdots & y_2(N-1) \\ \vdots & \vdots & \ddots & \vdots \\ y_N(0) & y_N(1) & \cdots & y_N(N-1) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(N-1)} \end{bmatrix}}_{x^*} = \underbrace{\begin{bmatrix} -y_1(N) \\ -y_2(N) \\ \vdots \\ -y_N(N) \end{bmatrix}}_{b}, \tag{2.10}$$

where, for later convenience, we denote the matrix on the left-hand side of (2.10) as $A \in \mathbb{R}^{N \times N}$, the vector of characteristic polynomial coefficients as $x^* \in \mathbb{R}^N$, and the vector on the right-hand side of (2.10) as $b \in \mathbb{R}^N$.

The matrix equation (2.10) suggests the following approach for finding the spectrum (2.1) of W: suppose each node $i \in \mathcal{V}$ selects an initial condition $y_i(0) \in \mathbb{R}$. Upon selecting the $y_i(0)$'s, suppose the nodes execute the discrete-time linear iteration (2.2) or equivalently (2.3) N times for $t \in \{0, 1, \ldots, N-1\}$. During the execution, suppose each node $i \in \mathcal{V}$ stores the resulting $N+1$ numbers $y_i(0), y_i(1), \ldots, y_i(N-1), y_i(N)$ in its local memory. Then, (2.10) is a set of N linear equations in which each node $i \in \mathcal{V}$ knows the entire row i of A and b, and in which the vector x^* of N characteristic polynomial coefficients $x^{(0)}, x^{(1)}, \ldots, x^{(N-1)}$ of W are the N unknowns. It follows that if A is nonsingular, and if the nodes are able to cooperatively solve (2.10) for the unique x^*, then each of them could determine on its own the N eigenvalues $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(N)}$ of W using (2.8) and a polynomial root-finding algorithm.

To realize the above approach, it is necessary that A in (2.10) is nonsingular. To see whether this can be ensured, observe from (2.4), (2.5), and (2.10) that A may be expressed as

$$A = \begin{bmatrix} y(0) & Wy(0) & \cdots & W^{N-1}y(0) \end{bmatrix}. \tag{2.11}$$

In the form (2.11), A is, interestingly, the controllability matrix of a fictitious discrete-time single-input linear system

$$z(t+1) = Wz(t) + y(0)u(t), \quad \forall t \in \mathbb{Z}_+, \tag{2.12}$$

where $z(t) \in \mathbb{R}^N$ is its state, $u(t) \in \mathbb{R}$ is its input, W is its state matrix, and $y(0)$ is its input matrix. Hence:

Proposition 2.1. *The matrix A in* (2.10) *or* (2.11) *is nonsingular if and only if the pair* $(W, y(0))$ *of the system* (2.12) *is controllable.*

Since W is given by the problem but $y(0)$ may be freely selected by the nodes, it may be possible to select $y(0)$ so that the pair $(W, y(0))$ is controllable. The following definition and lemmas examine this possibility:

Definition 2.1 ([126])**.** A square matrix with real entries is said to be *cyclic* if each of its distinct eigenvalues has a geometric multiplicity of 1.

Lemma 2.1. *If W is not cyclic, then for every $y(0) \in \mathbb{R}^N$, the pair $(W, y(0))$ is not controllable.*

Proof. Suppose W is not cyclic and let $y(0) \in \mathbb{R}^N$ be given. Then, by Definition 2.1, W has an eigenvalue $\lambda \in \mathbb{C}$ whose geometric multiplicity exceeds 1, i.e., $\text{rank}(W - \lambda I_N) < N - 1$. Since $y(0)$ is a column vector, $\text{rank}([W - \lambda I_N \mid y(0)]) < N$. Therefore, by statements (i) and (iv) of Theorem 3.1 in [126], the pair $(W, y(0))$ is not controllable. $\square$

Lemma 2.2. *If W is cyclic, then for almost every $y(0) \in \mathbb{R}^N$, the pair $(W, y(0))$ is controllable.*

Proof. According to Lemma 3.12 in [126], if $\mathcal{A} \in \mathbb{R}^{n \times n}$ is cyclic and $\mathcal{B} \in \mathbb{R}^{n \times m}$ is such that the pair $(\mathcal{A}, \mathcal{B})$ is controllable, then for almost every $v \in \mathbb{R}^m$, the pair $(\mathcal{A}, \mathcal{B}v)$ is controllable. Applying this lemma with $\mathcal{A} = W$, $\mathcal{B} = I_N$, and $v = y(0)$, and using the fact that the pair (W, I_N) is controllable, we conclude that so is the pair $(W, y(0))$. $\square$

Proposition 2.1 and Lemma 2.1 imply that W being cyclic is necessary for A in (2.10) or (2.11) to be nonsingular. Lemma 2.2, on the other hand, implies that W being cyclic is essentially sufficient because almost every $y(0) \in \mathbb{R}^N$ would work. This latter result is especially useful in a decentralized network because the result allows each node $i \in \mathcal{V}$ to select its $y_i(0) \in \mathbb{R}$ independently from other nodes and randomly from any continuous probability distribution before executing (2.2) or (2.3), and be almost sure that the resulting A would be nonsingular.

Motivated by the above analysis, in the rest of this chapter we consider separately the following two scenarios:

Scenario 1. The nodes know that W is cyclic.

Scenario 2. The nodes do not know whether W is cyclic, or know that W is not cyclic.

We consider Scenarios 1 and 2 separately because Scenario 1 is easier to deal with (in Chapter 2.4.1) and its treatment helps us deal with Scenario 2 (in Chapter 2.4.2). We note that both of these scenarios arise in applications. For instance, if the graph $\mathcal{G}$ represents a sensor network and the entries w_{ii} $\forall i \in \mathcal{V}$ and w_{ij} $\forall \{i,j\} \in \mathcal{E}$ of W represent random sensor measurements with continuous probability distributions, then Scenario 1 takes place as the nodes could say with near certainty that W is cyclic because almost every n-by-n matrix has n distinct eigenvalues and, thus, is cyclic. In contrast, if W represents the adjacency or Laplacian matrix of $\mathcal{G}$, then Scenario 2 takes place as W would be cyclic if $\mathcal{G}$ is, say, a path graph [122] and would not be cyclic if $\mathcal{G}$ is, say, a complete or cycle graph [122], which the nodes could not tell because they only have local information about $\mathcal{G}$.

To summarize, in this section we have transformed the problem of finding the spectrum (2.1) of W into one of solving the set of linear equations (2.10), in which each node $i \in \mathcal{V}$ knows the entire row i of A and b, and in which A can be made almost surely nonsingular in Scenario 1, but not necessarily so in Scenario 2.

2.4 Solving the Linear Equations

2.4.1 Scenario 1: Cyclic Case

In this subsection, we focus on Scenario 1 and develop a continuous-time distributed algorithm that enables the nodes to asymptotically solve the set of linear equations (2.10) with an exponential rate of convergence.

To facilitate the development, we assume that the nodes have executed (2.2) or (2.3) to arrive at (2.10). Moreover, since A in (2.10) can be made almost surely nonsingular in this Scenario 1, we assume that it *is* nonsingular throughout the subsection. With these assumptions, for each $i \in \mathcal{V}$ let $a_i = \begin{bmatrix} y_i(0) & y_i(1) & \cdots & y_i(N-1) \end{bmatrix}^T \in \mathbb{R}^N$ and $b_i = -y_i(N) \in \mathbb{R}$, so that (2.10) may be stated as

$$\underbrace{\begin{bmatrix} - a_1^T - \\ - a_2^T - \\ \vdots \\ - a_N^T - \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x^{(0)} \\ x^{(1)} \\ \vdots \\ x^{(N-1)} \end{bmatrix}}_{x^*} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}}_{b}, \qquad (2.13)$$

where a_i and b_i are known to node i because (2.2) or (2.3) has been executed. In addition to knowing a_i and b_i, suppose each node $i \in \mathcal{V}$ maintains in its local memory an estimate $x_i(t) = \begin{bmatrix} x_i^{(0)}(t) & x_i^{(1)}(t) & \cdots & x_i^{(N-1)}(t) \end{bmatrix}^T \in \mathbb{R}^N$ of the unknown, unique solution $x^* \in \mathbb{R}^N$, where here $t \in [0, \infty)$ denotes continuous-time (unlike in Chapter 2.3 where $t \in \mathbb{Z}_+$ denotes discrete-time). Furthermore, let $\mathbf{x}(t) = (x_1(t), x_2(t), \ldots, x_N(t)) \in \mathbb{R}^{N^2}$ and $\mathbf{x}^* = (x^*, x^*, \ldots, x^*) \in \mathbb{R}^{N^2}$

be vectors obtained by stacking the N estimates $x_i(t)$'s and N copies of the solution x^*.

To come up with a distributed algorithm that gradually drives $\mathbf{x}(t)$ to $\mathbf{x}^*$, consider a quadratic Lyapunov function candidate $V : \mathbb{R}^{N^2} \to \mathbb{R}$, defined as

$$\begin{aligned} V(\mathbf{x}) = &\sum_{i \in \mathcal{V}} \alpha_i (a_i^T x_i - b_i)^2 \\ &+ \sum_{\{i,j\} \in \mathcal{E}} \beta_{\{i,j\}} (x_i - x_j)^T (x_i - x_j), \end{aligned} \tag{2.14}$$

where $\alpha_i > 0 \ \forall i \in \mathcal{V}$ and $\beta_{\{i,j\}} > 0 \ \forall \{i,j\} \in \mathcal{E}$ are parameters. Notice that each term in the first summation in (2.14) is a measure of how far away from the hyperplane $\{z \in \mathbb{R}^N : a_i^T z = b_i\}$ the estimate $x_i(t)$ is. Moreover, because A is nonsingular and because of (2.13), the N hyperplanes $\{z \in \mathbb{R}^N : a_i^T z = b_i\}$ $\forall i \in \mathcal{V}$ have a unique intersection at x^*. Furthermore, the second summation in (2.14) is a measure of the disagreement among the estimates $x_i(t)$'s. Hence, both the first and second summations in (2.14) are only positive semidefinite functions of $\mathbf{x}$. However, as the following proposition shows, adding them up makes V a legitimate Lyapunov function candidate:

Proposition 2.2. *If A in* (2.13) *is nonsingular, then the function V in* (2.14) *is positive definite with respect to $\mathbf{x}^*$.*

Proof. Clearly, V is a positive semidefinite function of $\mathbf{x}$. To show that it is positive definite with respect to $\mathbf{x}^*$, we show that $V(\mathbf{x}) = 0$ if and only if $\mathbf{x} = \mathbf{x}^*$. Suppose $\mathbf{x} = \mathbf{x}^*$. Then, $a_i^T x_i - b_i = 0 \ \forall i \in \mathcal{V}$ according to (2.13). In addition, the second summation in (2.14) drops out. Therefore, $V(\mathbf{x}) = 0$.

Next, suppose $V(\mathbf{x}) = 0$. Then,

$$a_i^T x_i = b_i, \quad \forall i \in \mathcal{V}, \tag{2.15}$$

$$x_i = x_j, \quad \forall \{i,j\} \in \mathcal{E}. \tag{2.16}$$

Since $\mathcal{G}$ is connected, (2.16) implies that there exists $\tilde{x} \in \mathbb{R}^N$ such that $x_i = \tilde{x}$ $\forall i \in \mathcal{V}$. Substituting this into (2.15), we get $a_i^T \tilde{x} = b_i$ $\forall i \in \mathcal{V}$ or, equivalently, $A\tilde{x} = b$. Since A is nonsingular, we have $\tilde{x} = x^*$, so that $\mathbf{x} = \mathbf{x}^*$. □

Remark 2.1. Notice that V in (2.14) can also be written as

$$V(\mathbf{x}) = (\mathbf{x} - \mathbf{x}^*)^T P (\mathbf{x} - \mathbf{x}^*),$$

where $P = P^T \in \mathbb{R}^{N^2 \times N^2}$ is positive definite and given by

$$P = \begin{bmatrix} \alpha_1 a_1 a_1^T & & & 0 \\ & \alpha_2 a_2 a_2^T & & \\ & & \ddots & \\ 0 & & & \alpha_N a_N a_N^T \end{bmatrix} + L_\beta \otimes I_N,$$

where $\otimes$ denotes the Kronecker product and $L_\beta = [L_{ij}] \in \mathbb{R}^{N \times N}$ is a weighted Laplacian matrix of $\mathcal{G}$ with $L_{ii} = \sum_{j \in \mathcal{N}_i} \beta_{\{i,j\}}$, $L_{ij} = -\beta_{\{i,j\}}$ if $\{i,j\} \in \mathcal{E}$, and $L_{ij} = 0$ if $i \neq j$ and $\{i,j\} \notin \mathcal{E}$.

With Proposition 2.2 in hand, we next take the time derivative of V along the state trajectory $\mathbf{x}(t)$ to obtain

$$\begin{aligned} \dot{V}(\mathbf{x}(t)) = 2 \sum_{i \in \mathcal{V}} \Big[& \alpha_i (a_i^T x_i(t) - b_i) a_i \\ & + \sum_{j \in \mathcal{N}_i} \beta_{\{i,j\}} (x_i(t) - x_j(t)) \Big] \dot{x}_i(t), \quad \forall t \in [0, \infty). \end{aligned} \tag{2.17}$$

Examining (2.17), we see that $\dot{V}(\mathbf{x}(t))$ can be made negative semidefinite—at the very least—by letting each $\dot{x}_i(t)$ be the negative of the expression within the brackets in (2.17), i.e.,

$$\dot{x}_i(t) = -\alpha_i(a_i^T x_i(t) - b_i)a_i - \sum_{j\in\mathcal{N}_i} \beta_{\{i,j\}}(x_i(t) - x_j(t)),$$
$$\forall i \in \mathcal{V},\ \forall t \in [0,\infty). \tag{2.18}$$

The following theorem asserts that the continuous-time system (2.18) possesses an excellent property:

Theorem 2.1. *If A in* (2.13) *is nonsingular, then the system* (2.18) *has a unique equilibrium point at $\mathbf{x}^*$ that is globally exponentially stable, so that $\forall \mathbf{x}(0) \in \mathbb{R}^{N^2}$, $\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x}^*$, i.e., $\lim_{t\to\infty} x_i(t) = x^*$ $\forall i \in \mathcal{V}$.*

Proof. For each $i \in \mathcal{V}$, setting $\dot{x}_i(t)$ in (2.18) to zero yields

$$0 = -\alpha_i(a_i^T x_i - b_i)a_i - \sum_{j\in\mathcal{N}_i} \beta_{\{i,j\}}(x_i - x_j). \tag{2.19}$$

Summing both sides of (2.19) over $i \in \mathcal{V}$ gives

$$0 = \sum_{i\in\mathcal{V}} -\alpha_i(a_i^T x_i - b_i)a_i. \tag{2.20}$$

Due to (2.13) and to A being nonsingular, the vectors $a_1, a_2, \ldots, a_N$ in (2.20) are linearly independent in $\mathbb{R}^N$. Thus,

$$0 = -\alpha_i(a_i^T x_i - b_i), \quad \forall i \in \mathcal{V}. \tag{2.21}$$

Substituting (2.21) back into (2.19) results in

$$0 = \sum_{j\in\mathcal{N}_i} \beta_{\{i,j\}}(x_i - x_j), \quad \forall i \in \mathcal{V},$$

which is equivalent to

$$0 = (L_\beta \otimes I_N)\mathbf{x}, \tag{2.22}$$

where $\otimes$ and L_β have been defined in Remark 2.1. Since $\mathcal{G}$ is connected, (2.22) implies that $x_i = \tilde{x}\ \forall i \in \mathcal{V}$ for some $\tilde{x} \in \mathbb{R}^N$. Substituting this into (2.21) yields $a_i^T \tilde{x} = b_i\ \forall i \in \mathcal{V}$. Since A is nonsingular, we have $\tilde{x} = x^*$, i.e., $\mathbf{x} = \mathbf{x}^*$. Hence, the system (2.18) has a unique equilibrium point at $\mathbf{x}^*$. Since for each $i \in \mathcal{V}$ the right-hand side of (2.18) is the negative of the expression within the brackets in (2.17), $\dot{V}(\mathbf{x}(t))$ is negative definite with respect to $\mathbf{x}^*$. Therefore, the equilibrium point $\mathbf{x}^*$ is globally exponentially stable. $\square$

Having established Theorem 2.1, we now relate it back to the original problem of finding the spectrum (2.1) of W. To this end, suppose each node $i \in \mathcal{V}$ maintains in its local memory an estimate $\lambda_i^{(\ell)}(t) \in \mathbb{C}$ of the unknown, ℓth eigenvalue $\lambda^{(\ell)}$ of W for $\ell \in \{1, 2, \ldots, N\}$. Also suppose at each time $t \in [0, \infty)$, node i lets its N estimates $\lambda_i^{(\ell)}(t)$'s be the roots of an Nth-order polynomial formed by the estimate $x_i(t) = \begin{bmatrix} x_i^{(0)}(t) & x_i^{(1)}(t) & \cdots & x_i^{(N-1)}(t) \end{bmatrix}^T$ that is also stored in its local memory, i.e.,

$$\begin{aligned} &(\lambda - \lambda_i^{(1)}(t))(\lambda - \lambda_i^{(2)}(t)) \cdots (\lambda - \lambda_i^{(N)}(t)) \\ &\quad = \lambda^N + x_i^{(N-1)}(t)\lambda^{N-1} + \cdots + x_i^{(1)}(t)\lambda + x_i^{(0)}(t), \\ &\qquad\qquad \forall i \in \mathcal{V},\ \forall t \in [0, \infty), \end{aligned} \tag{2.23}$$

which can be implemented using a polynomial root-finding algorithm that is embedded in node i. Then, because $(\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(N)})$ in (2.8) is a continuous function of x^* and because $(\lambda_i^{(1)}(t), \lambda_i^{(2)}(t), \ldots, \lambda_i^{(N)}(t))$ in (2.23) is the *same* continuous function of $x_i(t)$, Theorem 2.1 implies that

$$\lim_{t \to \infty} \lambda_i^{(\ell)}(t) = \lambda^{(\ell)}, \quad \forall i \in \mathcal{V},\ \forall \ell \in \{1, 2, \ldots, N\}. \tag{2.24}$$

Equation (2.24), in turn, implies that the system (2.18) is a continuous-time distributed algorithm that enables the nodes to asymptotically learn the spectrum (2.1) of W.

Putting together the development in Sections 2.3 and 2.4.1, we obtain the following two-stage distributed algorithm, which is applicable to this Scenario 1:

Algorithm 2.1 (For Scenario 1)**.**

1. Each node $i \in \mathcal{V}$ selects its $y_i(0) \in \mathbb{R}$ independently from other nodes and randomly from any continuous probability distribution.
2. Upon completion, the nodes execute (2.2) or (2.3) N times for $t \in \{0, 1, \ldots, N-1\}$, so that each node $i \in \mathcal{V}$ gradually learns the entire row i of A and b in (2.10).
3. Upon completion, the nodes execute (2.18) and (2.23) indefinitely for $t \in [0, \infty)$, so that each node $i \in \mathcal{V}$ is able to continuously update its $x_i(t)$ and $\lambda_i^{(\ell)}(t)$'s. ■

Remark 2.2. The current literature offers a number of distributed algorithms [31,84–86] that may be used to solve linear equations (2.10). These algorithms are different from (2.18) in that they force the state of each node to stay in an affine set, whereas (2.18) allows the state to freely roam the state space. Additional differences between them are discussed in Chapter 3.1.

2.4.2 Scenario 2: Acyclic Case

In this subsection, we focus on Scenario 2 and provide a slightly different algorithm that enables the nodes to approximately solve (2.10) with an error that can be made small.

Recall that Scenario 2 represents a situation where the nodes either do not know whether W is cyclic, or somehow know that W is not cyclic. Consequently, they either do not know whether A in (2.10) is nonsingular, or know that A is singular. Although the nodes could still apply Algorithm 2.1, there is no guarantee that their estimates $x_i(t)$'s would converge to x^*. One way to address this issue is to have the nodes randomly perturb the matrix W and vector $y(0)$, so that the resulting A in (2.11) is hopefully nonsingular. Of course, such a random perturbation approach no longer allows them to asymptotically determine the exact spectrum of W. However, getting an estimate of the spectrum of W may be sufficient in some applications. Thus, we will adopt this random perturbation approach in this Scenario 2.

For notational simplicity, let the matrix associated with the graph $\mathcal{G}$ be denoted as $\overline{W} = [\overline{w}_{ij}] \in \mathbb{R}^{N \times N}$ instead of $W = [w_{ij}]$, and let W instead denote a perturbed version of $\overline{W}$. In addition, let $\overline{x}^{(\ell)}$'s and $\overline{\lambda}^{(\ell)}$'s denote, respectively, the characteristic polynomial coefficients and eigenvalues of $\overline{W}$ that the nodes wish to determine, and let $x^{(\ell)}$'s and $\lambda^{(\ell)}$'s denote those of W as before. Moreover, let the perturbed matrix W be obtained from $\overline{W}$ in a decentralized manner as follows: prior to executing (2.2) or (2.3), each node $i \in \mathcal{V}$ lets

$$w_{ii} = \overline{w}_{ii} + \delta_{ii}, \quad \forall i \in \mathcal{V}, \tag{2.25}$$

$$w_{ij} = \overline{w}_{ij} + \delta_{ij}, \quad \forall i \in \mathcal{V},\ \forall j \in \mathcal{N}_i, \tag{2.26}$$

where the δ_{ii}'s and δ_{ij}'s are independent, uniformly distributed random variables in the interval $[-a, a]$, so that $a > 0$ represents the perturbation magnitude. Notice that since $\overline{w}_{ij} = 0\ \forall i \in \mathcal{V}\ \forall j \notin \{i\} \cup \mathcal{N}_i$ by Assumption 2.1,

$$w_{ij} = 0, \quad \forall i \in \mathcal{V},\ \forall j \notin \{i\} \cup \mathcal{N}_i \tag{2.27}$$

as well. Also note that because the nodes are slated to select their $y_i(0)$'s independently and randomly from a continuous probability distribution, there is no need to further randomly perturb these $y_i(0)$'s.

The following lemma uses a structural controllability result to show that the aforementioned approach is effective:

Lemma 2.3. *If W is as defined in (2.25)–(2.27) and $y(0)$ is as defined in Step 1 of Algorithm 2.1, then A in (2.11) is almost surely nonsingular.*

Proof. Reconsider the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from Chapter 2.2. Let $\mathcal{S} = \{(\mathcal{A}, \mathcal{B}) \in \mathbb{R}^{N \times N} \times \mathbb{R}^N : \mathcal{A}_{ij} = 0 \text{ if } i \neq j \text{ and } \{i, j\} \notin \mathcal{E}\}$ and $\mathcal{S}_c = \{(\mathcal{A}, \mathcal{B}) \in \mathcal{S} : (\mathcal{A}, \mathcal{B}) \text{ is controllable}\} \subset \mathcal{S}$. In addition, let $\mathcal{A}^* = \operatorname{diag}(1, 2, \ldots, N) \in \mathbb{R}^{N \times N}$ and $\mathcal{B}^* \in \mathbb{R}^N$ be the all-one vector. Then, $(\mathcal{A}^*, \mathcal{B}^*) \in \mathcal{S}$ according to the definition of $\mathcal{S}$. Moreover, $(\mathcal{A}^*, \mathcal{B}^*) \in \mathcal{S}_c$ because the controllability matrix formed by $(\mathcal{A}^*, \mathcal{B}^*)$ is a Vandermonde matrix that is nonsingular. These two properties of $(\mathcal{A}^*, \mathcal{B}^*)$, along with the definition of structural controllability [127], imply that every $(\mathcal{A}, \mathcal{B}) \in \mathcal{S}$ is structurally controllable. Next, let $(\mathcal{A}, \mathcal{B}) \in \mathcal{S}$ and $\epsilon > 0$ be given. Then, by Proposition 1 of [127], there exists $(\mathcal{A}_c, \mathcal{B}_c) \in \mathcal{S}_c$ such that $\|\mathcal{A} - \mathcal{A}_c\| < \epsilon$ and $\|\mathcal{B} - \mathcal{B}_c\| < \epsilon$. Hence, $\mathcal{S}_c$ is a dense subset of $\mathcal{S}$. Lastly, note that $(W, y(0)) \in \mathcal{S}$ due to Assumption 2.1, (2.25)–(2.27), and Step 1 of Algorithm 2.1. Since $\mathcal{S}_c$ is a dense subset of $\mathcal{S}$, $(W, y(0))$ is almost surely in $\mathcal{S}_c$. Therefore, by Proposition 2.1, A in (2.11) is almost surely nonsingular. $\square$

As it follows from Lemma 2.3, by having the nodes perform the extra step described in (2.25)–(2.27), the results developed in Sections 2.3 and 2.4.1 become applicable to this Scenario 2. Furthermore, because both the characteristic polynomial coefficients and eigenvalues of a matrix are continuous functions of its entries, by having the nodes decrease the perturbation magnitude a toward zero, the differences between the $x^{(\ell)}$'s and $\lambda^{(\ell)}$'s of W and the $\overline{x}^{(\ell)}$'s and $\overline{\lambda}^{(\ell)}$'s of $\overline{W}$ can be made arbitrarily small, at least in principle. Note, however, that numerical issues may arise when a is too small, or when the resulting A is ill-conditioned. At present, we do not have answers to these numerical issues, and we believe they are important future research directions.

Based on the above, we obtain the following two-stage distributed algorithm for this Scenario 2:

Algorithm 2.2 (For Scenario 2)**.**

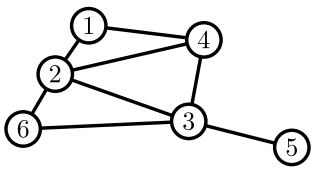
1. Each node $i \in \mathcal{V}$ executes (2.25)–(2.27) to obtain a perturbed matrix W.
2. The remaining steps are identical to those of Algorithm 2.1. ■

2.5 Simulation Results

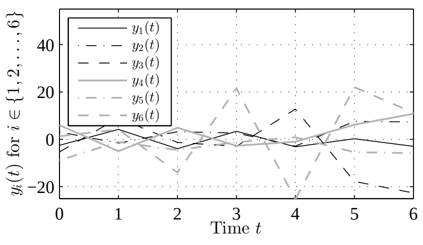
In this section, we present two sets of simulation results that demonstrate the effectiveness of Algorithm 2.1 for Scenario 1 and Algorithm 2.2 for Scenario 2.

2.5.1 Simulation of Algorithm 2.1 for Scenario 1

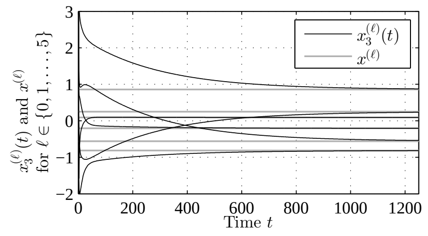
Consider a sensor network with $N = 6$ nodes, modeled as an undirected, connected graph $\mathcal{G}$, whose topology is shown in Figure 2.3(a). Suppose associated with the graph $\mathcal{G}$ is a 6-by-6 matrix W, whose entries satisfy As-



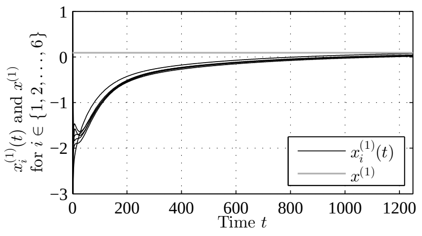
(a) A 6-node graph.



(b) Data points $y_i(t)$ for $i \in \{1, 2, \ldots, 6\}$ and $t \in \{0, 1, \ldots, 6\}$ that form the set of linear equations (2.10).



(c) Node 3's estimate $x_3^{(\ell)}(t)$ of the ℓth characteristic polynomial coefficient $x^{(\ell)}$ for $\ell \in \{0, 1, \ldots, 5\}$.



(d) Node i's estimate $x_i^{(1)}(t)$ of the first characteristic polynomial coefficient $x^{(1)}$ for $i \in \{1, 2, \ldots, 6\}$.

Figure 2.3: Performance of Algorithm 2.1 for Scenario 1.

sumption 2.1 and represent random sensor measurements given by

$$W = \begin{bmatrix} -0.10 & -0.24 & 0 & 0.78 & 0 & 0 \\ 0.24 & 0.53 & 0.39 & -0.04 & 0 & -0.19 \\ 0 & 0.34 & 0.21 & 1.15 & -0.13 & 0.71 \\ -0.26 & -0.21 & 0.32 & -0.54 & 0 & 0 \\ 0 & 0 & -0.45 & 0 & 0.39 & 0 \\ 0 & 0.47 & -0.84 & 0 & 0 & -1.35 \end{bmatrix}.$$

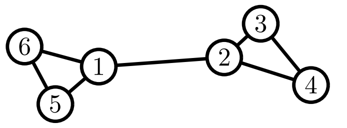
Assuming that such measurements are realizations of continuously distributed random variables, the nodes are almost certain that W is cyclic, so that Scenario 1 takes place. Thus, to determine all the eigenvalues $\lambda^{(\ell)}$'s of W, which are given by $-1.02 \pm 0.55i$, $-0.004 \pm 0.46i$, 0.38, and 0.81, the nodes may apply Algorithm 2.1.

Figures 2.3(b)–2.3(d) display the result of simulating Algorithm 2.1 with $\alpha_i = 10\ \forall i \in \mathcal{V}$ and $\beta_{\{i,j\}} = 10\ \forall \{i, j\} \in \mathcal{E}$. Specifically, Figure 2.3(b) shows the data points $y_i(t)$ for $i \in \{1, 2, \ldots, 6\}$ and $t \in \{0, 1, \ldots, 6\}$ that are used to form the set of linear equations (2.10). Figure 2.3(c) shows, as a function of time t, node 3's estimate $x_3^{(\ell)}(t)$ of the ℓth characteristic polynomial coefficient $x^{(\ell)}$ of W for $\ell \in \{0, 1, \ldots, 5\}$. Likewise, Figure 2.3(d) shows node i's estimate $x_i^{(1)}(t)$ of the first coefficient $x^{(1)}$ for $i \in \{1, 2, \ldots, 6\}$. (Note that instead of including plots of $x_i^{(\ell)}(t)$ for all $i \in \{1, 2, \ldots, 6\}$ and $\ell \in \{0, 1, \ldots, 5\}$, we included only two representative ones, in Figures 2.3(c) and 2.3(d).) Observe that despite having only local information about $\mathcal{G}$ and W, the nodes are able to utilize Algorithm 2.1 to asymptotically determine all the characteristic polynomial coefficients $x^{(\ell)}$'s of W and, hence, all its eigenvalues $\lambda^{(\ell)}$'s.

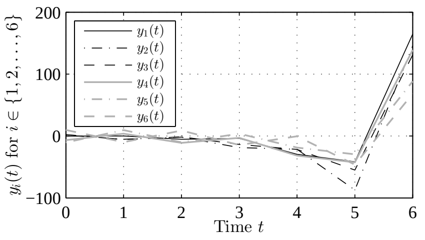
2.5.2 Simulation of Algorithm 2.2 for Scenario 2

Consider next an undirected, connected graph $\mathcal{G}$ with $N = 6$ nodes, whose topology is shown in Figure 2.4(a). Let $\overline{W}$ represent the adjacency matrix of $\mathcal{G}$ and suppose the nodes wish to determine all the eigenvalues $\overline{\lambda}^{(\ell)}$'s of $\overline{W}$, which are given by -1.73, -1, -1, -0.41, 1.73, and 2.41. Because they only have local information about $\mathcal{G}$, the nodes do not know whether $\overline{W}$ is cyclic, so that Scenario 2 takes place. (In fact, $\overline{W}$ in this particular example is not cyclic because it is symmetric and has repeated eigenvalues, at -1.) Therefore, the nodes have to apply Algorithm 2.2. In doing so, they let the perturbation magnitude be $a = 0.2$ and obtain from (2.25)–(2.27) a perturbed matrix W given by

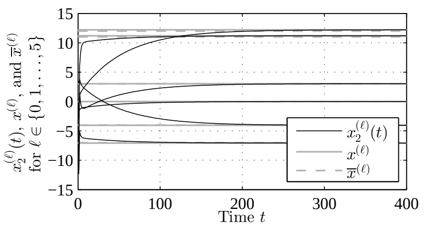
$$W = \begin{bmatrix} 0 & 1.04 & 0 & 0 & 1.01 & 0.94 \\ 0.98 & 0 & 1.04 & 1.12 & 0 & 0 \\ 0 & 0.98 & 0 & 1.06 & 0 & 0 \\ 0 & 0.95 & 1.01 & 0 & 0 & 0 \\ 0.98 & 0 & 0 & 0 & 0 & 1.01 \\ 0.97 & 0 & 0 & 0 & 0.92 & 0 \end{bmatrix},$$



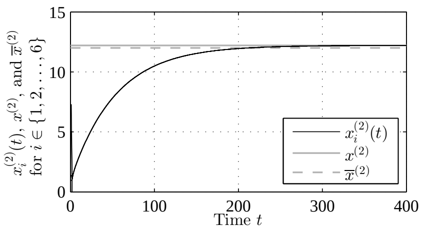
(a) A 6-node graph.



(b) Data points $y_i(t)$ for $i \in \{1, 2, \ldots, 6\}$ and $t \in \{0, 1, \ldots, 6\}$ that form the set of linear equations (2.10).



(c) Node 2's estimate $x_2^{(\ell)}(t)$ of the ℓth perturbed and true characteristic polynomial coefficients $x^{(\ell)}$ and $\overline{x}^{(\ell)}$ for $\ell \in \{0, 1, \ldots, 5\}$.



(d) Node i's estimate $x_i^{(2)}(t)$ of the 2nd perturbed and true characteristic polynomial coefficients $x^{(2)}$ and $\overline{x}^{(2)}$ for $i \in \{1, 2, \ldots, 6\}$.

Figure 2.4: Performance of Algorithm 2.2 for Scenario 2.

whose eigenvalues $\lambda^{(\ell)}$'s are -1.74, -0.97, -1.03, -0.40, 1.73, and 2.43, which are all distinct and slightly different from the eigenvalues $\overline{\lambda}^{(\ell)}$'s of $\overline{W}$.

Figures 2.4(b)–2.4(d) display the result of simulating Algorithm 2.2 with $\alpha_i = 100\ \forall i \in \mathcal{V}$ and $\beta_{\{i,j\}} = 10\ \forall \{i, j\} \in \mathcal{E}$, using a format similar to that of Figures 2.3(b)–2.3(d). The only difference is that Figures 2.4(c) and 2.4(d) show not only the characteristic polynomial coefficients $x^{(\ell)}$'s of the "perturbed" W, but also the characteristic polynomial coefficients $\overline{x}^{(\ell)}$'s of the "true" $\overline{W}$. Observe that with Algorithm 2.2, the nodes are able to asymptotically determine the $x^{(\ell)}$'s and $\lambda^{(\ell)}$'s. In other words, they are able to approximately calculate the $\overline{x}^{(\ell)}$'s and $\overline{\lambda}^{(\ell)}$'s with small errors.

2.6 Conclusion

In this chapter, we have designed and analyzed a two-stage distributed algorithm that enables nodes in a graph to cooperatively estimate the graph spectrum. We have shown that asymptotically accurate estimation can be achieved if the nodes know that the associated matrix is cyclic, and estimation with small errors can be achieved if they do not.

Chapter 3 A Distributed Algorithm for Solving General Linear Equations

3.1 Introduction

Solving a system of linear equations is a fundamental problem with countless applications. In this chapter, we address the problem of solving such equations over a network, where the equation data are scattered across the network. More specifically, we consider an undirected and connected graph with N nodes, accompanied by a system of m linear equations with n unknowns of the form

$$Ax = b, \tag{3.1}$$

where each row of $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ is known to at least one node, and where every node wishes to find a solution $x \in \mathbb{R}^n$ to (3.1), whenever it exists. Since each node knows only part of the equation data, none of them could solve (3.1) on its own. As a result, the nodes must cooperatively do so, preferably in a distributed fashion and preferably without having to share their equation data with others.

This chapter is intended to create an algorithm that equips the nodes with such capabilities. We develop a continuous-time distributed algorithm that allows the nodes to solve a general form of (3.1), where the number of nodes N, number of equations m, and number of unknowns n may be arbitrary.

In addition, the existence and uniqueness of a solution x are not assumed and not known by the nodes in advance, the set $\mathcal{K}_i$ of rows of A and b known to each node i may be arbitrary or even empty, and the only restriction is that every row of A and b is known to one or more nodes. We show that the algorithm enables the nodes to asymptotically agree on a solution when (3.1) has infinitely many solutions, and asymptotically determine the solution when (3.1) has exactly one. We also show that the algorithm enables at least one pair of neighboring nodes to asymptotically discover that no solution exists when (3.1) has none. Moreover, we prove that the algorithm—which is an affine networked dynamical system—is globally exponentially convergent and derive an explicit lower bound on its convergence rate, which it can do no worse than. Furthermore, we show that when A is square and nonsingular and when each row of A and b is known to exactly one node, the larger the algebraic connectivity of the graph, or the larger the smallest singular value of A (which is its distance to the nearest singular matrix), the larger this lower bound.

We note that the current literature offers a number of distributed algorithms for solving (3.1), including those reported in [31, 84–86]. The results in [31, 84–86], however, are different from the ones in this chapter in at least three ways: first, the graphs considered in [31, 84–86] may be directed with time-varying topologies, whereas the one considered here has to be undirected with a fixed topology. Second, the algorithms proposed in [31, 84–86] force the state of each node to stay in an affine set, following the idea of *constrained consensus*. In contrast, the algorithm here allows the state to freely roam the state space. Third, the convergence rate result here captures not only the impact of the graph topology, but also that of the problem (e.g., how close to being parallel the rows of A are). The latter is not captured in [31, 84–86]. Lastly, we note that there is a related line of work [82,83,87] on solving (3.1), but the setup is different: in [82,83,87], $A = \sum_{i=1}^{N} A_i$ and $b = \sum_{i=1}^{N} b_i$, where A_i is a symmetric positive definite matrix and b_i is a vector, both known to and only to node i.

The outline of this chapter is as follows: Chapter 3.2 formulates the problem. Chapters 3.3 and 3.4 design and analyze the algorithm. Chapter 3.5 analyzes its convergence rate. Finally, Chapter 3.7 concludes the chapter.

3.2 Problem Formulation

Consider a network modeled as an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ denotes the set of $N \geq 2$ nodes and $\mathcal{E} \subset \{\{i,j\} : i,j \in \mathcal{V}, i \neq j\}$ denotes the set of edges. Any two nodes $i,j \in \mathcal{V}$ are neighbors and can communicate if and only if $\{i,j\} \in \mathcal{E}$. The set of neighbors of each node $i \in \mathcal{V}$ is denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : \{i,j\} \in \mathcal{E}\}$, and the communications are assumed to be delay- and error-free, with no quantization.

Suppose associated with the graph $\mathcal{G}$ is a system of linear equations $Ax = Y$, which has $m \geq 1$ equations and $n \geq 1$ unknowns, and which can be partitioned as

$$\underbrace{\begin{bmatrix} — a_1^T — \\ — a_2^T — \\ \vdots \\ — a_m^T — \end{bmatrix}}_{A} x = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{Y}, \tag{3.2}$$

where $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, $Y \in \mathbb{R}^m$, $a_k \in \mathbb{R}^n \ \forall k \in \mathcal{K}$, $y_k \in \mathbb{R} \ \forall k \in \mathcal{K}$, and $\mathcal{K} = \{1, 2, \ldots, m\}$. Note that there is no restriction on the values of m, n, A, and Y. Thus, (3.2) either has a unique solution x, infinitely many solutions, or no solution.

Suppose each node $i \in \mathcal{V}$ knows only $\mathcal{N}_i$, a_k, and y_k $\forall k \in \mathcal{K}_i \subset \mathcal{K}$, which it prefers to not share with any of its neighbors due perhaps to security and privacy reasons. Also suppose $\cup_{i\in\mathcal{V}}\mathcal{K}_i = \mathcal{K}$, so that every row of A and Y is known to at least one node. Notice that for each $i \in \mathcal{V}$, the set $\mathcal{K}_i$ may be empty so that node i knows nothing about A and Y, or it may contain multiple elements so that node i knows multiple rows of A and Y.

Given the above, the goal of this chapter is to design a distributed algorithm that enables the N nodes to cooperatively find a solution x to (3.2), or determine that no solution exists.

3.3 Algorithm Design

In this section, we design a distributed algorithm that has the aforementioned features.

Reconsider the graph $\mathcal{G}$ and let us focus on a specific node $i \in \mathcal{V}$. Recall that node i knows a_k and y_k $\forall k \in \mathcal{K}_i$. Suppose we associate with node i a vector $x_i(t) \in \mathbb{R}^n$, which represents its estimate of the solution of (3.2) at time $t \in [0, \infty)$. Although node i does not know the entire matrix A and vector Y, it can "do its part" by forcing $x_i(t)$ to gradually satisfy

$$a_k^T x_i(t) = y_k, \quad \forall k \in \mathcal{K}_i, \tag{3.3}$$

i.e., the portion of A and Y that it knows. One way to satisfy (3.3) is to consider a Lyapunov-like function $V : \mathbb{R}^n \to \mathbb{R}$, defined as

$$V(x_i(t)) = \frac{1}{2}\sum_{k\in\mathcal{K}_i}(a_k^T x_i(t) - y_k)^2. \tag{3.4}$$

In general, V in (3.4) is not guaranteed to be positive definite and, thus, may not be a valid Lyapunov function. However, V does represent how far away

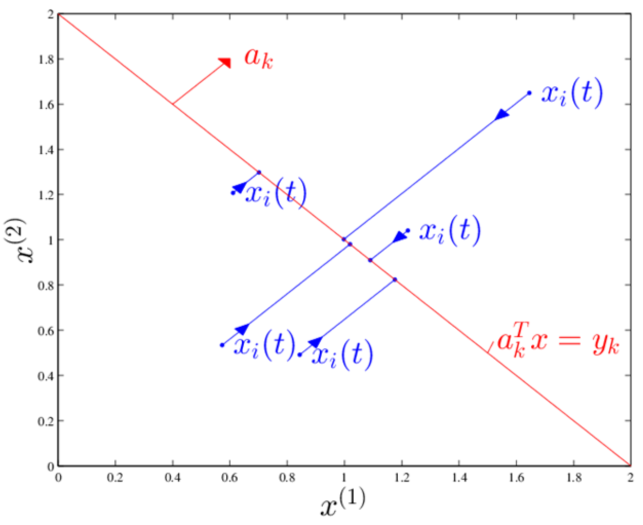


Figure 3.1: Illustration of the idea behind algorithm (3.5).

$x_i(t)$ is from satisfying (3.3). Thus, if node i updates $x_i(t)$ in such a way that $V(x_i(t))$ asymptotically decreases to zero, $x_i(t)$ would asymptotically satisfy (3.3). Motivated by this observation, let us take the time derivative of $V(x_i(t))$ along the trajectory $x_i(t)$:

$$\dot{V}(x_i(t)) = \sum_{k \in \mathcal{K}_i} (a_k^T x_i(t) - y_k) a_k^T \dot{x}_i(t).$$

To make $\dot{V}(x_i(t)) \leq 0$, a simple choice is to let

$$\dot{x}_i(t) = -\alpha_i \sum_{k \in \mathcal{K}_i} (a_k^T x_i(t) - y_k) a_k, \tag{3.5}$$

where $\alpha_i > 0$ is a design parameter. With (3.5), $x_i(t)$ is guaranteed to move in a direction where $V(x_i(t))$ decreases or stays the same, as illustrated in Figure 3.1 when $n = 2$ and $|\mathcal{K}_i| = 1$.

Since the goal is for the N nodes to cooperatively find a solution to (3.2), and since every row of A and Y is known to at least one node, if we force $x_i(t)$ of every node $i \in \mathcal{V}$ to not only asymptotically satisfy (3.3), but also achieve a consensus, the consensus value would be a solution to (3.2). In view of this and the basic idea from continuous-time distributed consensus [1,7], we add to (3.5) a "consensus" term to arrive at a continuous-time distributed algorithm

$$\dot{x}_i(t) = -\alpha_i \sum_{k\in\mathcal{K}_i} (a_k^T x_i(t) - y_k)a_k - \sum_{j\in\mathcal{N}_i} \beta_{\{i,j\}}(x_i(t) - x_j(t)), \quad \forall i \in \mathcal{V},\ \forall t \in [0,\infty), \tag{3.6}$$

where $\beta_{\{i,j\}} > 0\ \forall\{i,j\} \in \mathcal{E}$ are also design parameters.

To facilitate its analysis in the next section, note that algorithm (3.6) can be expressed in a matrix form as follows:

$$\dot{\mathbf{x}}(t) = -(\mathbf{P} + \mathbf{L})\mathbf{x}(t) + \mathbf{q}, \tag{3.7}$$

where $\mathbf{x}(t) \in \mathbb{R}^{nN}$ is a column vector formed by stacking the N $x_i(t)$'s, while $\mathbf{P} \in \mathbb{R}^{nN\times nN}$, $\mathbf{L} \in \mathbb{R}^{nN\times nN}$, and $\mathbf{q} \in \mathbb{R}^{nN}$ are given by

$$\mathbf{P} = \begin{bmatrix} \alpha_1 \sum_{k\in\mathcal{K}_1} a_k a_k^T & & & 0 \\ & \alpha_2 \sum_{k\in\mathcal{K}_2} a_k a_k^T & & \\ & & \ddots & \\ 0 & & & \alpha_N \sum_{k\in\mathcal{K}_N} a_k a_k^T \end{bmatrix},$$

$$\mathbf{L} = L_\beta \otimes I_n, \qquad \mathbf{q} = \begin{bmatrix} \alpha_1 \sum_{k\in\mathcal{K}_1} y_k a_k \\ \alpha_2 \sum_{k\in\mathcal{K}_2} y_k a_k \\ \vdots \\ \alpha_N \sum_{k\in\mathcal{K}_N} y_k a_k \end{bmatrix},$$

where $\otimes$ denotes the Kronecker product, $I_p \in \mathbb{R}^{p\times p}$ denotes the identity matrix, and $L_\beta = [L_{ij}] \in \mathbb{R}^{N\times N}$ is a weighted Laplacian matrix of $\mathcal{G}$ with $L_{ii} = \sum_{j\in\mathcal{N}_i} \beta_{\{i,j\}}$, $L_{ij} = -\beta_{\{i,j\}}$ if $\{i,j\} \in \mathcal{E}$, and $L_{ij} = 0$ if $i \neq j$ and $\{i,j\} \notin \mathcal{E}$.

3.4 Algorithm Analysis

In this section, we show that algorithm (3.6) or equivalently (3.7) has several appealing properties, which are reflected in three main results. First, we show that regardless of its initial condition $\mathbf{x}(0)$, the state $\mathbf{x}(t)$ is guaranteed to converge exponentially fast to a point $\mathbf{x}^*$ that depends on $\mathbf{x}(0)$ as well as the graph $\mathcal{G}$ and problem (3.2). Second, we show that when the solution set of (3.2) is not empty, all the $x_i(t)$'s are guaranteed to converge exponentially fast to the same point x^* in the solution set. Finally, we show that when the solution set is empty, at least one node in the graph $\mathcal{G}$ is able to asymptotically detect that.

To present the first main result, let

$$\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^{nN} : (\mathbf{P} + \mathbf{L})\mathbf{x} = \mathbf{q}\} \subset \mathbb{R}^{nN}$$

be the set of equilibrium points of (3.7). In addition, since $\mathbf{P} + \mathbf{L}$ is symmetric positive semidefinite, let its nN real eigenvalues be denoted as

$$0 = \lambda_1 = \lambda_2 = \cdots = \lambda_r < \lambda_{r+1} \leq \lambda_{r+2} \leq \cdots \leq \lambda_{nN},$$

where $0 \leq r \leq nN$, and its corresponding nN orthogonal eigenvectors be denoted as $u_1, u_2, \ldots, u_{nN} \in \mathbb{R}^{nN}$. Moreover, let

$$\Lambda = \operatorname{diag}(\lambda_{r+1}, \lambda_{r+2}, \ldots, \lambda_{nN}) \in \mathbb{R}^{(nN-r)\times(nN-r)},$$
$$\mathbf{U} = \begin{bmatrix} u_1 & u_2 & \cdots & u_{nN} \end{bmatrix} \in \mathbb{R}^{nN \times nN}.$$

Furthermore, let $\|\cdot\|$ denote the Euclidean norm and both $0_p \in \mathbb{R}^{p\times p}$ and $0_{p\times q} \in \mathbb{R}^{p\times q}$ be the all-zero matrices, which we will write as 0 whenever there is no confusion in their sizes.

With these notations, the first main result can be stated as follows:

Theorem 3.1. *For every $\mathbf{x}(0) \in \mathbb{R}^{nN}$, there exists a unique $\mathbf{x}^* \in \mathcal{S}$ such that*

$$\|\mathbf{x}(t) - \mathbf{x}^*\| \leq e^{-\lambda_{r+1}t}\|\mathbf{x}(0) - \mathbf{x}^*\|, \tag{3.8}$$

where $\mathbf{x}^*$ *is given by*

$$\mathbf{x}^* = \mathbf{U}\begin{bmatrix} I_r & 0 \\ 0 & 0_{nN-r} \end{bmatrix}\mathbf{U}^T\mathbf{x}(0) + \mathbf{U}\begin{bmatrix} 0_r & 0 \\ 0 & \Lambda^{-1} \end{bmatrix}\mathbf{U}^T\mathbf{q}. \tag{3.9}$$

Theorem 3.1 says that algorithm (3.7) is unconditionally exponentially convergent to a point that depends on the initial condition as well as the graph $\mathcal{G}$ and problem (3.2). In addition, as will be seen shortly, the theorem is instrumental in establishing a number of key properties of algorithm (3.7).

To prove Theorem 3.1, let $\mathbb{1}_N \in \mathbb{R}^N$ denote the all-one vector, $\mathcal{N}(M)$ denote the null space of any matrix M, and

$$\mathcal{X} = \{x \in \mathbb{R}^n : Ax = Y\} \subset \mathbb{R}^n$$

denote the solution set of (3.2). In addition, let

$$\tilde{\mathcal{S}} = \{\mathbb{1}_N \otimes x : x \in \mathcal{X}\} \subset \mathbb{R}^{nN},$$
$$\mathcal{S}_0 = \{\mathbf{x} : \mathbf{x} \in \mathcal{N}(\mathbf{P}+\mathbf{L})\} \subset \mathbb{R}^{nN},$$
$$\tilde{\mathcal{S}}_0 = \{\mathbb{1}_N \otimes x : x \in \mathcal{N}(A)\} \subset \mathbb{R}^{nN}.$$

Moreover, consider the following lemmas:

Lemma 3.1. $\mathcal{S}_0 = \tilde{\mathcal{S}}_0$.

Proof. First, we show that $\tilde{\mathcal{S}}_0 \subset \mathcal{S}_0$. Let $\mathbf{x} \in \tilde{\mathcal{S}}_0$. By definition of $\tilde{\mathcal{S}}_0$, we have $\mathbf{x} = \mathbb{1}_N \otimes x \in \mathbb{R}^{nN}$ for some $x \in \mathcal{N}(A)$. Since $x \in \mathcal{N}(A)$, $a_k^T x = 0\ \forall k \in \mathcal{K}$. Since $\mathcal{K}_i \subset \mathcal{K}\ \forall i \in \mathcal{V}$, $a_k^T x = 0\ \forall k \in \mathcal{K}_i\ \forall i \in \mathcal{V}$. Thus, by definition of $\mathbf{P}$,

$$\mathbf{P}\mathbf{x} = \begin{bmatrix} \alpha_1 \sum_{k\in\mathcal{K}_1} a_k a_k^T x \\ \alpha_2 \sum_{k\in\mathcal{K}_2} a_k a_k^T x \\ \vdots \\ \alpha_N \sum_{k\in\mathcal{K}_N} a_k a_k^T x \end{bmatrix} = 0.$$

Next, by definition of L_β, we have

$$\mathbf{L}\mathbf{x} = (L_\beta \otimes I_n)(\mathbb{1}_N \otimes x) = (L_\beta \mathbb{1}_N) \otimes (I_n x) = 0.$$

Hence, $(\mathbf{P}+\mathbf{L})\mathbf{x} = 0$, implying that $\mathbf{x} \in \mathcal{S}_0$, so that $\tilde{\mathcal{S}}_0 \subset \mathcal{S}_0$. Next, we show that $\mathcal{S}_0 \subset \tilde{\mathcal{S}}_0$. Let $\mathbf{x} \in \mathcal{S}_0$. Then, $(\mathbf{P}+\mathbf{L})\mathbf{x} = 0$ and $\mathbf{x}^T(\mathbf{P}+\mathbf{L})\mathbf{x} = 0$. Since both $\mathbf{P}$ and $\mathbf{L}$ are positive semidefinite, $\mathbf{x}$ satisfies $\mathbf{x}^T\mathbf{P}\mathbf{x} = 0$ and $\mathbf{x}^T\mathbf{L}\mathbf{x} = 0$. Since $\mathcal{G}$ is connected and $\mathbf{L} = L_\beta \otimes I_n$, we have $\mathbf{x} = \mathbb{1}_N \otimes x$ for some $x \in \mathbb{R}^n$. It follows that

$$\mathbf{x}^T\mathbf{P}\mathbf{x} = \sum_{i \in \mathcal{V}} \sum_{k \in \mathcal{K}_i} x^T(a_k a_k^T)x = 0.$$

Thus, $a_k^T x = 0\ \forall k \in \mathcal{K}_i\ \forall i \in \mathcal{V}$. Since $\cup_{i\in\mathcal{V}}\mathcal{K}_i = \mathcal{K}$, $a_k^T x = 0\ \forall k \in \mathcal{K}$. Hence, $x \in \mathcal{N}(A)$, implying that $\mathbf{x} \in \tilde{\mathcal{S}}_0$, so that $\mathcal{S}_0 \subset \tilde{\mathcal{S}}_0$. □

Lemma 3.2. *$\mathcal{S} \neq \emptyset$.*

Proof. First, we show that $\mathbf{q} \perp \tilde{\mathcal{S}}_0$. Let $\mathbf{x} \in \tilde{\mathcal{S}}_0$, so that $\mathbf{x} = \mathbb{1}_N \otimes x$ for some $x \in \mathcal{N}(A)$. Then, $a_k^T x = 0\ \forall k \in \mathcal{K}_i\ \forall i \in \mathcal{V}$. As a result,

$$\begin{aligned}\mathbf{x}^T\mathbf{q} &= \begin{bmatrix} x^T & x^T & \cdots & x^T \end{bmatrix} \begin{bmatrix} \alpha_1 \sum_{k\in\mathcal{K}_1} y_k a_k \\ \alpha_2 \sum_{k\in\mathcal{K}_2} y_k a_k \\ \vdots \\ \alpha_N \sum_{k\in\mathcal{K}_N} y_k a_k \end{bmatrix} \\ &= \sum_{i\in\mathcal{V}} \Big(\alpha_i \sum_{k\in\mathcal{K}_i} y_k a_k^T x \Big) = 0.\end{aligned}$$

Thus, $\mathbf{q} \perp \tilde{\mathcal{S}}_0$. By Lemma 3.1, $\mathbf{q} \perp \mathcal{S}_0$, i.e., $\mathbf{q} \perp \mathcal{N}(\mathbf{P}+\mathbf{L})$. Since $\mathbf{P}+\mathbf{L}$ is symmetric, its null space is orthogonal to its range space. Hence, $\mathbf{q}$ is in the range space of $\mathbf{P}+\mathbf{L}$, so that $\mathcal{S} \neq \emptyset$. □

With the above lemmas in hand, we now prove Theorem 3.1:

Proof of Theorem 3.1. Let $\mathbf{x}(0) \in \mathbb{R}^{nN}$ be given. Since $\mathcal{S} \neq \emptyset$ by Lemma 3.2, we can pick an $\tilde{\mathbf{x}} \in \mathcal{S}$. Let $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \tilde{\mathbf{x}}$, $\mathbf{z}(t) = \mathbf{U}^T\bar{\mathbf{x}}(t)$, and $\mathbf{z}(t) = (z_1(t), z_2(t), \ldots, z_N(t))$ where $z_i(t) \in \mathbb{R}^n$ $\forall i \in \mathcal{V}$. Then, (3.7) can be written as

$$\dot{\mathbf{z}}(t) = -\begin{bmatrix} 0_r & 0 \\ 0 & \Lambda \end{bmatrix} \mathbf{z}(t),$$

whose solution is

$$\mathbf{z}(t) = \begin{bmatrix} I_r & 0 \\ 0 & e^{-\Lambda t} \end{bmatrix} \mathbf{z}(0).$$

Let $\mathbf{z}^* = (z_1(0), z_2(0), \ldots, z_r(0), 0, 0, \ldots, 0) \in \mathbb{R}^{nN}$. Then,

$$\|\mathbf{z}(t) - \mathbf{z}^*\| \leq e^{-\lambda_{r+1}t}\|\mathbf{z}(0) - \mathbf{z}^*\|. \tag{3.10}$$

Let $\bar{\mathbf{x}}^* = \mathbf{U}\mathbf{z}^*$ and $\mathbf{x}^* = \tilde{\mathbf{x}} + \bar{\mathbf{x}}^*$. To show that $\mathbf{x}^* \in \mathcal{S}$, note that the first r columns of $\mathbf{U}$ are the r eigenvectors associated with the eigenvalue 0 of $(\mathbf{P}+\mathbf{L})$. Thus, the first r columns of $(\mathbf{P}+\mathbf{L})\mathbf{U}$ are zero, so that $(\mathbf{P}+\mathbf{L})\mathbf{U}\mathbf{z}^* = 0$. It follows that $\bar{\mathbf{x}}^* = \mathbf{U}\mathbf{z}^* \in \mathcal{S}_0$. Since $\tilde{\mathbf{x}} \in \mathcal{S}$, we have $\mathbf{x}^* = \tilde{\mathbf{x}} + \bar{\mathbf{x}}^* \in \mathcal{S}$. Due to $\mathbf{U}$ being orthogonal and due to (3.10), we obtain (3.8). Clearly, such an $\mathbf{x}^*$ is unique since (3.8) cannot be satisfied by two distinct $\mathbf{x}^*$.

Next, we show that $\mathbf{x}^*$ is given by (3.9). Since $\mathbf{z}(t) = \mathbf{U}^T\bar{\mathbf{x}}(t)$ and $\bar{\mathbf{x}}(t) = \mathbf{x}(t) - \tilde{\mathbf{x}}$, by definition of $\mathbf{z}^*$ we have

$$\begin{aligned}
\mathbf{x}^* &= \tilde{\mathbf{x}} + \bar{\mathbf{x}}^* \\
&= \tilde{\mathbf{x}} + \mathbf{U}\mathbf{z}^* \\
&= \tilde{\mathbf{x}} + \mathbf{U}\begin{bmatrix} I_r & 0 \\ 0 & 0_{nN-r} \end{bmatrix}\mathbf{U}^T(\mathbf{x}(0) - \tilde{\mathbf{x}}) \\
&= \mathbf{U}\begin{bmatrix} I_r & 0 \\ 0 & 0_{nN-r} \end{bmatrix}\mathbf{U}^T\mathbf{x}(0) + \mathbf{U}\begin{bmatrix} 0_r & 0 \\ 0 & I_{nN-r} \end{bmatrix}\mathbf{U}^T\tilde{\mathbf{x}}.
\end{aligned} \tag{3.11}$$

Since $\tilde{\mathbf{x}} \in \mathcal{S}$ and

$$\mathbf{P} + \mathbf{L} = \mathbf{U}\begin{bmatrix} 0_r & 0 \\ 0 & \Lambda \end{bmatrix}\mathbf{U}^T,$$

we have

$$\mathbf{q} = (\mathbf{P} + \mathbf{L})\tilde{\mathbf{x}} = \mathbf{U} \begin{bmatrix} 0_r & 0 \\ 0 & \Lambda \end{bmatrix} \mathbf{U}^T \tilde{\mathbf{x}}.$$

Pre-multiplying both sides by

$$\mathbf{U} \begin{bmatrix} 0_r & 0 \\ 0 & \Lambda^{-1} \end{bmatrix} \mathbf{U}^T,$$

we get

$$\mathbf{U} \begin{bmatrix} 0_r & 0 \\ 0 & \Lambda^{-1} \end{bmatrix} \mathbf{U}^T \mathbf{q} = \mathbf{U} \begin{bmatrix} 0_r & 0 \\ 0 & I_{nN-r} \end{bmatrix} \mathbf{U}^T \tilde{\mathbf{x}}.$$

Substituting the above into (3.11), we obtain (3.9). □

To establish the second main result, consider the following lemma:

Lemma 3.3. *If $\mathcal{X} \neq \emptyset$, then $\mathcal{S} = \tilde{\mathcal{S}}$.*

Proof. Suppose $\mathcal{X} \neq \emptyset$. Pick an $\tilde{x} \in \mathcal{X}$ and let $\tilde{\mathbf{x}} = \mathbb{1}_N \otimes \tilde{x}$. By definition of L_β,

$$\mathbf{L}\tilde{\mathbf{x}} = (L_\beta \otimes I_n)(\mathbb{1}_N \otimes \tilde{x}) = (L_\beta \mathbb{1}_N) \otimes (I_n \tilde{x}) = 0.$$

Thus, we have

$$(\mathbf{P} + \mathbf{L})\tilde{\mathbf{x}} - \mathbf{q} = \begin{bmatrix} \alpha_1 \sum_{k \in \mathcal{K}_1} a_k (a_k^T \tilde{x} - y_k) \\ \alpha_2 \sum_{k \in \mathcal{K}_2} a_k (a_k^T \tilde{x} - y_k) \\ \vdots \\ \alpha_N \sum_{k \in \mathcal{K}_N} a_k (a_k^T \tilde{x} - y_k) \end{bmatrix}.$$

Since $\tilde{x} \in \mathcal{X}$, $a_k^T \tilde{x} - y_k = 0 \; \forall k \in \mathcal{K}_i \; \forall i \in \mathcal{V}$. Hence, $(\mathbf{P}+\mathbf{L})\tilde{\mathbf{x}} = \mathbf{q}$, so that $\tilde{\mathbf{x}} \in \mathcal{S}$. Note that $\mathcal{S}$ is the set of all solutions to a system of linear equations, $\tilde{\mathbf{x}}$ is a particular solution, and $\mathcal{S}_0$ is the set of all homogeneous solutions. Therefore,

$$\mathcal{S} = \{\tilde{\mathbf{x}} + \mathbf{v} : \mathbf{v} \in \mathcal{S}_0\}. \tag{3.12}$$

Similarly, $\mathcal{X}$ is the set of all solutions to another system of linear equations (i.e., to (3.2) to be precise), $\tilde{x}$ is a particular solution, and $\mathcal{N}(A)$ is the set of all homogeneous solutions. Thus,

$$\mathcal{X} = \{\tilde{x} + v : v \in \mathcal{N}(A)\}. \tag{3.13}$$

Applying $\mathcal{S}_0 = \tilde{\mathcal{S}}_0$ from Lemma 3.1 to (3.12), we have

$$\begin{aligned} \mathcal{S} &= \{\tilde{\mathbf{x}} + \mathbf{v} : \mathbf{v} \in \tilde{\mathcal{S}}_0\} \\ &= \{(\mathbb{1}_N \otimes \tilde{x}) + (\mathbb{1}_N \otimes v) : v \in \mathcal{N}(A)\} \\ &= \{\mathbb{1}_N \otimes (\tilde{x} + v) : v \in \mathcal{N}(A)\}. \end{aligned} \tag{3.14}$$

Applying (3.13) to (3.14), we have

$$\begin{aligned} \mathcal{S} &= \{\mathbb{1}_N \otimes (\tilde{x} + v) : v \in \mathcal{N}(A)\} \\ &= \{\mathbb{1}_N \otimes (\tilde{x} + v) : \tilde{x} + v \in \mathcal{X}\} \\ &= \tilde{\mathcal{S}}, \end{aligned}$$

as desired. $\square$

With Lemma 3.3, we can now state the second main result:

Theorem 3.2. *Let $\mathbf{x}(0) \in \mathbb{R}^{nN}$ be given and let $\mathbf{x}^* \in \mathcal{S}$ be the limit of $\mathbf{x}(t)$ from Theorem 3.1. If $\mathcal{X} \neq \emptyset$, then $\mathbf{x}^* = \mathbb{1}_N \otimes x^*$ for some $x^* \in \mathcal{X}$. In addition,*

$$\|x_i(t) - x^*\| \leq e^{-\lambda_{r+1} t}\|\mathbf{x}(0) - \mathbf{x}^*\|, \quad \forall i \in \mathcal{V}. \tag{3.15}$$

Proof. Let $\mathbf{x}(0) \in \mathbb{R}^{nN}$ be given and suppose $\mathcal{X} \neq \emptyset$. By Theorem 3.1, there exists a unique $\mathbf{x}^* \in \mathcal{S}$ such that (3.8) holds. By Lemma 3.3, $\mathbf{x}^* \in \tilde{\mathcal{S}}$. Hence, $\mathbf{x}^* = \mathbb{1}_N \otimes x^*$ for some $x^* \in \mathcal{X}$. Because $\|x_i(t) - x^*\| \leq \|\mathbf{x}(t) - \mathbf{x}^*\|$ $\forall i \in \mathcal{V}$ and because of (3.8), (3.15) holds. $\square$

Theorem 3.2 shows that with algorithm (3.7), when (3.2) has one or more solutions, i.e., $\mathcal{X} \neq \emptyset$, all the estimates $x_i(t)$'s are guaranteed to converge exponentially fast to the same solution $x^* \in \mathcal{X}$. Obviously, this implies that when (3.2) has a unique solution x^*, all the $x_i(t)$'s would go to x^*.

Finally, we address the question of what would happen when (3.2) has no solution, i.e., $\mathcal{X} = \emptyset$. We have the following third main result:

Theorem 3.3. *Let $\mathbf{x}(0) \in \mathbb{R}^{nN}$ be given and let $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_N^*) \in \mathcal{S}$ be the limit of $\mathbf{x}(t)$ from Theorem 3.1. Then, $\mathcal{X} = \emptyset$ if and only if there exists $i \in \mathcal{V}$ such that condition (i) or (ii) below holds:*

(i) There exists $k \in \mathcal{K}_i$ such that $a_k^T x_i^* \neq y_k$.

(ii) There exists $j \in \mathcal{N}_i$ such that $x_i^* \neq x_j^*$.

Proof. ($\Rightarrow$) We show that the contrapositive is true. Suppose for every $i \in \mathcal{V}$, we have: (i') $a_k^T x_i^* = y_k$ $\forall k \in \mathcal{K}_i$; and (ii') $x_i^* = x_j^*$ $\forall j \in \mathcal{N}_i$. Because (ii') holds for each $i \in \mathcal{V}$ and because $\mathcal{G}$ is connected, we have $x_1^* = x_2^* = \cdots = x_N^* = x^*$ for some $x^* \in \mathbb{R}^n$. This, along with the fact that condition (i') holds for each $i \in \mathcal{V}$, implies that $a_k^T x^* = y_k$ $\forall k \in \mathcal{K}_i$ $\forall i \in \mathcal{V}$. Because $\cup_{i \in \mathcal{V}} \mathcal{K}_i = \mathcal{K}$ and because of (3.2), we have $x^* \in \mathcal{X}$, so that $\mathcal{X} \neq \emptyset$.
($\Leftarrow$) Again, we show that the contrapositive is true. Suppose $\mathcal{X} \neq \emptyset$. Then, by Theorem 3.2, $x_i^* = x^*$ $\forall i \in \mathcal{V}$ for some $x^* \in \mathcal{X}$. It follows that conditions (i) and (ii) are false for each $i \in \mathcal{V}$. $\square$

Observe that conditions (i) and (ii) can be checked locally by every node $i \in \mathcal{V}$. Thus, Theorem 3.3 says that when (3.2) has no solution, i.e., $\mathcal{X} = \emptyset$, at least one node in the graph $\mathcal{G}$ is able to asymptotically detect that.

3.5 Convergence Rate Analysis

In this section, we derive for a special case an explicit lower bound on the convergence rate of algorithm (3.7). We show that this lower bound depends on the problem (i.e., A), the graph (i.e., the algebraic connectivity of $\mathcal{G}$), and the algorithm parameters (i.e., the α_i's and $\beta_{\{i,j\}}$'s).

To begin, we define the special case as follows:

Assumption 3.1. Suppose: (i) $m = n$; (ii) $\mathcal{K}_i = \{i\}$ $\forall i \in \mathcal{V}$; (iii) A is nonsingular; and (iv) $\alpha_i = \alpha$ $\forall i \in \mathcal{V}$ and $\beta_{\{i,j\}} = \beta$ $\forall \{i,j\} \in \mathcal{E}$.

Note that Assumption 3.1 defines a special case where A is N-by-N and nonsingular, each node $i \in \mathcal{V}$ knows row i and only row i of A and Y, and the parameters α_i's and $\beta_{\{i,j\}}$'s of algorithm (3.7) are identical over graph $\mathcal{G}$. For this special case, we have the following lemma:

Lemma 3.4. *With Assumption 3.1,* $\mathbf{P} + \mathbf{L}$ *is positive definite.*

Proof. Since A is nonsingular, (3.2) has a unique solution, so that $\mathcal{X}$ has exactly one element. This implies that $\tilde{\mathcal{S}}$ also has exactly one element. By Lemma 3.3, so does $\mathcal{S}$. Thus, by definition of $\mathcal{S}$, $\mathbf{P} + \mathbf{L}$ is nonsingular. Since $\mathbf{P} + \mathbf{L}$ is symmetric positive semidefinite, this means that it is actually positive definite. $\square$

Recall from Chapter 3.4 that the N^2 eigenvalues of $\mathbf{P} + \mathbf{L}$ are denoted as $\lambda_1, \lambda_2, \ldots, \lambda_{N^2}$, which satisfy

$$0 = \lambda_1 = \lambda_2 = \cdots = \lambda_r < \lambda_{r+1} \leq \lambda_{r+2} \leq \cdots \leq \lambda_{N^2}.$$

By Lemma 3.4, we have $r = 0$, i.e., $\lambda_{r+1} = \lambda_1 > 0$. Observe from Theorems 3.1 and 3.2 that λ_1 characterizes the convergence rate of algorithm (3.7). Indeed, the larger λ_1, the faster the exponential convergence.

To derive a lower bound on λ_1 that algorithm (3.7) cannot converge slower than, consider the following notations: let $\lambda_{\min}(X) > 0$ be the smallest positive eigenvalue of any symmetric positive semidefinite matrix X that is not a zero matrix. In addition, let $\hat{A} \in \mathbb{R}^{N\times N}$ denote the row-normalized version of A, i.e.,

$$\hat{A} = \begin{bmatrix} — \frac{a_1^T}{\|a_1\|} — \\ — \frac{a_2^T}{\|a_2\|} — \\ \vdots \\ — \frac{a_N^T}{\|a_N\|} — \end{bmatrix}.$$

Moreover, let $\mu = \lambda_{\min}(\mathbf{P}) > 0$, $\gamma = \lambda_{\min}(\mathbf{L}) > 0$, and $\sigma > 0$ be the smallest singular value of $\hat{A}$.

Observe from the definition of $\mathbf{P}$ and from Assumption 3.1 that $\mathbf{P}$ is an N^2-by-N^2, block diagonal matrix where each of the N blocks is an N-by-N, rank-1 matrix. Thus, $\mathbf{P}$ has N positive eigenvalues at $\alpha\|a_i\|^2$ $\forall i \in \mathcal{V}$ and $N^2 - N$ eigenvalues at 0, so that $\mu = \lambda_{\min}(\mathbf{P}) = \alpha \min_{i\in\mathcal{V}}(\|a_i\|^2)$. In addition, observe from the definition of $\mathbf{L}$ and from Assumption 3.1 that $\mathbf{L} = L_\beta \otimes I_N = \beta L \otimes I_N$, where $L \in \mathbb{R}^{N\times N}$ is the standard Laplacian matrix of $\mathcal{G}$. Hence, $\gamma = \lambda_{\min}(\mathbf{L}) = \beta\lambda_{\min}(L)$, where $\lambda_{\min}(L)$ is the algebraic connectivity of $\mathcal{G}$.

Based on the above, an explicit lower bound λ^* on the convergence rate λ_1 of algorithm (3.7) can be stated as follows:

Theorem 3.4. *If Assumption 3.1 holds, then $\lambda_1 \geq \lambda^* > 0$, where*

$$\lambda^* = \frac{\mu + \gamma - \sqrt{(\mu+\gamma)^2 - 4\mu\gamma\sigma^2/N}}{2}. \tag{3.16}$$

In addition, $\lambda_1 = \lambda^*$ if and only if all the positive eigenvalues of $\mathbf{P}$ *are identical and all the positive eigenvalues of* $\mathbf{L}$ *are identical.*

Proof. See Chapter 3.5.1. □

To understand the implication of Theorem 3.4, notice that μ, γ, and σ are all positive. However, if any of them is near zero, their product $\mu\gamma\sigma$, which appears in (3.16), would be near zero. As a result, the lower bound λ^* in (3.16) would be near zero as well, suggesting that algorithm (3.7) may converge very slowly. Because $\mu = \alpha \min_{i\in\mathcal{V}}(\|a_i\|^2)$, $\gamma = \beta\lambda_{\min}(L)$, $\lambda_{\min}(L)$ is the algebraic connectivity of $\mathcal{G}$, and σ is the smallest singular value of $\hat{A}$, this implies that algorithm (3.7) may converge very slowly if one or more of the following conditions hold: (i) A is nearly singular, (ii) $\mathcal{G}$ is poorly connected, (iii) α is small, and (iv) β is small. On the contrary, if conditions (i)–(iv) do not hold—that is, A is far from being singular, $\mathcal{G}$ is well-connected, α is large, and β is large—then the lower bound λ^* in (3.16) would be large, suggesting that algorithm (3.7) would converge rapidly. Therefore, Theorem 3.4 offers meaningful insights into the performance of algorithm (3.7).

Figure 3.2 provides a contour plot that shows how the lower bound λ^* on the convergence rate λ_1 of algorithm (3.7) depends on the smallest singular value σ of $\hat{A}$ and the algebraic connectivity $\lambda_{\min}(L)$ of $\mathcal{G}$, when $(N, \alpha, \beta) = (100, 100, 1)$. Computed using Theorem 3.4, this contour plot demonstrates the interplay among algorithm performance, problem characteristics, and graph connectivity.

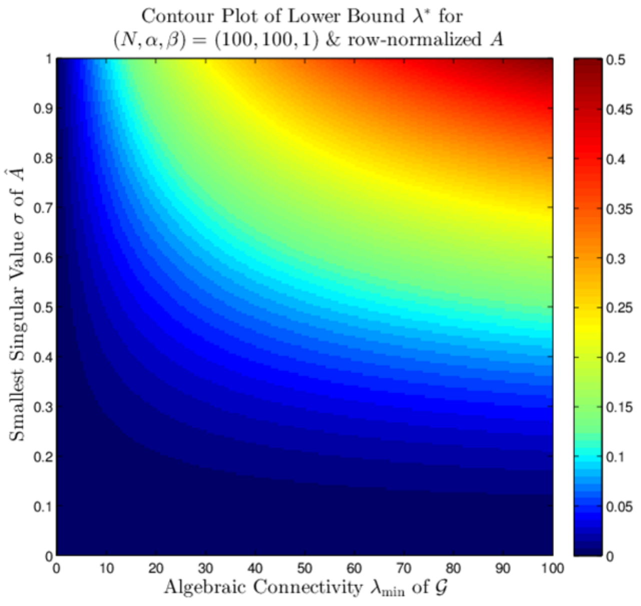


Figure 3.2: Contour plot showing how the lower bound λ^* on the convergence rate λ_1 of algorithm (3.7) depends on the smallest singular value σ of $\hat{A}$ and the algebraic connectivity $\lambda_{\min}(L)$ of $\mathcal{G}$, when $(N, \alpha, \beta) = (100, 100, 1)$.

3.5.1 Proof of Theorem 3.4

We first show that

$$(\mu + \gamma)^2 - 4\mu\gamma\sigma^2/N \geq 0, \tag{3.17}$$

so that λ^* in (3.16) is real. By Assumption 3.4, A is nonsingular. Thus, the a_k's are linear independent. It follows that $\hat{A}$ is also nonsingular, so that $\sigma > 0$. Therefore,

$$\sigma^2 = \lambda_{\min}(\hat{A}\hat{A}^T) \leq \frac{1}{N} \operatorname{tr}(\hat{A}\hat{A}^T) = 1.$$

Since $N \geq 1$,

$$(\mu + \gamma)^2 - 4\mu\gamma\sigma^2/N \geq (\mu + \gamma)^2 - 4\mu\gamma \geq 0,$$

thus establishing (3.17). Next, note from (3.16) that λ^* can be written as

$$\lambda^* = \frac{4\mu\gamma\sigma^2/N}{2(\mu + \gamma + \sqrt{(\mu + \gamma)^2 - 4\mu\gamma\sigma^2/N})}.$$

Due to (3.17) and $\sigma > 0$, we have $\lambda^* > 0$, confirming part of the claim of Theorem 3.4.

In the rest of the proof, we will show that $\lambda_1 \geq \lambda^*$. To do so, we first characterize the eigenvalues and eigenvectors of $\mathbf{P}$ and $\mathbf{L}$. For convenience, let $e_i \in \mathbb{R}^N$ $\forall i \in \mathcal{V}$ be standard basis vectors. We have the following lemma:

Lemma 3.5. *The N normalized eigenvectors corresponding to the N positive eigenvalues $\alpha\|a_1\|^2, \alpha\|a_2\|^2, \ldots, \alpha\|a_N\|^2$ of $\mathbf{P}$ are given by*

$$e_1 \otimes \frac{a_1}{\|a_1\|}, e_2 \otimes \frac{a_2}{\|a_2\|}, \ldots, e_N \otimes \frac{a_N}{\|a_N\|}.$$

Proof. By straightforward verification. □

Let $\mu_1, \mu_2, \ldots, \mu_{N^2} \in \mathbb{R}$ denote the N^2 eigenvalues of $\mathbf{P}$, $v_1, v_2, \ldots, v_{N^2} \in \mathbb{R}^{N^2}$ denote their corresponding normalized eigenvectors, $\gamma_1, \gamma_2, \ldots, \gamma_{N^2} \in \mathbb{R}$ denote the N^2 eigenvalues of $\mathbf{L}$, and $w_1, w_2, \ldots, w_{N^2} \in \mathbb{R}^{N^2}$ denote their corresponding normalized eigenvectors. Then, by Lemma 3.5,

$$\begin{aligned} &\mu_1 = \mu_2 = \cdots = \mu_{N^2-N} = 0, \\ &\mu_{N^2-N+i} = \alpha\|a_i\|^2, \quad \forall i \in \mathcal{V}, \\ &v_{N^2-N+i} = e_i \otimes \frac{a_i}{\|a_i\|}, \quad \forall i \in \mathcal{V}. \end{aligned} \tag{3.18}$$

In addition, since $\mathbf{L} = \beta L \otimes I_N$,

$$\begin{aligned} &\gamma_1 = \gamma_2 = \cdots = \gamma_N = 0, \\ &\gamma_{N+1}, \gamma_{N+2}, \ldots, \gamma_{N^2} > 0, \\ &w_i = \frac{1}{\sqrt{N}} \mathbb{1}_N \otimes e_i, \quad \forall i \in \mathcal{V}. \end{aligned} \tag{3.19}$$

Lemma 3.6. *The N^2 vectors $v_1, v_2, \ldots, v_{N^2-N}$ and $w_1, w_2, \ldots, w_N$ are linear independent and form a basis of $\mathbb{R}^{N^2}$.*

Proof. Suppose there exist $\eta_1, \eta_2, \ldots, \eta_{N^2-N} \in \mathbb{R}$ and $\rho_1, \rho_2, \ldots, \rho_N \in \mathbb{R}$, not all zero, such that

$$z_1 + z_2 = 0, \tag{3.20}$$

where

$$z_1 = \eta_1 v_1 + \eta_2 v_2 + \cdots + \eta_{N^2-N} v_{N^2-N},$$
$$z_2 = \rho_1 w_1 + \rho_2 w_2 + \cdots + \rho_N w_N.$$

Since z_1 is a linear combination of the eigenvectors associated with the eigenvalue 0 of $\mathbf{P}$, we have $\mathbf{P}z_1 = 0$. In the same fashion, $\mathbf{L}z_2 = 0$. Thus, from (3.20),

$$0 = (z_1 + z_2)^T(\mathbf{P} + \mathbf{L})(z_1 + z_2) = z_2^T \mathbf{P} z_2 + z_1^T \mathbf{L} z_1.$$

It follows that

$$z_1^T(\mathbf{P} + \mathbf{L})z_1 = 0.$$

Since $\mathbf{P}+\mathbf{L}$ is positive definite by Lemma 3.4, $z_1 = 0$. Due to (3.20), $z_2 = -z_1 = 0$. Since $v_1, v_2, \ldots, v_{N^2-N}$ are the eigenvectors of $\mathbf{P}$, they are linear independent. This, along with $z_1 = 0$, implies that $\eta_1 = \eta_2 = \cdots = \eta_{N^2-N} = 0$. Similarly, since $w_1, w_2, \ldots, w_N$ are the eigenvectors of $\mathbf{L}$, they are linear independent. This, together with $z_2 = 0$, implies that $\rho_1 = \rho_2 = \cdots = \rho_N = 0$, which contradicts the hypothesis. Hence, $v_1, v_2, \ldots, v_{N^2-N}$ and $w_1, w_2, \ldots, w_N$ are linear independent and form a basis of $\mathbb{R}^{N^2}$. □

Lemma 3.7. *For any $x \in \mathbb{R}^{N^2}$, $x^T(\mathbf{P} + \mathbf{L})x - \lambda^* x^T x \geq 0$.*

Proof. First, let us express $\mathbf{P}$ in a dyadic form using the μ_i's and v_i's in (3.18):

$$\mathbf{P} = \sum_{i=1}^{N^2} \mu_i v_i v_i^T = \sum_{i=N^2-N+1}^{N^2} \mu_i v_i v_i^T.$$

By definition of μ, we have

$$\mathbf{P} \geq \mu \sum_{i=N^2-N+1}^{N^2} v_i v_i^T = \mu\Big(I_{N^2} - \sum_{i=1}^{N^2-N} v_i v_i^T\Big). \tag{3.21}$$

Similarly, we have

$$\begin{aligned}\mathbf{L} &= \sum_{i=1}^{N^2} \gamma_i w_i w_i^T = \sum_{i=N+1}^{N^2} \gamma_i w_i w_i^T \\ &\geq \gamma \sum_{i=N+1}^{N^2} w_i w_i^T = \gamma\Big(I_{N^2} - \sum_{i=1}^{N} w_i w_i^T\Big). \end{aligned} \tag{3.22}$$

Next, let $x = x_1 + x_2$ where $x_1 = \sum_{i=1}^{N^2-N} p_i v_i$, $x_2 = \sum_{j=1}^{N} q_j w_j$, $p_i \in \mathbb{R}$ $\forall i \in \{1, 2, \ldots, N^2 - N\}$, and $q_j \in \mathbb{R}$ $\forall j \in \{1, 2, \ldots, N\}$. Using (3.18), (3.19), (3.21), (3.22), and the expression of x, we can write

$$\begin{aligned} x^T(\mathbf{P}+\mathbf{L})x &= x_2^T \mathbf{P} x_2 + x_1^T \mathbf{L} x_1 \\ &\geq \mu\Big(\sum_{j=1}^{N} q_j w_j\Big)^T \Big(I_{N^2} - \sum_{i=1}^{N^2-N} v_i v_i^T\Big)\Big(\sum_{j=1}^{N} q_j w_j\Big) \\ &\quad + \gamma\Big(\sum_{i=1}^{N^2-N} p_i v_i\Big)^T \Big(I_{N^2} - \sum_{j=1}^{N} w_j w_j^T\Big)\Big(\sum_{i=1}^{N^2-N} p_i v_i\Big) \\ &= y^T \begin{bmatrix} \gamma(I_{N^2-N} - \Phi^T\Phi) & 0 \\ 0 & \mu(I_N - \Phi\Phi^T) \end{bmatrix} y, \end{aligned} \tag{3.23}$$

where

$$y = \begin{bmatrix} p_1 & p_2 & \cdots & p_{N^2-N} & q_1 & q_2 & \cdots & q_N \end{bmatrix}^T \in \mathbb{R}^{N^2}, \tag{3.24}$$

$$\Phi = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_N^T \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_{N^2-N} \end{bmatrix} \in \mathbb{R}^{N\times(N^2-N)}. \tag{3.25}$$

Similarly, we can write

$$x^T x = \Big(\sum_{i=1}^{N^2-N} p_i v_i + \sum_{j=1}^{N} q_j w_j\Big)^T \Big(\sum_{i=1}^{N^2-N} p_i v_i + \sum_{j=1}^{N} q_j w_j\Big)$$

$$= y^T \begin{bmatrix} I_{N^2-N} & \Phi^T \\ \Phi & I_N \end{bmatrix} y. \tag{3.26}$$

Now, let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_\kappa > 0$ denote the singular values of Φ, where κ is the rank of Φ. In addition, let $\Sigma_\kappa = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_\kappa)$, so that the full singular value decomposition of Φ can be written as

$$\Phi = G\Sigma H^T, \tag{3.27}$$

where

$$\Sigma = \begin{bmatrix} \Sigma_\kappa & 0_{\kappa\times(N^2-N-\kappa)} \\ 0_{(N-\kappa)\times\kappa} & 0_{(N-\kappa)\times(N^2-N-\kappa)} \end{bmatrix} \in \mathbb{R}^{N\times(N^2-N)},$$

and $G \in \mathbb{R}^{N\times N}$ and $H \in \mathbb{R}^{(N^2-N)\times(N^2-N)}$ are orthogonal matrices. Furthermore, let

$$z = \begin{bmatrix} H & 0 \\ 0 & G \end{bmatrix} y.$$

This, along with (3.27), allows us to rewrite (3.23) and (3.26) as

$$y^T \begin{bmatrix} \gamma(I_{N^2-N} - \Phi^T\Phi) & 0 \\ 0 & \mu(I_N - \Phi\Phi^T) \end{bmatrix} y = z^T \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} z, \tag{3.28}$$

$$y^T \begin{bmatrix} I_{N^2-N} & \Phi^T \\ \Phi & I_N \end{bmatrix} y = z^T \begin{bmatrix} I_{N^2-N} & \Sigma^T \\ \Sigma & I_N \end{bmatrix} z, \tag{3.29}$$

where $D_1 \in \mathbb{R}^{(N^2-N)\times(N^2-N)}$ and $D_2 \in \mathbb{R}^{N\times N}$ are given by

$$D_1 = \gamma \operatorname{diag}(1-\sigma_1^2, 1-\sigma_2^2, \ldots, 1-\sigma_\kappa^2, 1, 1, \ldots, 1),$$

$$D_2 = \mu \operatorname{diag}(1-\sigma_1^2, 1-\sigma_2^2, \ldots, 1-\sigma_\kappa^2, 1, 1, \ldots, 1).$$

Next, let $\tilde{z} = \Pi z$, where $\Pi \in \mathbb{R}^{N^2\times N^2}$ is a permutation matrix such that (3.28) and (3.29) can be stated as

$$z^T \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} z = \tilde{z}^T \begin{bmatrix} C_1 & & & & 0 \\ & \ddots & & & \\ & & C_\kappa & & \\ & & & \gamma I_{N^2-N-\kappa} & \\ 0 & & & & \mu I_{N-\kappa} \end{bmatrix} \tilde{z}, \tag{3.30}$$

$$z^T \begin{bmatrix} I_{N^2-N} & \Sigma^T \\ \Sigma & I_N \end{bmatrix} z = \tilde{z}^T \begin{bmatrix} \tilde{C}_1 & & & & 0 \\ & \ddots & & & \\ & & \tilde{C}_\kappa & & \\ & & & I_{N^2-N-\kappa} & \\ 0 & & & & I_{N-\kappa} \end{bmatrix} \tilde{z}, \tag{3.31}$$

where $C_i \in \mathbb{R}^{2\times 2}$ and $\tilde{C}_i \in \mathbb{R}^{2\times 2}$ for each $i \in \{1, 2, \ldots, \kappa\}$ are given by

$$C_i = \begin{bmatrix} \gamma(1-\sigma_i^2) & 0 \\ 0 & \mu(1-\sigma_i^2) \end{bmatrix},$$
$$\tilde{C}_i = \begin{bmatrix} 1 & \sigma_i \\ \sigma_i & 1 \end{bmatrix}.$$

Equations (3.23), (3.26), (3.28), (3.29), (3.30), and (3.31) imply that

$$x^T(\mathbf{P}+\mathbf{L})x - \lambda^* x^T x$$
$$\geq \tilde{z}^T \begin{bmatrix} C_1 - \lambda^*\tilde{C}_1 & & & & 0 \\ & \ddots & & & \\ & & C_\kappa - \lambda^*\tilde{C}_\kappa & & \\ & & & (\gamma-\lambda^*)I_{N^2-N-\kappa} & \\ 0 & & & & (\mu-\lambda^*)I_{N-\kappa} \end{bmatrix} \tilde{z}. \tag{3.32}$$

To show that the right-hand side of (3.32) is nonnegative for any $x \in \mathbb{R}^{N^2}$ or equivalently for any $\tilde{z} \in \mathbb{R}^{N^2}$, it suffices to show that the following three conditions hold: (i) $\gamma \geq \lambda^*$, (ii) $\mu \geq \lambda^*$, and (iii) $C_i - \lambda^*\tilde{C}_i$ is positive semidefinite $\forall i \in \{1, 2, \ldots, \kappa\}$. Since $N \geq 1$ and $\sigma \leq 1$, we have

$$\lambda^* \leq \frac{\mu+\gamma-\sqrt{(\mu+\gamma)^2-4\mu\gamma}}{2} = \frac{\mu+\gamma-|\mu-\gamma|}{2} = \min\{\mu,\gamma\}.$$

Thus, conditions (i) and (ii) hold. To show that (iii) holds as well, notice from (3.25), (3.18), and (3.19) that

$$\Phi\Phi^T = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_N^T \end{bmatrix} \begin{bmatrix} v_1 & v_2 & \cdots & v_{N^2-N} \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_{N^2-N}^T \end{bmatrix} \begin{bmatrix} w_1 & w_2 & \cdots & w_N \end{bmatrix}$$

$$= \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_N^T \end{bmatrix} \Big(I_{N^2} - \sum_{i=N^2-N+1}^{N^2} v_i v_i^T \Big) \begin{bmatrix} w_1 & w_2 & \cdots & w_N \end{bmatrix}$$

$$= I_N - \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_N^T \end{bmatrix} \begin{bmatrix} v_{N^2-N+1} & v_{N^2-N+2} & \cdots & v_{N^2} \end{bmatrix} \begin{bmatrix} v_{N^2-N+1}^T \\ v_{N^2-N+2}^T \\ \vdots \\ v_{N^2}^T \end{bmatrix} \begin{bmatrix} w_1 & w_2 & \cdots & w_N \end{bmatrix}$$

$$= I_N - \frac{1}{N} \hat{A}^T \hat{A}. \tag{3.33}$$

By definition of σ and σ_1, and using (3.33), we have

$$\begin{aligned} \sigma_1^2 &= \lambda_{\max}(\Phi\Phi^T) = \lambda_{\max}(I_N - \frac{1}{N}\hat{A}^T\hat{A}) \\ &= 1 - \frac{1}{N}\sigma^2. \end{aligned} \tag{3.34}$$

Therefore, for all $i \in \{1, 2, \dots, \kappa\}$,

$$\begin{aligned} \lambda^* &= \frac{\mu + \gamma - \sqrt{(\mu+\gamma)^2 - 4\mu\gamma(1-\sigma_1^2)}}{2} \\ &\le \frac{\mu + \gamma - \sqrt{(\mu+\gamma)^2 - 4\mu\gamma(1-\sigma_i^2)}}{2} \\ &= \frac{4\mu\gamma(1-\sigma_i^2)}{2(\mu+\gamma+\sqrt{(\mu-\gamma)^2+4\mu\gamma\sigma_i^2})} \\ &\le \frac{2\mu\gamma(1-\sigma_i^2)}{\mu+\gamma+|\mu-\gamma|} \\ &= \min(\mu,\gamma)(1-\sigma_i^2). \end{aligned} \tag{3.35}$$

As a result, for all $i \in \{1, 2, \dots, \kappa\}$,

$$\begin{aligned} \det(C_i - \lambda^* \tilde{C}_i) &= (\gamma(1-\sigma_i^2) - \lambda^*)(\mu(1-\sigma_i^2) - \lambda^*) - \lambda^{*2}\sigma_i^2 \\ &= (1-\sigma_i^2)(\lambda^{*2} - (\mu+\gamma)\lambda^* + \mu\gamma(1-\sigma_i^2)) \ge 0. \end{aligned} \tag{3.36}$$

Combining (3.35) and (3.36), we see that condition (iii) holds. This establishes the lemma. □

By the Rayleigh quotient and Lemma 3.7, we have

$$\lambda_1 = \min_{x \neq 0} \frac{x^T(\mathbf{P}+\mathbf{L})x}{x^T x} \geq \lambda^*.$$

Finally, to prove the second part of Theorem 3.4, note that the inequality in (3.23) becomes an equality if and only if all the positive eigenvalues of $\mathbf{P}$ are identical and all the positive eigenvalues of $\mathbf{L}$ are also identical. Therefore, $\lambda_1 = \lambda^*$ if and only if the latter two conditions hold.

3.5.2 Why Convergence may be Slow

As was mentioned immediately after Theorem 3.4, algorithm (3.7) may converge slowly when A is nearly singular. In this subsection, we provide a simple example that explains graphically why its convergence may be slow. Consider an undirected and connected graph with $N = 2$ nodes, where A and Y are given by

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4.4 \end{bmatrix}, \quad Y = \begin{bmatrix} 3 \\ 6.4 \end{bmatrix},$$

row 1 of A and Y is known to node 1, and row 2 of A and Y is known to node 2. Suppose the nodes use algorithm (3.7) with $\alpha_1 = \alpha_2 = \beta_{\{1,2\}} = 1$. Then, we have

$$\begin{aligned} \dot{x}_1(t) &= -(a_1^T x_1(t) - y_1)a_1 - (x_1(t) - x_2(t)) \\ \dot{x}_2(t) &= -(a_2^T x_2(t) - y_2)a_2 - (x_2(t) - x_1(t)). \end{aligned} \tag{3.37}$$

Also suppose at some time t, $x_1(t)$ and $x_2(t)$ are as shown in Figure 3.3. Since A is nearly singular, the lines $a_1^T x = y_1$ and $a_2^T x = y_2$ are nearly parallel but still have a unique intersection which is the unique solution. Since $x_1(t)$ is close to the line $a_1^T x = y_1$, $x_2(t)$ is close to the line $a_2^T x = y_2$, and $x_1(t)$ and $x_2(t)$ are

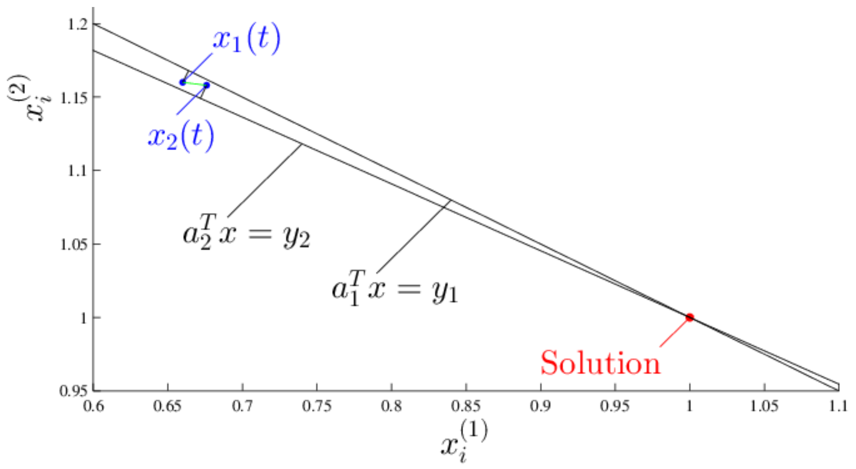


Figure 3.3: A simple example explaining why algorithm (3.7) may converge slowly when A is nearly singular.

close to each other, the right-hand side of (3.37) is small, causing both $\dot{x}_1(t)$ and $\dot{x}_2(t)$ to be small as well. As a result, both $x_1(t)$ and $x_2(t)$ would converge slowly to the unique solution. Note that if A is far from being singular, for which the lines $a_1^T x = y_1$ and $a_2^T x = y_2$ are far from being parallel, it would not have been possible for $x_1(t)$ and $x_2(t)$ to be simultaneously close to the lines $a_1^T x = y_1$ and $a_2^T x = y_2$ and close to each other—unless $x_1(t)$ and $x_2(t)$ are already close to the unique solution. This simple example explains why certain types of A may cause algorithm (3.7) to perform poorly.

3.6 Simulation Results

In this section, we present two sets of simulation results that demonstrate the effectiveness of algorithm (3.6) or (3.7).

3.6.1 Two 15-Node Graphs

Consider an undirected and connected graph $\mathcal{G}$ with $N = 15$ nodes, whose topology is shown in Figure 3.4(a). Suppose associated with this graph are a 15-by-15 matrix A and a 15-by-1 vector Y with randomly generated entries, such that A is nonsingular. Also suppose each node i knows row i of A and Y, and they wish to find the unique solution x of $Ax = Y$ using algorithm (3.7).

Figures 3.4(b) and 3.4(c) display the simulation result. Specifically, Figure 3.4(b) shows node 3's estimate $x_3^{(\ell)}(t)$ of the ℓth entry of the solution $x^{(\ell)}$ (calculated by Theorem 3.1) as a function of time t for $\ell \in \{1, 2, \ldots, 15\}$. Figure 3.4(c) shows node i's estimate $x_i^{(1)}(t)$ of the 1st entry of the solution $x^{(1)}$ for $i \in \{1, 2, \ldots, 15\}$. (Note that instead of including plots of $x_i^{(\ell)}(t)$ for every $i \in \{1, 2, \ldots, 15\}$ and every $\ell \in \{1, 2, \ldots, 15\}$, we included only two representative ones, in Figures 3.4(b) and 3.4(c).) Observe that despite having only local information on $\mathcal{G}$ and partial information on A and Y, the nodes are able to use algorithm (3.6) to asymptotically find the unique solution x^* of $Ax = Y$.

Next, consider another undirected and connected graph $\mathcal{G}$ with $N = 15$ nodes as shown in Figure 3.5(a). As before, suppose A is a 15-by-15 matrix and Y is a 15-by-1 vector with randomly generated entries. However, here A is singular and Y is *not* in the range space of A. Also suppose the nodes want to use algorithm (3.7) to find a solution x of $Ax = Y$, despite each node i knowing only row i of A and Y, and despite none of them knowing that a solution actually does not exist.

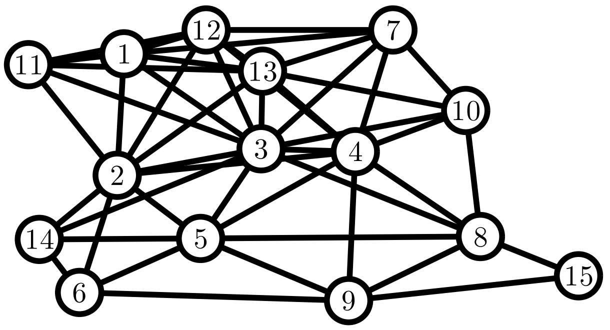
Figures 3.5(b) and 3.5(c) display the simulation result using a format that is identical to that of Figures 3.4(b) and 3.4(c). Observe that Figure 3.5(b) is qualitatively similar to Figure 3.4(b). However, Figure 3.5(c) is quite different from Figure 3.4(c) in that for the latter, all the $x_i^{(1)}(t)$'s converge to the same value $x^{(1)}$, whereas for the former, all the $x_i^{(1)}(t)$'s converge to different values. This implies that at least one node in the graph is able to asymptotically detect that a solution does not exist simply by comparing its $x_i^{(1)}(t)$ with its neighbor's $x_j^{(1)}(t)$.

3.6.2 A 5-Node Graph

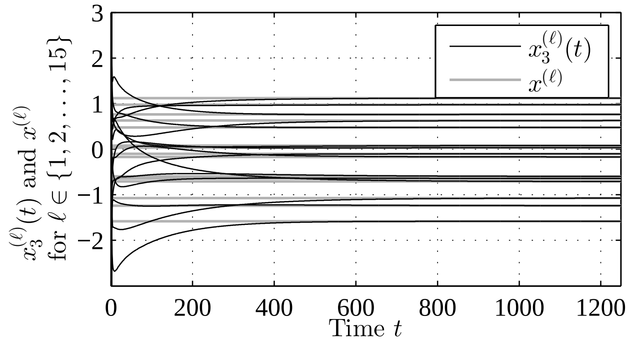
In this subsection, we present the second set of simulation results, for a graph $\mathcal{G}$ with $N = 5$ nodes, whose topology is shown in Figures 3.6(a), 3.7(a), and 3.8(a). Suppose associated with this graph are a 5-by-2 matrix A and a 5-by-1 vector Y, whose entries are shown in Figures 3.6(b), 3.7(b), and 3.8(b), which correspond to cases when $Ax = Y$ has no solution, a unique solution and infinitely many solutions, respectively. Also suppose each node i knows row i of A and Y, and they wish to agree on a solution or discover that no solution exists using algorithm (3.7). Figures 3.6(c), 3.7(c), 3.8(c), and 3.8(d) display the simulation result, from which we can see that the $x_i(t)$'s converge to different values when there is no solution, converge to the unique solution when there is exactly one, and converge to an initial-condition-dependent solution when there are infinitely many solutions.

3.7 Conclusion

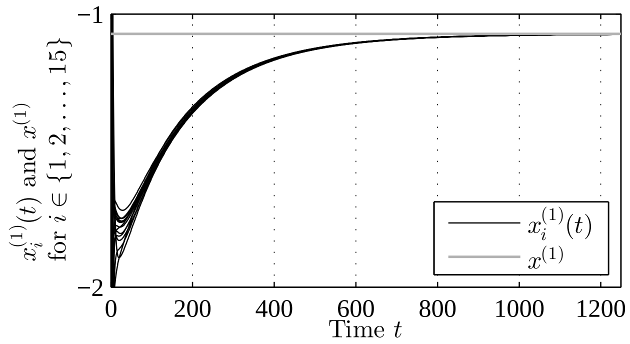
In this chapter, we have designed a continuous-time distributed algorithm, with which nodes in a network can cooperatively solve a general system of linear equations, whose data are scattered across the network. We have shown that the algorithm is able to asymptotically detect whether a solution exists, asymptotically find one when it does, and globally exponentially converge with a rate that can be explicitly bounded from below.



(a) A 15-node graph.

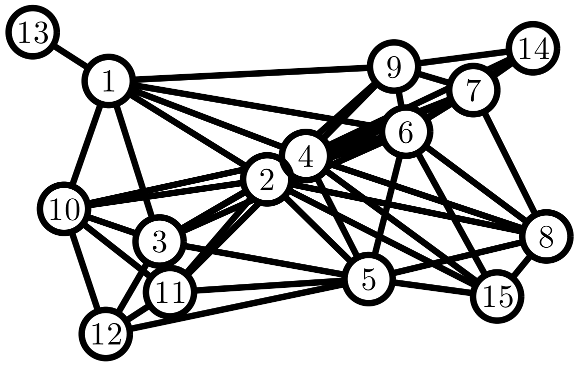


(b) Node 3's estimate $x_3^{(\ell)}(t)$ of the ℓth entry of the solution $x^{(\ell)}$ for $\ell \in \{1, 2, \ldots, 15\}$.

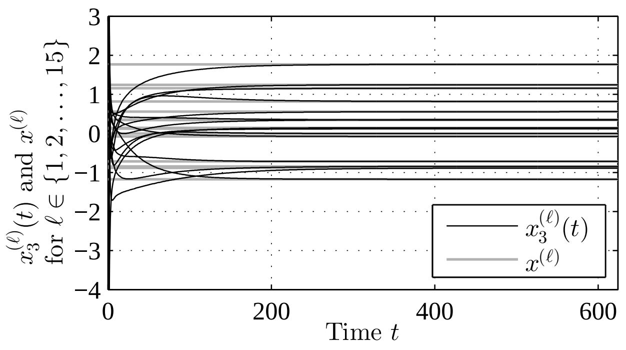


(c) Node i's estimate $x_i^{(1)}(t)$ of the 1st entry of the solution $x^{(1)}$ for $i \in \{1, 2, \ldots, 15\}$.

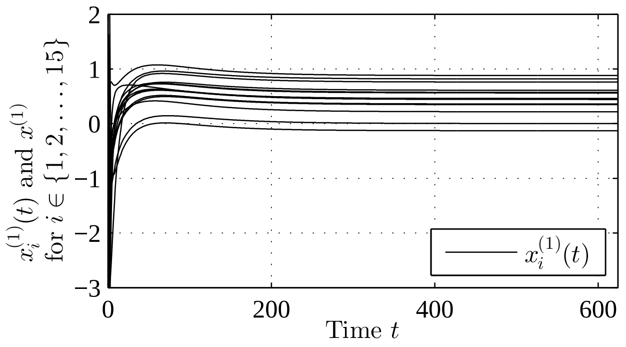
Figure 3.4: Performance of algorithm (3.6) when there is a unique solution.



(a) A 15-node graph.

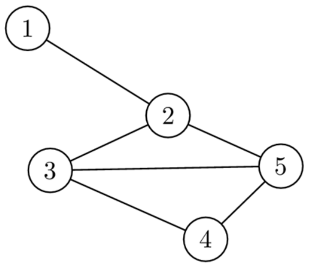


(b) Node 3's estimate $x_3^{(\ell)}(t)$ of the ℓth entry of the equilibrium $x^{(\ell)}$ from Theorem 3.1 for $\ell \in \{1,2,\ldots,15\}$.



(c) Node i's estimate $x_i^{(1)}(t)$ of the 1st entry of the equilibrium $x^{(1)}$ for $i \in \{1,2,\ldots,15\}$.

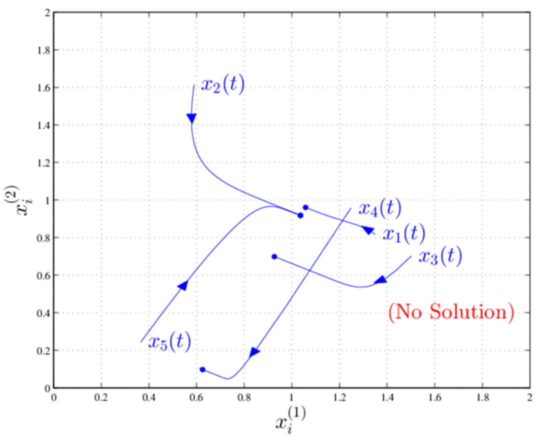
Figure 3.5: Performance of algorithm (3.6) when there is no solution.



(a) A 5-node graph.

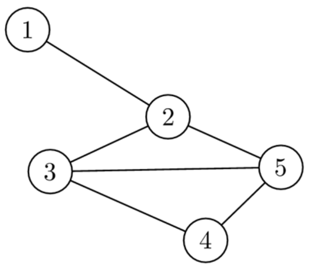
$$\begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \\ 4 & 8 \\ 2 & 4 \end{bmatrix} x = \begin{bmatrix} 3 \\ 6 \\ 7 \\ 3 \\ 6 \end{bmatrix}$$

(b) Linear equations where each node observes one row.



(c) Node i's estimate $x_i(t)$ of the solution (which does not exist).

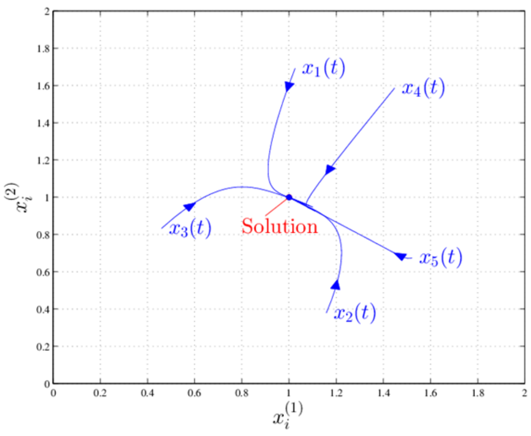
Figure 3.6: Performance of algorithm (3.6) when there is no solution.



(a) A 5-node graph.

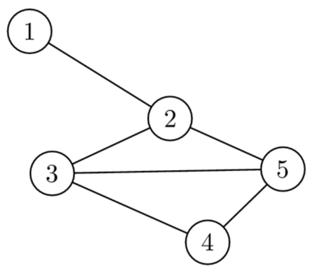
$$\begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 2 & 4 \\ 4 & 6 \\ 6 & 9 \end{bmatrix} x = \begin{bmatrix} 3 \\ 5 \\ 6 \\ 10 \\ 15 \end{bmatrix}$$

(b) Linear equations where each node observes one row.



(c) Node i's estimate $x_i(t)$ of the solution.

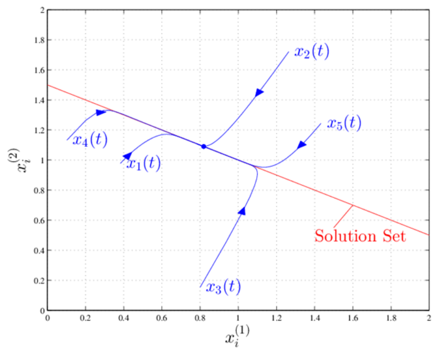
Figure 3.7: Performance of algorithm (3.6) when there is a unique solution.



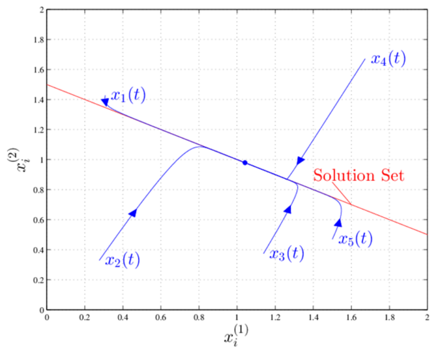
(a) A 5-node graph.

$$\begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \\ 4 & 8 \\ 2 & 4 \end{bmatrix} x = \begin{bmatrix} 3 \\ 6 \\ 9 \\ 12 \\ 6 \end{bmatrix}$$

(b) Linear equations where each node observes one row.



(c) Node i's estimate $x_i(t)$ of the solution.



(d) Node i's estimate $x_i(t)$ of the solution.

Figure 3.8: Performance of algorithm (3.6) when there are infinitely many solutions.

Chapter 4 Continuous-Time Distributed Computation of the Perron-Frobenius Eigenvector

4.1 Introduction

The Perron-Frobenius theorem has numerous applications [128]. For instance, the theorem plays a central role in the analysis of Markov chains and has been applied to distributed power control in wireless networks [129], Leontief's input-output model in commodity pricing [128], and population growth models [128], to name just a few. The Perron-Frobenius eigenvector, whose existence and uniqueness is guaranteed by the theorem, can be used to measure the relative importance of nodes in a graph in what is called their *eigenvector centrality*. Indeed, eigenvector centrality has been used by Google in its well-known PageRank algorithm to measure the relative importance of webpages [130] and by Leo Katz to measure the relative degree of influence of actors within a social network [121]. Basically, what eigenvector centrality does is it assigns a higher score to a node if the node happens to be connected to other high-scoring nodes, in a "circular" fashion like how eigenvectors of a matrix are defined. Figure 4.1 illustrates the concept of eigenvector centrality, in which the color of a node represents its centrality score, or how important it is, in the graph (e.g., red indicates a high score while blue indicates a low).

With the emergence of increasingly complex networks that often have to operate without a designated leader [121], it is becoming desirable that nodes

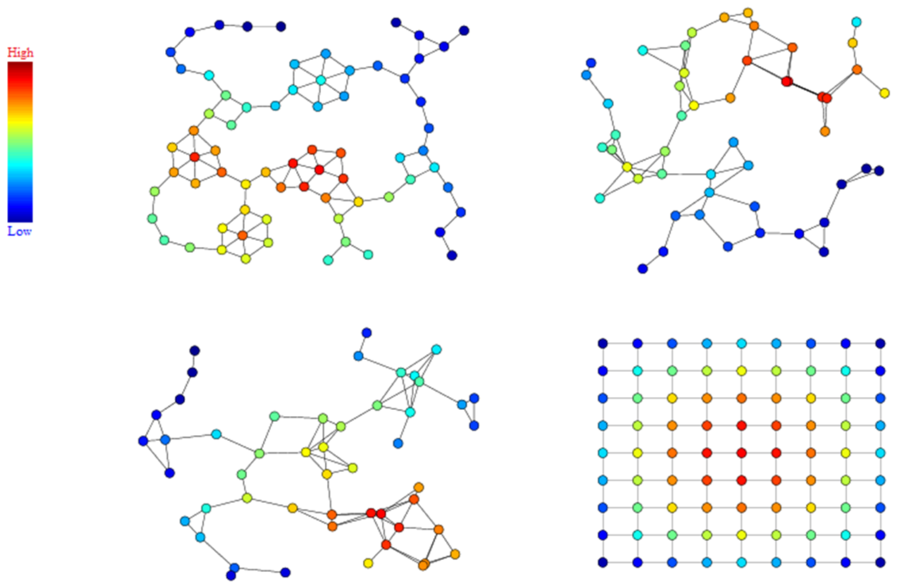


Figure 4.1: Examples illustrating the concept of eigenvector centrality.

in such a network can analyze the network themselves, such as decentralizedly computing the Perron-Frobenius eigenvector of a matrix associated with the network. Unfortunately however, most of the existing methods on computing such an eigenvector are centralized and based on the power method [130, 131]. That said, a few distributed algorithms for computing eigenvectors using random walk have been proposed in [74–76]. Other related work include [123–125] that consider distributed estimation of Laplacian eigenvalues, [77–79] that focus on distributed estimation of the second smallest Laplacian eigenvalue (i.e., the algebraic connectivity), and [132] that studies distributed estimation of arbitrary graph spectra.

In this chapter, we develop a class of continuous-time distributed algorithms, which enable each node i in an undirected and connected graph to compute the ith entry of the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue. The only assumption these algorithms need is that each node i knows row i of the matrix and who its neighbors are. We show that each continuous-time distributed algorithm in the class is a nonlinear networked dynamical system with a skew-symmetric structure, whose state is guaranteed to stay on a sphere and remain nonnegative at all times. Moreover, using LaSalle's invariance principle [133], we show that the state must converge asymptotically to said eigenvector, from which the corresponding eigenvalue can then be computed. Furthermore, we show that under some mild conditions, convergence at an $O(\frac{1}{t})$ rate can be achieved.

The outline of this chapter is as follows: Chapter 4.2 formulates the problem. Chapter 4.3 designs the class of continuous-time distributed algorithms. Chapter 4.4 analyzes their convergence behaviors and considers two special cases. Chapter 4.5 demonstrates their effectiveness via simulation. Finally, Chapter 4.6 provides some concluding remarks. Throughout the chapter, for any $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, we write $x \geq 0$ if $x_i \geq 0$ for all i; $x > 0$ if $x \geq 0$ and $x \neq 0$; and $x \gg 0$ if $x_i > 0$ for all i. In addition, we let $\mathbb{Z}_{\geq 0} = \{0, 1, \ldots\}$ denote the set of nonnegative integers.

4.2 Problem Formulation

Consider a network modeled as an undirected and connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ denotes the set of $N \geq 2$ nodes and $\mathcal{E} \subset \{\{i, j\} : i, j \in \mathcal{V}, i \neq j\}$ denotes the set of edges. Any two nodes $i, j \in \mathcal{V}$ are neighbors and can communicate if and only if $\{i, j\} \in \mathcal{E}$. The set of neighbors of each node $i \in \mathcal{V}$ is denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : \{i, j\} \in \mathcal{E}\}$, and the communications are assumed to be delay- and error-free, with no quantization.

Suppose associated with the graph $\mathcal{G}$ is a square matrix $W = [w_{ij}] \in \mathbb{R}^{N \times N}$ satisfying the following assumption:

Assumption 4.1. The matrix W is such that: (i) for each $i, j \in \mathcal{V}$ with $i \neq j$, if $\{i, j\} \notin \mathcal{E}$, then $w_{ij} = w_{ji} = 0$; (ii) for each $i, j \in \mathcal{V}$ with $i \neq j$, if $\{i, j\} \in \mathcal{E}$, then $w_{ij} = w_{ji} \geq 0$; and (iii) W is irreducible.

Notice that statements (i) and (ii) in Assumption 4.1 imply that W is symmetric, so that all its N eigenvalues are real. Moreover, these two statements require the off-diagonal entries of W to be nonnegative but place no restriction on its diagonal entries. Thus, W is a Metzler matrix [134]. Furthermore, the two statements allow w_{ij} and w_{ji} to be zero when $\{i, j\} \in \mathcal{E}$. Hence, although $\mathcal{G}$ is connected, they do not imply that W is irreducible [135]. Therefore, statement (iii) is not redundant. Lastly, note that W can be the adjacency matrix of graph $\mathcal{G}$, or any other matrix "induced" by $\mathcal{G}$ as long as Assumption 4.1 holds.

In addition to Assumption 4.1, the following slightly more restrictive assumption on W will be used in the latter part of the chapter:

Assumption 4.2. The matrix W is such that: (i) for each $i, j \in \mathcal{V}$ with $i \neq j$, if $\{i, j\} \notin \mathcal{E}$, then $w_{ij} = w_{ji} = 0$; and (ii) for each $i, j \in \mathcal{V}$ with $i \neq j$, if $\{i, j\} \in \mathcal{E}$, then $w_{ij} = w_{ji} > 0$.

Note that statement (i) in Assumption 4.2 is identical to that in Assumption 4.1, whereas statement (ii) in Assumption 4.2 is slightly more stringent than that in Assumption 4.1. Also note that statement (iii) in Assumption 4.1 is not needed in Assumption 4.2 because the latter and the connectedness of $\mathcal{G}$ imply that W is irreducible. Thus, Assumption 4.2 is mild and is only slightly stronger than Assumption 4.1. This also implies that results obtained under Assumption 4.1 are valid under Assumption 4.2, but not necessarily the other way around.

With Assumption 4.1, the following can be said about W:

Proposition 4.1. *The matrix W has the following properties: (i) the largest eigenvalue of W, denoted as λ^*, is simple; (ii) the eigenspace associated with λ^*, denoted as $\mathcal{X}^*$, is one-dimensional and is of the form $\mathcal{X}^* = \{\alpha x^* : \alpha \in \mathbb{R}, \alpha \neq 0\}$, where $x^* \in \mathbb{R}^N$ and $x^* \gg 0$; and (iii) if $x > 0$ is an eigenvector of W, then $x \in \mathcal{X}^*$.*

Proof. Since W is a Metzler matrix, there exists $\alpha \in \mathbb{R}$ such that $\tilde{W} \triangleq W + \alpha I$ is a nonnegative matrix. Since W is symmetric and irreducible, so is $\tilde{W}$. Thus, the Perron-Frobenius theorem is applicable to $\tilde{W}$. Next, observe that $\lambda \in \mathbb{R}$ is an eigenvalue of W if and only if $\lambda + \alpha$ is an eigenvalue of $\tilde{W}$. In addition, $x \in \mathbb{R}^N$ is an eigenvector of W if and only if it is an eigenvector of $\tilde{W}$. These two observations, along with the Perron-Frobenius theorem as applied to $\tilde{W}$, imply properties (i)–(iii). $\square$

Remark 4.1. If W is nonnegative as opposed to just being Metzler, λ^* in (i) and $x^* \in \mathcal{X}^*$ in (ii) of Proposition 4.1 would be the Perron-Frobenius eigenvalue and eigenvector of W, respectively.

Suppose each node $i \in \mathcal{V}$ knows only $\mathcal{N}_i$, w_{ii}, and w_{ij} $\forall j \in \mathcal{N}_i$, which it prefers to not share with any of its neighbors due perhaps to security and privacy reasons. Yet, despite having only such local information about graph $\mathcal{G}$ and matrix W, suppose every node $i \in \mathcal{V}$ wants to determine the largest eigenvalue λ^* of W and the ith entry x_i^* of an eigenvector x^* from the eigenspace $\mathcal{X}^*$.

Given the above, the goal is to devise a distributed algorithm that enables every node $i \in \mathcal{V}$ to asymptotically determine the aforementioned λ^* and x_i^*.

4.3 Design of Continuous-Time Algorithms

In this section, we develop a class of continuous-time distributed algorithms that achieve the stated goal.

First, let $t \geq 0$ denote time. In addition, suppose each node $i \in \mathcal{V}$ maintains in its memory a variable $x_i(t) \in \mathbb{R}$, which represents its estimate of the ith entry x_i^* of some eigenvector $x^* \in \mathcal{X}^*$ at time t, and another variable $y_i(t) \in \mathbb{R}$, which is defined as

$$y_i(t) = w_{ii}x_i(t) + \sum_{j \in \mathcal{N}_i} w_{ij}x_j(t). \tag{4.1}$$

Moreover, let

$$x(t) = (x_1(t), x_2(t), \ldots, x_N(t)) \in \mathbb{R}^N,$$
$$y(t) = (y_1(t), y_2(t), \ldots, y_N(t)) \in \mathbb{R}^N,$$

which we will sometimes write as $x = (x_1, x_2, \ldots, x_N)$ and $y = (y_1, y_2, \ldots, y_N)$ for convenience. Note that since each node $i \in \mathcal{V}$ knows $\mathcal{N}_i$, w_{ii}, and w_{ij} $\forall j \in \mathcal{N}_i$, it can determine $y_i(t)$ by querying every neighbor $j \in \mathcal{N}_i$ for $x_j(t)$. Furthermore, with the vector notation, (4.1) may be compactly written as

$$y = Wx. \tag{4.2}$$

Next, observe from Proposition 4.1 that for any $x^* \in \mathcal{X}^*$ and any $x \in \mathbb{R}^N$ with $\|x\| = \|x^*\|$, we have the tight bound

$$x^T W x \leq x^{*T} W x^* = \lambda^* \|x^*\|^2,$$

where $\|\cdot\|$ denotes the 2-norm. This observation suggests that, if the N nodes are able to distributively update their estimates $x(t)$ so that $\|x(t)\|$ remains constant $\forall t \geq 0$ while $x(t)^T W x(t)$ keeps increasing, then $x(t)$ might converge to an $x^* \in \mathcal{X}^*$. Motivated by this idea, suppose the N nodes update $x(t)$ using

$$\dot{x}(t) = Sx(t), \tag{4.3}$$

where $S = [s_{ij}] \in \mathbb{R}^{N\times N}$ is a skew-symmetric matrix, i.e., $S = -S^T$ or equivalently

$$s_{ii} = 0 \quad \text{and} \quad s_{ij} = -s_{ji}, \quad \forall i, j \in \mathcal{V}. \tag{4.4}$$

Since S is skew-symmetric, $z^T S z = 0$ for all $z \in \mathbb{R}^N$. This, together with (4.3), implies that

$$\frac{d}{dt}\|x(t)\|^2 = 2x(t)^T \dot{x}(t) = 2x(t)^T S x(t) = 0, \quad \forall t \geq 0.$$

Thus, $\|x(t)\|$ remains constant $\forall t \geq 0$, as desired. Because any two nodes $i, j \in \mathcal{V}$ can communicate if and only if $\{i, j\} \in \mathcal{E}$, for (4.3) to be distributively implementable, it is necessary and sufficient that

$$s_{ij} = s_{ji} = 0, \quad \forall \{i, j\} \notin \mathcal{E}. \tag{4.5}$$

Hence, in what follows, we will impose condition (4.5) on S, in addition to condition (4.4).

To ensure that $x(t)^T W x(t)$ keeps increasing, consider a quadratic function $J : \mathbb{R}^N \to \mathbb{R}$, defined as

$$J(x) = x^T W x. \tag{4.6}$$

Taking the time derivative of J in (4.6) along the solution $x(t)$ of (4.3) and using (4.2), (4.4), and (4.5), we obtain

$$\begin{aligned}\dot{J}(x(t)) &= 2x(t)^T W \dot{x}(t) = 2y(t)^T S x(t) \\ &= 2\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}} s_{ij} y_i(t) x_j(t) \\ &= 2\sum_{\{i,j\}\in\mathcal{E}} s_{ij}(y_i(t)x_j(t) - y_j(t)x_i(t)) &\text{(4.7a)} \\ &= 2\sum_{\{i,j\}\in\mathcal{E}} s_{ji}(y_j(t)x_i(t) - y_i(t)x_j(t)), &\text{(4.7b)}\end{aligned}$$

where (4.7a) and (4.7b) are equivalent due to (4.4), so that we can just look at, say, (4.7a). Note that if s_{ij} in (4.7a) is such that

$$s_{ij}(y_i(t)x_j(t) - y_j(t)x_i(t)) \geq 0, \quad \forall \{i,j\} \in \mathcal{E},\ \forall t \geq 0, \tag{4.8}$$

then $\dot{J}(x(t)) \geq 0$ $\forall t \geq 0$, so that $J(x(t)) = x(t)^T W x(t)$ in (4.6) increases monotonically, perhaps even approaches its maximum. A simple way to satisfy (4.8) is to let s_{ij} *not* be a constant but rather be a function of x, denoted as $s_{ij}(x)$ and defined as

$$s_{ij}(x) = y_i x_j - y_j x_i, \quad \forall \{i,j\} \in \mathcal{E}. \tag{4.9}$$

Notice from (4.1) or (4.2) that $s_{ij}(x)$ in (4.9) is indeed a function of x. In fact, $s_{ij}(x)$ is implementable by each pair of neighboring nodes i and j because they have access to $x_i(t)$, $x_j(t)$, $y_i(t)$, and $y_j(t)$ $\forall t \geq 0$. Moreover, with (4.9), not only that (4.8) holds, $J(x(t)) = x(t)^T W x(t)$ is strictly increasing whenever $y_i(t)x_j(t) - y_j(t)x_i(t)) \neq 0$ for at least one $\{i,j\} \in \mathcal{E}$.

Although (4.9) is a simple way to satisfy (4.8), more general choices are possible, such as letting

$$s_{ij}(x) = \phi_{\{i,j\}}(y_i x_j - y_j x_i), \quad \forall \{i,j\} \in \mathcal{E}, \tag{4.10}$$

where each function $\phi_{\{i,j\}} : \mathbb{R} \to \mathbb{R}$ is selected by neighboring nodes i and j and has the following properties:

P1. $\phi_{\{i,j\}}$ is locally Lipschitz.

P2. $\phi_{\{i,j\}}$ is inside the first and third quadrants, i.e., $z\phi_{\{i,j\}}(z) > 0$ $\forall z \in \mathbb{R}$, $z \neq 0$.

P3. $\phi_{\{i,j\}}$ is odd, i.e., $\phi_{\{i,j\}}(z) = -\phi_{\{i,j\}}(-z)$ $\forall z \in \mathbb{R}$.

Property P1, along with (4.1) and (4.10), ensures that the nonlinear dynamical system (4.3) has a unique solution. Property P2, together with (4.10), ensures that (4.8) holds and that $J(x(t)) = x(t)^T W x(t)$ is strictly increasing whenever $y_i(t)x_j(t) - y_j(t)x_i(t)) \neq 0$ for at least one $\{i,j\} \in \mathcal{E}$. Property P3, along with (4.10), ensures that

$$\begin{aligned} s_{ij}(x) &= \phi_{\{i,j\}}(y_i x_j - y_j x_i) \\ &= -\phi_{\{i,j\}}(y_j x_i - y_i x_j) = -s_{ji}(x), \quad \forall \{i,j\} \in \mathcal{E}, \end{aligned}$$

thereby preserving the skew-symmetricity of S.

Putting together the above development, we obtain a class of continuous-time distributed algorithms described by

$$\dot{x}(t) = S(x(t))x(t), \tag{4.11}$$

where $S(x(t)) = [s_{ij}(x(t))] \in \mathbb{R}^{N\times N}$ is given by

$$s_{ij}(x(t)) = \begin{cases} \phi_{\{i,j\}}(y_i(t)x_j(t) - y_j(t)x_i(t)), & \text{if } \{i,j\} \in \mathcal{E}, \\ 0, & \text{otherwise,} \end{cases}$$

and each $\phi_{\{i,j\}}$ is endowed with Properties P1–P3. Distinguished by the choices of the $\phi_{\{i,j\}}$'s, this class of algorithms can also be expressed as

$$\dot{x}_i(t) = \sum_{j\in\mathcal{N}_i} \phi_{\{i,j\}}(y_i(t)x_j(t) - y_j(t)x_i(t))x_j(t), \qquad \forall i \in \mathcal{V},\ \forall t \geq 0. \tag{4.12}$$

As for the initial state $x(0)$ of the algorithms, the only restriction is that $x(0) > 0$, which the nodes can easily implement, and the reason for that will be clear shortly.

4.4 Analysis of Continuous-Time Algorithms

4.4.1 Convergence Analysis

In this subsection, we show that each algorithm in the class has certain appealing features and achieves the goal.

Theorem 4.1. *Consider the network modeled in Chapter 4.2 and let Assumption 4.1 hold. Suppose the continuous-time algorithm* (4.11) *with Properties P1–P3 is used and the initial state $x(0) > 0$. Then, $x(t)$ asymptotically converges to x^* as $t \to \infty$, where $x^* \in \mathcal{X}^*$ and $\|x^*\| = \|x(0)\|$.*

To prove Theorem 4.1, we first introduce the following notations and lemmas. To begin, let initial state $x(0) > 0$ be given and let $c = \|x(0)\| > 0$. In addition, let $\mathcal{S}^{N-1} = \{z \in \mathbb{R}^N : z^T z = c^2\}$ be the sphere of radius c centered at the origin of $\mathbb{R}^N$, $\mathbb{R}^N_+ = \{z \in \mathbb{R}^N : z \geq 0\}$ be the nonnegative orthant in $\mathbb{R}^N$, and $\Omega = \mathcal{S}^{N-1} \cap \mathbb{R}^N_+$ be their intersection.

With this setup, the following lemmas can be established concerning algorithm (4.11) or (4.12):

Lemma 4.1. *Consider the setup of Theorem 4.1. The following statements hold: (i) the sphere $\mathcal{S}^{N-1}$ is a positively invariant set; (ii) the nonnegative orthant $\mathbb{R}_+^N$ is a positively invariant set; (iii) the set $\Omega = \mathcal{S}^{N-1} \cap \mathbb{R}_+^N$ is a compact, positively invariant set.*

Proof. To prove (i), let $\tau \geq 0$ and $x(\tau) \in \mathcal{S}^{N-1}$. Due to (4.11) and the fact that $S(z)$ is skew-symmetric for all $z \in \mathbb{R}^N$, we have

$$\frac{d}{dt}x(t)^T x(t) = 2x(t)^T \dot{x}(t) = 2x(t)^T S(x(t))x(t) = 0, \quad \forall t \geq \tau.$$

Thus, $x(t) \in \mathcal{S}^{N-1}$ $\forall t \geq \tau$, implying that $\mathcal{S}^{N-1}$ is positively invariant.

To prove (ii), suppose $x(t) \in \mathbb{R}_+^N$ with $x_i(t) = 0$ for some $i \in \mathcal{V}$ and $t \geq 0$. Then, from (4.12),

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i} \phi_{\{i,j\}}(y_i(t)x_j(t))x_j(t). \tag{4.13}$$

Since $x(t) \in \mathbb{R}_+^N$, $x_j(t) \geq 0$ $\forall j \in \mathcal{N}_i$. Because $x_i(t) = 0$ and because of Assumption 4.1 and (4.1), $y_i(t) = \sum_{j \in \mathcal{N}_i} w_{ij}x_j(t) \geq 0$. Thus, $y_i(t)x_j(t) \geq 0$ $\forall j \in \mathcal{N}_i$. Due to Properties P2 and P3, $\phi_{\{i,j\}}(y_i(t)x_j(t)) \geq 0$ $\forall j \in \mathcal{N}_i$. Hence, from (4.13), we have $\dot{x}_i(t) \geq 0$. It follows that whenever $x(t) \in \mathbb{R}_+^N$ and $x_i(t) = 0$, we have $\dot{x}_i(t) \geq 0$. Therefore, $\mathbb{R}_+^N$ is positively invariant.

To prove (iii), note that $\mathcal{S}^{N-1}$ is compact and $\mathbb{R}_+^N$ is closed. Thus, being their intersection, Ω is compact. Since (i) and (ii) hold, both $\mathcal{S}^{N-1}$ and $\mathbb{R}_+^N$ are positively invariant. Hence, being their intersection, so is Ω. $\square$

Lemma 4.2. *Consider the setup of Theorem 4.1. The intersection of Ω and $\mathcal{X}^*$ has exactly one element, denoted as x^*, i.e., $\Omega \cap \mathcal{X}^* = \{x^*\}$. In addition, $x^* \gg 0$ and $\|x^*\| = \|x(0)\|$.*

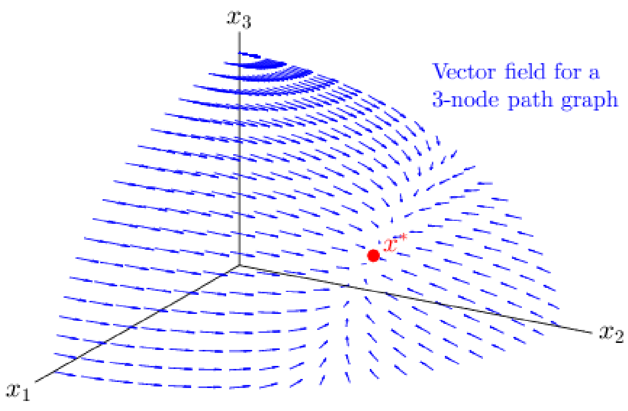


Figure 4.2: Vector field for a 3-node path graph.

Proof. By statement (ii) of Proposition 4.1, $\mathcal{X}^*$ is one-dimensional and is of the form $\mathcal{X}^* = \{\alpha z : \alpha \in \mathbb{R}, \alpha \neq 0\}$, where $z \in \mathbb{R}^N$ and $z \gg 0$. Thus, $\mathbb{R}_+^N \cap \mathcal{X}^*$ is a ray of the form $\{\alpha z : \alpha \in \mathbb{R}, \alpha > 0\}$. It follows that $\Omega \cap \mathcal{X}^* = \mathcal{S}^{N-1} \cap \mathbb{R}_+^N \cap \mathcal{X}^*$ has exactly one element denoted as x^*. Since $x^* \in \mathbb{R}_+^N \cap \mathcal{X}^*$, $x^* \gg 0$. Since $x^* \in \mathcal{S}^{N-1}$, $\|x^*\| = c = \|x(0)\|$. $\square$

Lemmas 4.1 and 4.2 explain why the restriction $x(0) > 0$ is imposed on the initial state $x(0)$ of the algorithms. With this restriction, we have $x(0) \in \mathcal{S}^{N-1}$ by definition and $x(0) \in \mathbb{R}_+^N$, so that $x(0) \in \Omega$. Since Ω is positively invariant according to Lemma 4.1, $x(t)$ is guaranteed to remain in Ω $\forall t \geq 0$. Since the goal is for $x(t)$ to converge to an eigenvector in $\mathcal{X}^*$, the fact that the intersection of Ω and $\mathcal{X}^*$ has exactly one element x^* according to Lemma 4.2, is appealing: it means that we may focus on establishing that the vector field on Ω must be such that $x(t)$ asymptotically converges to that x^*, as illustrated in Figure 4.2 for a 3-node path graph. Our tool for doing so is the LaSalle's invariance principle.

Consider a quadratic function $V : \mathbb{R}^N \to \mathbb{R}$, defined as

$$V(x) = \frac{1}{2}(x^{*T}Wx^* - x^TWx), \tag{4.14}$$

where x^* is from Lemma 4.2. Differentiating V in (4.14) with respect to time t and using (4.2), (4.11), and Properties P2 and P3, we obtain

$$\begin{aligned}\dot{V}(x(t)) &= -x(t)^TW\dot{x}(t)\\ &= -y(t)^TS(x(t))x(t)\\ &= -\sum_{\{i,j\}\in\mathcal{E}}(y_i(t)x_j(t) - y_j(t)x_i(t))\\ &\qquad\qquad \times \phi_{\{i,j\}}(y_i(t)x_j(t) - y_j(t)x_i(t))\\ &\leq 0. \end{aligned}\tag{4.15}$$

Next, let $E \subset \Omega \subset \mathbb{R}^N$ be defined as

$$E = \{z \in \Omega : \dot{V}(z) = 0\}. \tag{4.16}$$

Then, we have:

Lemma 4.3. *Consider the setup of Theorem 4.1. Then, set E is given by $E = \{x^*\}$.*

Proof. First, we show that $x^* \in E$. Notice from (4.2) that when $x = x^* = (x_1^*, x_2^*, \ldots, x_N^*)$,

$$y = Wx^* = \lambda^*x^* = (\lambda^*x_1^*, \lambda^*x_2^*, \ldots, \lambda^*x_N^*).$$

Thus,

$$y_ix_j - y_jx_i = \lambda^*x_i^*x_j^* - \lambda^*x_j^*x_i^* = 0, \quad \forall\{i,j\} \in \mathcal{E}. \tag{4.17}$$

Substituting (4.17) into (4.15) yields $\dot{V}(x^*) = 0$. Hence, $x^* \in E$.

Next, we show that x^* is the only element in E. Assume, to the contrary, that there exists another element $x \in E$ with $x \neq x^*$. Due to (4.15) and Properties P2 and P3, we have

$$(y_i x_j - y_j x_i)\phi_{\{i,j\}}(y_i x_j - y_j x_i) = 0, \quad \forall \{i,j\} \in \mathcal{E}. \tag{4.18}$$

Again due to Properties P2 and P3, $\phi_{\{i,j\}}(z) = 0$ if and only if $z = 0$. Thus, (4.18) implies that

$$y_i x_j - y_j x_i = 0, \quad \forall \{i,j\} \in \mathcal{E}. \tag{4.19}$$

Now since $x \in E \subset \Omega$, we may consider the following two cases: (i) $x \gg 0$, which means that $x_i > 0$ $\forall i \in \mathcal{V}$; and (ii) $x > 0$ with $x_i = 0$ for some $i \in \mathcal{V}$. For case (i), let $\alpha_i = y_i/x_i$ $\forall i \in \mathcal{V}$. Substituting $y_i = \alpha_i x_i$ into (4.19) and using the fact that $x \gg 0$, we deduce that

$$\alpha_i - \alpha_j = 0, \quad \forall \{i,j\} \in \mathcal{E}. \tag{4.20}$$

Since the graph $\mathcal{G}$ is connected, (4.20) implies that there exists α such that $\alpha_i = \alpha$ $\forall i \in \mathcal{V}$. Since $y = Wx = \alpha x$, α is an eigenvalue of W and $x \gg 0$ is the corresponding eigenvector. By statement (iii) of Proposition 4.1, we conclude that $\alpha = \lambda^*$ and $x \in \mathcal{X}^*$. By Lemma 4.2, $x = x^*$, which is a contradiction.

For case (ii), let $\mathcal{I} = \{i \in \mathcal{V} : x_i = 0\}$. Then, $\mathcal{I} \neq \emptyset$ and $\mathcal{I} \subsetneq \mathcal{V}$. Let M denote the number of elements in $\mathcal{I}$. Then, $0 < M < N$. Pick any $i \in \mathcal{I}$ so that $x_i = 0$. Then, from (4.1) and statements (i) and (ii) of Assumption 4.1,

$$y_i = w_{ii} x_i + \sum_{j \in \mathcal{N}_i} w_{ij} x_j = \sum_{j \in \mathcal{N}_i} w_{ij} x_j \geq 0. \tag{4.21}$$

From (4.21), either $y_i > 0$ or $y_i = 0$. Suppose $y_i > 0$. By substituting $x_i = 0$ into (4.19), we have $y_i x_j = 0$ $\forall j \in \mathcal{N}_i$. Since $y_i > 0$, $x_j = 0$ $\forall j \in \mathcal{N}_i$.

Substituting this into (4.21) yields $y_i = 0$, contradicting the assumption that $y_i > 0$. Therefore, $y_i = 0$. Since $i \in \mathcal{I}$ is arbitrary, we have $y_i = 0\ \forall i \in \mathcal{I}$. Next, by permuting the indices of x and y, (4.2) can be written as

$$\begin{bmatrix} y_{\mathcal{I}} \\ y_{\mathcal{V}-\mathcal{I}} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \begin{bmatrix} x_{\mathcal{I}} \\ x_{\mathcal{V}-\mathcal{I}} \end{bmatrix},$$

where $x_{\mathcal{I}} \in \mathbb{R}^M$, $x_{\mathcal{I}} = 0$, $x_{\mathcal{V}-\mathcal{I}} \in \mathbb{R}^{N-M}$, $x_{\mathcal{V}-\mathcal{I}} \gg 0$, $y_{\mathcal{I}} \in \mathbb{R}^M$, $y_{\mathcal{I}} = 0$, $y_{\mathcal{V}-\mathcal{I}} \in \mathbb{R}^{N-M}$, $W_{11} \in \mathbb{R}^{M\times M}$, $W_{12} \in \mathbb{R}^{M\times(N-M)}$, $W_{21} \in \mathbb{R}^{(N-M)\times M}$, and $W_{22} \in \mathbb{R}^{(N-M)\times(N-M)}$. Because $x_{\mathcal{I}} = 0$, $x_{\mathcal{V}-\mathcal{I}} \gg 0$, and $y_{\mathcal{I}} = 0$ and because of statements (i) and (ii) in Assumption 4.1, we have $W_{12} = 0$. This is a contradiction since W is irreducible by statement (iii) in Assumption 4.1.

Combining the contradiction in case (i) and the one in case (ii), we conclude that x^* is the only element in E. $\square$

With the above lemmas in hand, we now prove Theorem 4.1:

Proof of Theorem 4.1. Let $x(0) > 0$ be given and let c, $\mathcal{S}^{N-1}$, and Ω be as defined earlier. In addition, let x^* be as defined in Lemma 4.2, V and $E \subset \Omega$ be as defined in (4.14) and (4.16), and $M \subset E$ be the largest invariant set in E. Notice from Lemma 4.1 that Ω is compact and positively invariant with respect to algorithm (4.11), and from (4.15) that $\dot{V}(x) \leq 0$ for all $x \in \Omega$. Also note from Lemma 4.3 that $E = \{x^*\}$. Since $M \subset E = \{x^*\}$, either $M = \emptyset$ or $M = \{x^*\}$. Since x^* is an equilibrium point of (4.11), $M = \{x^*\}$. By LaSalle's invariance principle [133], $x(t)$ asymptotically converges to x^* as $t \to \infty$, as desired. $\square$

Note that since $\lim_{t\to\infty} x(t) = x^*$ and $x^* \gg 0$ by Lemma 4.2, there exists $t' \geq 0$ such that for all $t \geq t'$, $x(t) \gg 0$. Hence, for all $t \geq t'$, each node $i \in \mathcal{V}$ could maintain an estimate $\lambda_i(t) \in \mathbb{R}$ of the unknown eigenvalue λ^* and define it as $\lambda_i(t) = y_i(t)/x_i(t)$ without facing any division-by-zero issue. Moreover, since $\lim_{t\to\infty} x(t) = x^*$, we have $\lim_{t\to\infty} y(t) = \lambda^* x^*$ from (4.2), so that $\lim_{t\to\infty} \lambda_i(t) = \lambda^*\ \forall i \in \mathcal{V}$. Therefore, the determination of λ^* is a by-product of the determination of x^* using algorithm (4.11) or (4.12).

4.4.2 Convergence Rate

In this subsection, we derive the convergence rate of a class of continuous-time algorithms. To enable the derivation, let Assumption 4.2 hold. In addition, let $\phi_{\{i,j\}}(z)\ \forall \{i,j\} \in \mathcal{E}$ satisfy Properties P1 and P3 in Chapter 4.3 and the following property:

P2'. There exists $\kappa > 0$ such that $z\phi_{\{i,j\}}(z) \geq \kappa z^2\ \forall \{i,j\} \in \mathcal{E}, \forall z \in \mathbb{R}$.

Note that Assumption 4.2 and Property P2' here are slightly more restrictive than Assumption 4.1 and Property P2, respectively. Therefore, all the results from Chapters 4.3 and 4.4, which are obtained under the latter, are valid here. This means that we do not have to show again, for example, that $x(t)$ will remain on Ω and will asymptotically converge to x^*.

Let $D \in \{1, 2, \ldots, N-1\}$ be the diameter of graph $\mathcal{G}$. In addition, let $\lambda_2 \in \mathbb{R}$ and $\lambda_N \in \mathbb{R}$ be the second largest eigenvalue and the smallest eigenvalue of W, respectively, which by Proposition 4.1 satisfy $\lambda_N \leq \lambda_2 < \lambda^*$. Moreover, let

$$\bar{w} = \max_{i\in\mathcal{V}} \sum_{j\in\mathcal{N}_i} w_{ij} > 0, \tag{4.22}$$

$$\bar{w}' = \max_{i\in\mathcal{V}} |w_{ii}| \geq 0, \tag{4.23}$$

$$\underline{w} = \min_{\{i,j\}\in\mathcal{E}} w_{ij} > 0, \tag{4.24}$$

$$\beta = \frac{2D+1}{D} \max\{\beta', \beta''\} > 2, \tag{4.25}$$

$$\beta' = \max_{\{i,j\}\in\mathcal{E}} \frac{|w_{ii}| + |w_{jj}|}{w_{ij}} \geq 0, \tag{4.26}$$

$$\beta'' = \max_{i\in\mathcal{V}, j\in\mathcal{N}_i} \frac{\sum_{l\in\mathcal{N}_i} w_{il}}{w_{ij}} \geq 1. \tag{4.27}$$

Since $\mathcal{G}$ is connected, $w_{ij} > 0\ \forall\{i,j\} \in \mathcal{E}$ by Assumption 4.2, and $D \geq 1$, we see that the constants β, β', and β'' are well-defined and the rightmost inequalities in (4.22)–(4.27) are satisfied. Lastly, let $x(0) > 0$ be given and $c = \|x(0)\| > 0$, so that by Lemmas 4.1 and 4.2,

$$\|x^*\| = \|x(t)\| = c, \quad \forall t \geq 0, \tag{4.28}$$

$$0 \leq x_i(t) \leq c, \quad \forall i \in \mathcal{V},\ \forall t \geq 0. \tag{4.29}$$

The following theorem characterizes the convergence rate of the aforementioned class of algorithms:

Theorem 4.2. *Consider the network modeled in Chapter 4.2 and let Assumption 4.2 hold. Suppose algorithm* (4.12) *is used. Let Properties P1, P2', and P3 hold and let the initial state satisfy* $x(0) > 0$*. Then, for each* $t \geq 0$*,*

$$V(x(t)) \leq \frac{V(x(0))}{V(x(0))\gamma t + 1}, \tag{4.30}$$

where V *is as defined in* (4.14) *and*

$$\gamma = \frac{4\kappa \min\left\{\frac{1}{D^2}, \frac{\underline{w}^2}{(2D+1)^2(\lambda^* - \lambda_N)^2}\right\}}{\beta^{2D^2+2D} N^2}. \tag{4.31}$$

In addition,

$$\|x(t) - x^*\| \leq \sqrt{\frac{4V(x(0))}{(\lambda^* - \lambda_2)(V(x(0))\gamma t + 1)}}. \tag{4.32}$$

To appreciate Theorem 4.2, observe that γ characterizes the convergence rate. If γ is near zero, algorithm (4.11) may converge very slowly. Because γ depends on κ, $\underline{w}$, D, $\lambda^* - \lambda_N$, and β, slow convergence may occur if one or more of the following conditions hold: (i) the algorithm parameter κ is small, (ii) graph $\mathcal{G}$ has a large diameter D, (iii) the entries of W have large diagonal elements but small off-diagonal ones so that β is large, and (iv) the eigenvalues of W are very close to one another so that $\lambda^* - \lambda_N$ is small. On the flip side, if conditions (i)–(iv) do not hold, then the value of γ in (4.32) would be large, suggesting that algorithm (4.11) would converge quickly. Therefore, Theorem 4.2 provides a glimpse into the performance of algorithm (4.11).

To prove Theorem 4.2, consider the following notations and lemmas. Let

$$\underline{x}(t) = \min_{i \in \mathcal{V}} x_i(t) \geq 0, \quad \forall t \geq 0, \tag{4.33}$$

$$\bar{x}_i(t) = \max_{j \in \mathcal{N}_i} x_j(t) \leq c, \quad \forall i \in \mathcal{V},\ \forall t \geq 0. \tag{4.34}$$

Due to (4.29), the rightmost inequalities in (4.33) and (4.34) are satisfied.

Lemma 4.4. *Suppose Assumption 4.2 holds. For each $t \geq 0$, if $\|x(t)\| = \|x^*\|$ and $x(t) > 0$, then*

$$\|x(t) - x^*\| \leq \sqrt{\frac{4V(x(t))}{\lambda^* - \lambda_2}}. \tag{4.35}$$

Proof. Let $t \geq 0$ be given and suppose $\|x(t)\| = \|x^*\|$ and $x(t) > 0$. Let $x(t)$ be uniquely decomposed as $x(t) = x_\parallel(t) + x_\perp(t)$, where $x_\parallel(t) \in \mathbb{R}^N$ is parallel to x^* and $x_\perp(t) \in \mathbb{R}^N$ is normal to x^*. Then, we have

$$\|x(t)\|^2 = \|x_\perp(t)\|^2 + \|x_\parallel(t)\|^2, \tag{4.36}$$

$$x_\perp(t)^T x_\parallel(t) = 0. \tag{4.37}$$

Using (4.36) and (4.37), we obtain

$$\begin{aligned}\|x(t)-x^*\|^2 &= \|x_\perp(t)-(x^*-x_\parallel(t))\|^2\\ &= \|x_\perp(t)\|^2+\|x^*-x_\parallel(t)\|^2\\ &= \|x_\perp(t)\|^2+\|x^*\|^2+\|x_\parallel(t)\|^2-2x^{*T}x_\parallel(t)\\ &= 2\|x_\perp(t)\|^2+2\|x_\parallel(t)\|^2-2x^{*T}x_\parallel(t). \qquad (4.38)\end{aligned}$$

Since $x^* > 0$ by Proposition 4.1 and $x(t) > 0$, we have $x^{*T}x(t) > 0$ and so $x^{*T}x_\parallel(t) > 0$. Thus,

$$2\|x_\parallel(t)\|^2-2x^{*T}x_\parallel(t) = 2(\|x_\parallel(t)\|(\|x_\parallel(t)\|-\|x^*\|)) \le 0.$$

This, along with (4.38), implies that

$$\|x(t)-x^*\| \le \sqrt{2}\|x_\perp(t)\|. \qquad (4.39)$$

Next, by definition of $x_\perp(t)$ and $x_\parallel(t)$, and from (4.36) and (4.37), we can write (4.14) as

$$\begin{aligned}V(x(t)) &= \frac{1}{2}\Big(x^{*T}Wx^*-(x_\perp(t)+x_\parallel(t))^TW(x_\perp(t)+x_\parallel(t))\Big)\\ &= \frac{1}{2}(\lambda^*\|x^*\|^2-\lambda^*\|x_\parallel(t)\|^2-x_\perp(t)^TWx_\perp(t))\\ &= \frac{1}{2}(\lambda^*\|x_\perp(t)\|^2-x_\perp(t)^TWx_\perp(t))\\ &\ge \frac{1}{2}(\lambda^*\|x_\perp(t)\|^2-\lambda_2\|x_\perp(t)\|^2).\end{aligned}$$

Therefore,

$$\|x_\perp(t)\| \le \sqrt{\frac{2V(x(t))}{\lambda^*-\lambda_2}}.$$

This, along with (4.39), implies (4.35). □

Remark 4.2. Although t is used in Lemma 4.4, it turns out that an identical result can be proved when t is replaced by discrete-time k with the condition that $\|x(k)\| = \|x^*\|$ and $x(k) > 0$ hold. This property will be exploited later in Chapter 5.4 when we introduce and analyze a gossip algorithm.

Lemma 4.4 shows that there is a one-to-one relationship between $\|x(t) - x^*\|$ and $V(x(t))$. Thus, in the following lemmas, we will focus on $V(x(t))$ instead of $x(t)$.

Lemma 4.5. *Consider the setup of Theorem 4.2. For each* $t \geq 0$*, if* $\underline{x}(t) > 0$*, then*

$$\dot{V}(x(t)) \leq -\frac{4\kappa \underline{x}(t)^4}{c^4 D^2} V(x(t))^2, \tag{4.40}$$

where $\dot{V}$ *is defined in* (4.15).

Proof. Let $t \geq 0$ be given and suppose $\underline{x}(t) > 0$. Then, $x_i(t) > 0$ $\forall i \in \mathcal{V}$ by (4.33). Define $z_i(t) = y_i(t)/x_i(t)$ and $\tilde{z}_i(t) = z_i(t) - \lambda^*$ $\forall i \in \mathcal{V}$. Moreover, let $p \in \arg\max_{i\in\mathcal{V}} \tilde{z}_i(t)$ and $q \in \arg\min_{i\in\mathcal{V}, i\neq p} \tilde{z}_i(t)$. By the Perron-Frobenius theorem, $\tilde{z}_p(t) \geq 0$ and $\tilde{z}_q(t) \leq 0$. Since graph $\mathcal{G}$ is connected, there exists a shortest path between nodes p and q, whose length is less than or equal to D. Therefore, there exists $\{r, s\} \in \mathcal{E}$ lying on that shortest path such that

$$\tilde{z}_r(t) - \tilde{z}_s(t) \geq \frac{1}{D}(\tilde{z}_p(t) - \tilde{z}_q(t)).$$

Since $\tilde{z}_p(t) \geq 0$ and $\tilde{z}_q(t) \leq 0$,

$$\tilde{z}_r(t) - \tilde{z}_s(t) \geq -\frac{1}{D}\tilde{z}_q(t). \tag{4.41}$$

Next, (4.15), Property P2', (4.33), and (4.41) imply that

$$\dot{V}(x(t)) = -\sum_{\{i,j\}\in\mathcal{E}} (y_i(t)x_j(t) - y_j(t)x_i(t))$$

$$
\begin{aligned}
&\times \phi_{\{i,j\}}(y_i(t)x_j(t) - y_j(t)x_i(t)) \\
&\le -\kappa \sum_{\{i,j\}\in\mathcal{E}} (y_i(t)x_j(t) - y_j(t)x_i(t))^2 \\
&= -\kappa \sum_{\{i,j\}\in\mathcal{E}} (\tilde{z}_i(t) - \tilde{z}_j(t))^2 x_i(t)^2 x_j(t)^2 \\
&\le -\kappa \underline{x}(t)^4 \sum_{\{i,j\}\in\mathcal{E}} (\tilde{z}_i(t) - \tilde{z}_j(t))^2 \\
&\le -\kappa \underline{x}(t)^4 (\tilde{z}_r(t) - \tilde{z}_s(t))^2 \\
&\le -\kappa \frac{\underline{x}(t)^4}{D^2} \tilde{z}_q(t)^2. \qquad (4.42)
\end{aligned}
$$

Due to (4.14), (4.1), and (4.28),

$$
\begin{aligned}
V(x(t)) &= \frac{1}{2}\Big(\lambda^* c^2 - \sum_{i\in\mathcal{V}} z_i(t) x_i(t)^2\Big) \\
&= \frac{1}{2}\Big(\lambda^* c^2 - \sum_{i\in\mathcal{V}} (\tilde{z}_i(t) + \lambda^*) x_i(t)^2\Big) \\
&= -\frac{1}{2}\sum_{i\in\mathcal{V}} \tilde{z}_i(t) x_i(t)^2 \le -\frac{1}{2}\tilde{z}_q(t) c^2. \qquad (4.43)
\end{aligned}
$$

Since $\tilde{z}_q \le 0$, (4.43) implies that

$$
V(x(t))^2 \le \frac{c^4}{4}\tilde{z}_q(t)^2.
$$

Combining this with (4.42) yields (4.40). □

The above lemma indicates that the larger $\underline{x}(t)$ and $V(x(t))$, the faster $V(x(t))$ drops to zero. Besides, by Theorem 4.1, $x(t)$ will converge to x^* as $t \to \infty$ so that $\underline{x}(t)$ will converge to the smallest entry of vector x^*. Therefore, if one of the entries of x^* is close to zero, the rate at which $V(x(t))$ drops would be small when $x(t)$ is close to x^*. Actually, when W is "close" to reducible, at least one entry of x^* would be close to zero. Indeed, note that $\underline{x}(t)$ is not given and it may depend on the initial state $x(0)$, matrix W and other algorithm parameters. Thus, in next few lemmas we study the upper bound of $V(\dot{x}(t))$ when $\underline{x}(t)$ is small.

Lemma 4.6. *Suppose Assumption 4.2 holds. For each $t \geq 0$, if $\|x(t)\| = c$, $x(t) > 0$, and*

$$0 < \underline{x}(t) < \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}}, \tag{4.44}$$

then there exists $\{u,v\} \in \mathcal{E}$ such that

$$\begin{aligned} & w_{uv} x_u(t)^2 - \bar{w} \bar{x}_u(t) x_v(t) - (|w_{uu}| + |w_{vv}|) x_u(t) x_v(t) \\ & \geq \frac{\underline{w} c^2}{(2D+1)\beta^{D^2+D} N} > 0. \end{aligned} \tag{4.45}$$

Proof. Let $t \geq 0$ be given and suppose (4.44) holds. Let $p \in \arg\max_{i \in \mathcal{V}} x_i(t)$ and $q \in \arg\min_{i \in \mathcal{V}, i \neq p} x_i(t)$. Since $\mathcal{G}$ is connected, there exists a shortest path from nodes p to q with length $L \in \{1, 2, \ldots, D\}$. Let the nodes lying on this shortest path be denoted as $r_0, r_1, \ldots, r_{L-1}, r_L$, where $r_0 = p$ and $r_L = q$. It follows that $\forall m \in \{0, 1, \ldots, L-1\}$, we have $\{r_m, r_{m+1}\} \in \mathcal{E}$ and $w_{r_m r_{m+1}} > 0$ thanks to Assumption 4.2.

We first show that there exists $l \in \{0, 1, \ldots, L-1\}$ such that

$$x_{r_m}(t) \leq \beta^{m+1} x_{r_{m+1}}(t), \quad \forall m \in \{0, 1, \ldots, l-1\}, \tag{4.46}$$

$$x_{r_l}(t) > \beta^{l+1} x_{r_{l+1}}(t). \tag{4.47}$$

Assume, to the contrary, that $x_{r_m} \leq \beta^{m+1} x_{r_{m+1}}$ $\forall m \in \{0, 1, \ldots, L-1\}$. Then,

$$\begin{aligned} x_{r_L}(t) & \geq \frac{1}{\beta^L} x_{r_{L-1}}(t) \geq \frac{1}{\beta^{L+(L-1)}} x_{r_{L-2}}(t) \\ & \geq \cdots \geq \frac{1}{\beta^{\frac{L^2+L}{2}}} x_{r_0}(t). \end{aligned} \tag{4.48}$$

Since $x_{r_0}(t) = x_p(t) \geq x_i(t)\ \forall i \in \mathcal{V}$ and due to (4.28), $x_{r_0}(t) \geq \frac{c}{\sqrt{N}}$. This, together with $D \geq L \geq 1$ and $\beta \geq 2$, as well as (4.33) and (4.48), implies that

$$\underline{x}(t) = x_{r_L}(t) \geq \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}},$$

which contradicts (4.44).

Next, let $l \in \{0, 1, \ldots, L-1\}$ be such that (4.46) and (4.47) hold. Then, due to (4.46), (4.25), and $D \geq 1$,

$$\begin{aligned} x_{r_l}(t) &\geq \frac{1}{\beta^l} x_{r_{l-1}}(t) \geq \frac{1}{\beta^{l+(l-1)}} x_{r_{l-2}}(t) \\ &\geq \cdots \geq \frac{1}{\beta^{\frac{l^2+l}{2}}} x_{r_0}(t) \geq \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}} > 0. \end{aligned} \tag{4.49}$$

Let $s_0 = r_{l+1}$ and $s_1 = r_l$. In addition, let $s_2 \in \arg\max_{i \in \mathcal{N}_{s_1}} x_i(t)$, $s_3 \in \arg\max_{i \in \mathcal{N}_{s_2}} x_i(t)$, ... , $s_{l+2} \in \arg\max_{i \in \mathcal{N}_{s_{l+1}}} x_i(t)$. We now show that there exists $h \in \{1, 2, \ldots, l+1\}$ such that

$$x_{s_{m+1}}(t) > \beta^{l-m+1} x_{s_m}(t), \quad \forall m \in \{1, 2, \ldots, h-1\}, \tag{4.50}$$

$$x_{s_{h+1}}(t) \leq \beta^{l-h+1} x_{s_h}(t). \tag{4.51}$$

Assume, to the contrary, that $x_{s_{m+1}}(t) > \beta^{l-m+1} x_{s_m}(t)\ \forall m \in \{1, 2, \ldots, l+1\}$. Then,

$$\begin{aligned} x_{s_{l+1}}(t) &> \beta x_{s_l}(t) > \beta^{1+2} x_{s_{l-1}}(t) \\ &> \cdots > \beta^{\frac{l^2+l}{2}} x_{s_1}(t) = \beta^{\frac{l^2+l}{2}} x_{r_l}(t). \end{aligned} \tag{4.52}$$

Applying (4.49) to (4.52), we have $x_{s_{l+1}}(t) > x_{r_0}(t) = x_p(t)$, which contradicts the definition of p.

Next, let $h \in \{1, 2, \ldots, l+1\}$ be such that (4.50) and (4.51) hold. Then, applying (4.50) and the inequality $\beta > 2$ alternately, and using $s_1 = r_l$ and (4.49), we obtain

$$x_{s_h}(t) > \beta^{l-h+2} x_{s_{h-1}}(t) > x_{s_{h-1}}(t) > \beta^{l-h+3} x_{s_{h-2}}(t)$$

$$
\begin{aligned}
&> x_{s_{h-2}}(t) > \cdots > \beta^l x_{s_1}(t) > x_{s_1}(t) = x_{r_l}(t) \\
&\geq \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}} > 0. \qquad (4.53)
\end{aligned}
$$

Note from the definitions of s_0 and s_1 and from (4.47) that $x_{s_1} > \beta^{l+1} x_{s_0}$. This, along with (4.50), implies that

$$
x_{s_h}(t) > \beta^{l-h+2} x_{s_{h-1}}(t). \qquad (4.54)
$$

Now, let $u = s_h$ and $v = s_{h-1}$. Then, by definition of s_{h+1} and (4.34), $\bar{x}_u = x_{s_{h+1}}$. Due to (4.53), Assumption 4.2, (4.54), (4.51), (4.25), and (4.24), and the fact that $0 < \frac{1}{\beta^{l-h+1}} \leq 1$, we have

$$
\begin{aligned}
&w_{uv} x_u(t)^2 - \bar{w} \bar{x}_u(t) x_v(t) - (|w_{uu}| + |w_{vv}|) x_u(t) x_v(t) \\
&= w_{s_h s_{h-1}} x_{s_h}(t)^2 \Big(1 - \frac{\bar{w}}{w_{s_h s_{h-1}}} \frac{x_{s_{h-1}}(t)}{x_{s_h}(t)} \frac{x_{s_{h+1}}(t)}{x_{s_h}(t)} \\
&\quad - \frac{|w_{s_{h-1} s_{h-1}}| + |w_{s_h s_h}|}{w_{s_h s_{h-1}}} \frac{x_{s_{h-1}}(t)}{x_{s_h}(t)} \Big) \\
&\geq w_{s_h s_{h-1}} x_{s_h}(t)^2 \Big(1 - \beta \frac{D}{2D+1} \frac{1}{\beta^{l-h+2}} \beta^{l-h+1} \\
&\quad - \beta \frac{D}{2D+1} \frac{1}{\beta^{l-h+2}} \Big) \\
&\geq \frac{w}{2D+1} x_{s_h}(t)^2. \qquad (4.55)
\end{aligned}
$$

Applying (4.53) to (4.55), we obtain the first inequality in (4.45). Due to (4.24) and (4.25) and since $D \geq 1$, $N \geq 2$, and $c > 0$, we get the second inequality. $\square$

Remark 4.3. As in Remark 4.2, although t is used in Lemma 4.6, it turns out that an identical result can be proved when t is replaced by discrete-time k with the condition that $\|x(k)\| = c$ and $x(k) > 0$. We will make use of this property later in Chapter 5.4 when we introduce and analyze a gossip algorithm.

Lemma 4.7. *Consider the setup of Theorem 4.2. For each $t \geq 0$, if* (4.44) *holds, then*

$$\dot{V}(x(t)) \leq -\frac{\kappa \underline{w}^2 c^4}{(2D+1)^2 \beta^{2D^2+2D} N^2}. \tag{4.56}$$

Proof. Let $t \geq 0$ be given and suppose (4.44) holds. By Lemma 4.6, there exists $\{u, v\} \in \mathcal{E}$ such that (4.45) holds. Because of (4.1), Assumption 4.2, (4.29), (4.22), (4.34), and (4.45),

$$\begin{aligned}
&y_v(t)x_u(t) - y_u(t)x_v(t) \\
&\quad = \Big(w_{vv}x_v(t) + \sum_{j \in \mathcal{N}_v} w_{vj}x_j(t)\Big)x_u(t) \\
&\qquad - \Big(w_{uu}x_u(t) + \sum_{j \in \mathcal{N}_u} w_{uj}x_j(t)\Big)x_v(t) \\
&\quad \geq (-|w_{vv}|x_v(t) + w_{uv}x_u(t))x_u(t) \\
&\qquad - (|w_{uu}|x_u(t) + \bar{w}\bar{x}_u(t))x_v(t) \\
&\quad \geq w_{uv}x_u(t)^2 - \bar{w}\bar{x}_u(t)x_v(t) - (|w_{uu}| + |w_{vv}|)x_u(t)x_v(t) \\
&\quad \geq \frac{\underline{w}c^2}{(2D+1)\beta^{D^2+D}N} > 0.
\end{aligned} \tag{4.57}$$

Due to (4.15), Property P2', and (4.57),

$$\begin{aligned}
\dot{V}(x(t)) &= -\sum_{\{i,j\} \in \mathcal{E}} (y_i(t)x_j(t) - y_j(t)x_i(t)) \\
&\qquad \times \phi_{\{i,j\}}(y_i(t)x_j(t) - y_j(t)x_i(t)) \\
&\leq -\kappa \sum_{\{i,j\} \in \mathcal{E}} (y_i(t)x_j(t) - y_j(t)x_i(t))^2 \\
&\leq -\kappa(y_v(t)x_u(t) - y_u(t)x_v(t))^2 \\
&\leq -\frac{\kappa \underline{w}^2 c^4}{(2D+1)^2 \beta^{2D^2+2D} N^2},
\end{aligned}$$

which is exactly (4.56). □

With the above lemmas in hand, we now prove Theorem 4.2:

Proof of Theorem 4.2. Let $t \geq 0$ be given. If

$$\underline{x}(t) \geq \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}} > 0,$$

then from Lemma 4.5,

$$\dot{V}(x(t)) \leq -\underbrace{\frac{4\kappa}{\beta^{2D^2+2D} D^2 N^2}}_{\gamma_1} V(x(t))^2. \tag{4.58}$$

If, instead,

$$\underline{x}(t) < \frac{1}{\beta^{\frac{D^2+D}{2}}} \frac{c}{\sqrt{N}},$$

then from Lemma 4.7,

$$\dot{V}(x(t)) \leq -\frac{\kappa \underline{w}^2 c^4}{(2D+1)^2 \beta^{2D^2+2D} N^2}. \tag{4.59}$$

Due to (4.28) and the Rayleigh quotient,

$$V(x(t)) \leq \frac{1}{2}(\lambda^* c^2 - \lambda_N c^2).$$

It follows that

$$\frac{4V(x(t))^2}{(\lambda^* - \lambda_N)^2 c^4} \leq 1.$$

This, along with (4.59), implies that

$$\dot{V}(x(t)) \leq -\underbrace{\frac{4\kappa \underline{w}^2}{(2D+1)^2 \beta^{2D^2+2D} N^2 (\lambda^* - \lambda_N)^2}}_{\gamma_2} V(x(t))^2. \tag{4.60}$$

Let γ be as defined in (4.31). Note from (4.24) and (4.25) tha $\gamma > 0$. Also note that $\gamma = \min\{\gamma_1, \gamma_2\}$, where γ_1 and γ_2 are defined in (4.58) and (4.60). Thus, from (4.58) and (4.60), we have $\dot{V}(x(t)) \leq -\gamma V(x(t))^2$. Next, consider the scalar differential equation $\dot{v}(t) = -\gamma v(t)^2$ with initial state $v(0) = V(x(0)) \geq 0$. The solution to this differential equation is

$$v(t) = \frac{v(0)}{v(0)\gamma t + 1}, \quad \forall t \geq 0.$$

By the comparison lemma [133], $V(x(t)) \leq v(t)$ $\forall t \geq 0$. Therefore, (4.30) holds. Finally, using (4.30) and Lemma 4.4, we obtain (4.32). □

4.4.3 Special Cases

In this subsection, we consider two special cases for which less conservative lower bounds on the convergence rate can be obtained.

Proposition 4.2. *Consider the setup of Theorem 4.2 and suppose: $\mathcal{G}$ be a complete graph; W is the adjacency matrix of $\mathcal{G}$; $\phi_{\{i,j\}}(z) = z$ $\forall \{i,j\} \in \mathcal{E}$; and the initial state $x(t)$ satisfies $x(t) > 0$ and $\|x(0)\| = 1$. Then, for each $t \geq 0$,*

$$e^{-4t}V(x(0)) \leq V(x(t)) \leq e^{-2t}V(x(0)). \tag{4.61}$$

Proof. Let $t \geq 0$ be given. Since $\mathcal{G}$ is complete and W is its adjacency matrix, $\lambda^* = N - 1$. It follows from (4.14) that

$$\begin{aligned}
V(x(t)) &= \frac{1}{2}\Big(\lambda^* \sum_{i\in\mathcal{V}} x_i(t)^2 - \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}-\{i\}} x_i(t)x_j(t)\Big) \\
&= \frac{1}{2}\Big((N-1)\sum_{i\in\mathcal{V}} x_i(t)^2 - \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}-\{i\}} x_i(t)x_j(t)\Big) \\
&= \frac{1}{2}\Big(\frac{1}{2}\sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}-\{i\}} (x_i(t)^2 + x_j(t)^2) \\
&\quad - \sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}-\{i\}} x_i(x)x_j(t)\Big) \\
&= \frac{1}{4}\sum_{i\in\mathcal{V}} \sum_{j\in\mathcal{V}-\{i\}} (x_i(t) - x_j(t))^2
\end{aligned}$$

$$= \frac{1}{2} \sum_{\{i,j\}\in\mathcal{E}} (x_i(t) - x_j(t))^2. \tag{4.62}$$

Because of (4.1) and (4.62),

$$\begin{aligned}
\dot{V}(x(t)) &= -\sum_{\{i,j\}\in\mathcal{E}} (y_i(t)x_j(t) - y_j(t)x_i(t))^2 \\
&= -\sum_{\{i,j\}\in\mathcal{E}} \Big(x_j(t)^2 + x_j(t) \sum_{k\in\mathcal{V}-\{i,j\}} x_k(t) \\
&\qquad - x_i(t)^2 - x_i(t) \sum_{k\in\mathcal{V}-\{i,j\}} x_k(t)\Big)^2 \\
&= -\sum_{\{i,j\}\in\mathcal{E}} (x_j(t) - x_i(t))^2 \Big(\sum_{k\in\mathcal{V}} x_k(t)\Big)^2 \\
&= -2\Big(\sum_{k\in\mathcal{V}} x_k(t)\Big)^2 V(x(t)). \qquad (4.63)
\end{aligned}$$

Since $x(t) > 0$ and $\|x(t)\| = \|x(0)\| = 1$, we have $1 \leq (\sum_{k\in\mathcal{V}} x_k(t))^2 \leq 2$. This, along with (4.63), implies that $-4V(x(t)) \leq \dot{V}(x(t)) \leq -2V(x(t))$, so that (4.61) holds. □

Note that unlike (4.30) in Theorem 4.2, $V(x(t))$ decreases with exponential rate when $\mathcal{G}$ is a complete graph.

Corollary 4.1. *Consider the setup of Theorem 4.2 and suppose: $\mathcal{G}$ is a K-regular graph with $K \geq 2$; W is the adjacency matrix of $\mathcal{G}$; $\phi_{\{i,j\}}(z) = z$ $\forall\{i,j\} \in \mathcal{E}$; and the initial state $x(t)$ satisfies $x(t) > 0$ and $\|x(0)\| = 1$. Then, for each $t \geq 0$, $V(x(t))$ satisfies* (4.30), *where*

$$\gamma = \frac{1}{(k-1)^2(2D+1)^2(\frac{2D+1}{D}(k+1))^{2D^2+2D}N^2}. \tag{4.64}$$

Proof. Since the setup here is a special case of that of Theorem 4.2, $V(x(t))$ satisfies (4.30). In addition, $\kappa = 1$ and $c = 1$. Moreover, since $\mathcal{G}$ is a K-regular graph and W is its adjacency matrix, we have $\bar{w} = k - 1$, $\bar{w}' = 0$, $\underline{w} = 1$,

$\beta'' = k - 1$, $\beta' = 0$, $\beta = \frac{2D+1}{D}(k+1)$, and $\lambda^* = K - 1$. Since $|\lambda_N| \leq \lambda^*$, we have $\lambda_N \geq -(K-1)$. Substituting these into (4.31) yields (4.64). $\square$

4.5 Simulation Results

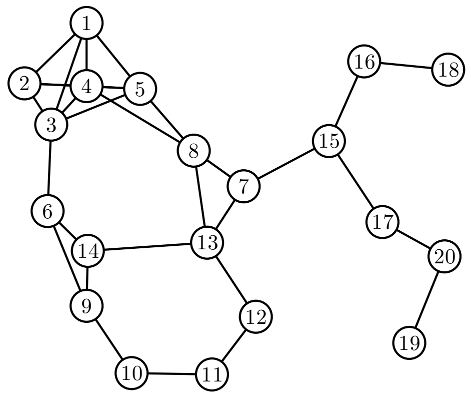
In this section, we present simulation results that demonstrate the effectiveness of the continuous-time algorithm (4.12).

Consider a network with $N = 20$ nodes, modeled as an undirected and connected graph $\mathcal{G}$, whose topology is shown in Figure 4.3(a). Suppose W is the adjacency matrix of $\mathcal{G}$, whose entries satisfy Assumption 4.1. Suppose algorithm (4.12) is used with each node $i \in \mathcal{V}$ letting its initial state satisfying $x_i(0) > 0$ and each pair of neighboring nodes i and j letting $\phi_{\{i,j\}}(z) = z$.

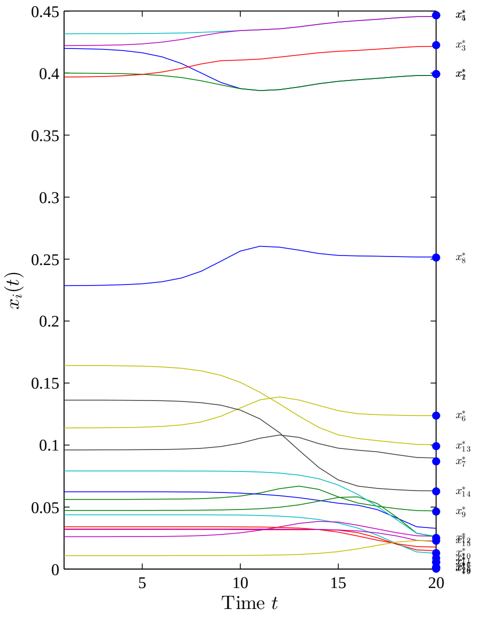
Figures 4.3(b) and 4.3(c) display the simulation result. Specifically, Figure 4.3(b) shows that node i's estimate $x_i(t)$ converges asymptotically to x_i^* $\forall i \in \mathcal{V}$, while Figure 4.3(c) shows that the value of $x(t)^T W x(t)$ converges to $x^{*T} W x^*$, both agreeing with expectation.

4.6 Conclusion

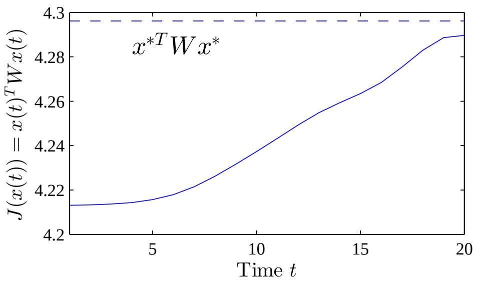
In this chapter, we have developed a class of continuous-time distributed algorithms, with which nodes in an undirected and connected graph can compute the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue, with only partial information about the graph and the matrix. In addition, explicit lower bounds on the convergence rate under slightly more restrictive condition has been derived.



(a) A 20-node graph.



(b) Node i's estimate $x_i(t)$ of x_i^* $\forall i \in \mathcal{V}$.



(c) Value of $x(t)^T W x(t)$.

Figure 4.3: Performance of continuous-time algorithm (4.12).

Chapter 5 Asynchronous Gossip Computation of the Perron-Frobenius Eigenvector

5.1 Introduction

In the previous chapter, we developed a class of continuous-time distributed algorithms, which enable each node i in an undirected and connected graph to compute the ith entry of the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue. The basic idea is to keep $x(t)$ constant while maximizing the value of $x(t)^T W x(t)$. It turns out that this idea can be extended to a discrete-time setting, leading to an asynchronous gossip algorithm for computing the Perron-Frobenius eigenvector, which is provably asymptotically convergent at an $O(\frac{1}{k})$ rate under a mild assumption on the gossiping pattern.

The outline of this chapter is as follows: Chapter 5.2 formulates the problem. Chapter 5.3 designs the gossip algorithm, while Chapter 5.4 analyzes its behavior. Chapter 5.5 provides simulation result for the gossip algorithm. Finally, Chapter 5.6 provides some concluding remarks. As before, throughout this chapter, for any $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, we write $x \geq 0$ if $x_i \geq 0$ for all i; $x > 0$ if $x \geq 0$ and $x \neq 0$; and $x \gg 0$ if $x_i > 0$ for all i. Moreover, we let $\mathbb{Z}_{\geq 0} = \{0, 1, \ldots\}$ denote the set of nonnegative integers.

5.2 Problem Formulation

Consider a network modeled as an undirected, connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ denotes the set of $N \geq 2$ nodes and $\mathcal{E} \subset \{\{i,j\} : i,j \in \mathcal{V}, i \neq j\}$ denotes the set of edges. Any two nodes $i,j \in \mathcal{V}$ are neighbors and can communicate if and only if $\{i,j\} \in \mathcal{E}$. The set of neighbors of each node $i \in \mathcal{V}$ is denoted as $\mathcal{N}_i = \{j \in \mathcal{V} : \{i,j\} \in \mathcal{E}\}$, and the communications are assumed to be delay- and error-free, with no quantization.

Suppose associated with the graph $\mathcal{G}$ is a square matrix $W = [w_{ij}] \in \mathbb{R}^{N \times N}$ satisfying the following assumption:

Assumption 5.1. The matrix W is such that: (i) for each $i,j \in \mathcal{V}$ with $i \neq j$, if $\{i,j\} \notin \mathcal{E}$, then $w_{ij} = w_{ji} = 0$; and (ii) for each $i,j \in \mathcal{V}$ with $i \neq j$, if $\{i,j\} \in \mathcal{E}$, then $w_{ij} = w_{ji} > 0$.

As was proved in Proposition 4.1, the matrix W has the following properties: (i) the largest eigenvalue of W, denoted as λ^*, is simple; (ii) the eigenspace associated with λ^*, denoted as $\mathcal{X}^*$, is one-dimensional and is of the form $\mathcal{X}^* = \{\alpha x^* : \alpha \in \mathbb{R}, \alpha \neq 0\}$, where $x^* \in \mathbb{R}^N$ and $x^* \gg 0$; and (iii) if $x > 0$ is an eigenvector of W, then $x \in \mathcal{X}^*$.

In addition to Assumption 5.1, we have another assumption on how neighboring nodes would gossip in the graph.

Assumption 5.2. There exists a positive integer B such that for each $k \in \mathbb{Z}_{\geq 0}$ and each $\{i,j\} \in \mathcal{E}$, there exists a time $\ell \in \{k, k+1, \ldots, k+B-1\}$ such that nodes i and j gossip at time ℓ.

Assumption 5.2 is mild as it can be satisfied by having, for example, every pair of neighboring nodes gossip sufficiently often.

As in Chapter 4, suppose each node $i \in \mathcal{V}$ knows only $\mathcal{N}_i$, w_{ii}, and w_{ij} $\forall j \in \mathcal{N}_i$, which it prefers to not share with any of its neighbors due perhaps to security and privacy reasons. Yet, despite having only such local information about graph $\mathcal{G}$ and matrix W, suppose every node $i \in \mathcal{V}$ wants to determine the largest eigenvalue λ^* of W and the ith entry x_i^* of an eigenvector x^* from the eigenspace $\mathcal{X}^*$. The goal of this chapter is to construct a gossip algorithm that enables every node $i \in \mathcal{V}$ to asymptotically determine the aforementioned λ^* and x_i^*.

5.3 Design of Gossip Algorithm

In Chapter 4.3, we designed a class of continuous-time distributed algorithms based on forcing $\|x(t)\|$ to be constant and making $x(t)^T W x(t)$ strictly increasing whenever possible. As it turns out, this idea may be extended to a gossip setting with a slightly more restrictive Assumption 5.1 instead of Assumption 4.1. In addition, we will show at the end of this section that a modified version of the gossip algorithm can be used under Assumption 4.1.

To demonstrate the design, let $k \in \mathbb{Z}_{\geq 0}$ denote discrete-time and $x_i(k)$ represent node i's estimate of x^* at time k. In addition, let

$$
\begin{aligned}
x(k) &= (x_1(k), x_2(k), \ldots, x_N(k)),\\
y(k) &= (y_1(k), y_2(k), \ldots, y_N(k)),\\
y(k) &= Wx(k),\\
J(k) &= x(k)^T W x(k),
\end{aligned}
$$

as before.

Suppose at each time $k \in \mathbb{Z}_{\geq 0}$, a single pair of neighboring nodes, say nodes i and j, gossip with each other and update their estimates from $x_i(k)$ and $x_j(k)$ to $x_i(k+1)$ and $x_j(k+1)$, while the rest of the N nodes remain idle, i.e., $x_\ell(k+1) = x_\ell(k)$ $\forall \ell \in \mathcal{V} - \{i,j\}$. Following the same idea as in the continuous-time case, the new vector $x(k+1)$ should be such that

$$\|x(k+1)\| = \|x(k)\|, \tag{5.1}$$

$$J(k+1) \geq J(k). \tag{5.2}$$

Since the rest of the N nodes do not update their estimates, (5.1) is equivalent to demanding that

$$x_i^2(k+1) + x_j^2(k+1) = x_i^2(k) + x_j^2(k). \tag{5.3}$$

On the other hand, one way to ensure (5.2) is to select $x_i(k+1)$ and $x_j(k+1)$ to maximize $J(k+1)$ subject to (5.3). Since this is an equality-constrained maximization problem, by the method of Lagrange multipliers, we have

$$\begin{bmatrix} \frac{\partial J(k+1)}{\partial x_i(k+1)} \\ \frac{\partial J(k+1)}{\partial x_j(k+1)} \end{bmatrix} = \alpha \begin{bmatrix} \frac{\partial (x_i^2(k+1)+x_j^2(k+1))}{\partial x_i(k+1)} \\ \frac{\partial (x_i^2(k+1)+x_j^2(k+1))}{\partial x_j(k+1)} \end{bmatrix}, \tag{5.4}$$

where $\alpha \in \mathbb{R}$ is the Lagrange multiplier. Clearly, (5.4) can be simplified to

$$\begin{bmatrix} y_i(k+1) \\ y_j(k+1) \end{bmatrix} = \alpha \begin{bmatrix} x_i(k+1) \\ x_j(k+1) \end{bmatrix}. \tag{5.5}$$

Although (5.3) and (5.5) provide only a set of necessary conditions for optimality, as Lemmas 5.1 and 5.2 below show, within the positive orthant these two equations have a unique solution, which happens to maximize $J(k+1)$ subject to (5.3):

Lemma 5.1. *If $x(k) \gg 0$, then within the positive orthant $(x_i(k+1), x_j(k+1)) \gg 0$, (5.3) and (5.5) have a unique solution, denoted as $(\chi_i, \chi_j, \alpha^*)$.*

Proof. Let $x(k) \gg 0$ and $c = \sqrt{x_i^2(k) + x_j^2(k)} > 0$. Consider the positive orthant $(x_i(k+1), x_j(k+1)) \gg 0$. In addition, let

$$f(x_i(k+1), x_j(k+1)) = \frac{y_i(k+1)}{x_i(k+1)} - \frac{y_j(k+1)}{x_j(k+1)}, \tag{5.6}$$

which is well-defined. Note that $f(x_i(k+1), x_j(k+1)) = 0$ if and only if (5.5) holds for some $\alpha \in \mathbb{R}$. Also, because $y = Wx$ and $x_\ell(k+1) = x_\ell(k)$ $\forall \ell \in \mathcal{V} - \{i, j\}$,

$$\begin{aligned} &f(x_i(k+1), x_j(k+1)) \\ &\quad = w_{ii} + \frac{w_{ij}x_j(k+1) + \sum_{\ell \in \mathcal{N}_i, \ell \neq j} w_{i\ell}x_\ell(k)}{x_i(k+1)} \\ &\quad\quad - w_{jj} - \frac{w_{ji}x_i(k+1) + \sum_{\ell \in \mathcal{N}_j, \ell \neq i} w_{j\ell}x_\ell(k)}{x_j(k+1)}. \end{aligned} \tag{5.7}$$

Note that as $x_i(k+1)$ increases from 0 to c, to satisfy (5.3) $x_j(k+1)$ must strictly and continuously decreases from c to 0. Also note that because $x(k) \gg 0$, $((x_i(k+1), x_j(k+1)) \gg 0$, and W is Metzler and irreducible, we have $w_{ij}x_j(k+1) + \sum_{\ell \in \mathcal{N}_i, \ell \neq j} w_{i\ell}x_\ell(k) > 0$ and $w_{ji}x_i(k+1) + \sum_{\ell \in \mathcal{N}_j, \ell \neq i} w_{j\ell}x_\ell(k) > 0$. Thus, as $x_i(k+1)$ increases from 0 to c, $f(x_i(k+1), x_j(k+1))$ in (5.7) strictly and continuously decreases from $+\infty$ to $-\infty$. It follows from the intermediate value theorem that there exists a unique $x_i(k+1) \in (0, c)$, denoted as χ_i, and the corresponding $x_j(k+1) \in (0, c)$, denoted as χ_j, such that $f(\chi_i, \chi_j) = 0$. This, along with (5.6), implies that (5.3) and (5.5) have a unique solution $(\chi_i, \chi_j) \gg 0$ and $\alpha^* \in \mathbb{R}$. □

Lemma 5.2. *If $x(k) \gg 0$, then $(\chi_i, \chi_j) \gg 0$ from Lemma 5.1 maximizes $J(k+1)$ subject to* (5.3).

Proof. Let $x(k) \gg 0$ and $(\chi_i, \chi_j) \gg 0$ be from Lemma 5.1. In addition, let $\chi \in \mathbb{R}^N$ be a vector whose entries i and j are χ_i and χ_j, respectively, and whose entry ℓ is $x_\ell(k)$ $\forall \ell \in \mathcal{V} - \{i,j\}$. Moreover, recall that $x_i(k+1)$ and $x_j(k+1)$ are the optimization variables, and that $x_\ell(k+1) = x_\ell(k)$ $\forall \ell \in \mathcal{V} - \{i,j\}$. Furthermore, let $f(z) = z^T W z$. With these notations, Lemma 5.2 is proved if we can show that $f(\chi) \geq f(x(k+1))$ for all $x_i(k+1)$ and $x_j(k+1)$ satisfying (5.3). To this end, let $\tilde{x} = \chi - x(k+1)$ and $\tilde{x}_\ell$ denote entry ℓ of $\tilde{x}$ $\forall \ell \in \mathcal{V}$. Then, since $\chi_i^2 + \chi_j^2 = x_i^2(k+1) + x_j^2(k+1) = x_i^2(k) + x_j^2(k)$, it is straightforward to show that

$$\tilde{x}_i^2 + \tilde{x}_j^2 = 2\chi_i \tilde{x}_i + 2\chi_j \tilde{x}_j. \tag{5.8}$$

Moreover, by using $W = W^T$, α^* from Lemma 5.1, (5.5), and (5.8), we can write

$$\begin{aligned}
&f(\chi) - f(x(k+1)) \\
&\quad = \chi^T W \chi - x(k+1)^T W x(k+1) \\
&\quad = (\chi - x(k+1))^T (W\chi + Wx(k+1)) \\
&\quad = \begin{bmatrix} \tilde{x}_i & \tilde{x}_j \end{bmatrix} \begin{bmatrix} 2\alpha^* \chi_i - w_{ii}\tilde{x}_i - w_{ij}\tilde{x}_j \\ 2\alpha^* \chi_j - w_{jj}\tilde{x}_j - w_{ji}\tilde{x}_i \end{bmatrix} \\
&\quad = \begin{bmatrix} \tilde{x}_i & \tilde{x}_j \end{bmatrix} \begin{bmatrix} \alpha^* - w_{ii} & -w_{ij} \\ -w_{ji} & \alpha^* - w_{jj} \end{bmatrix} \begin{bmatrix} \tilde{x}_i \\ \tilde{x}_j \end{bmatrix}.
\end{aligned} \tag{5.9}$$

Furthermore, due again to (5.5) and to $x(k) \gg 0$, $(\chi_i, \chi_j) \gg 0$, and W being Metzler,

$$\begin{aligned}
\alpha^* - w_{ii} &= \frac{w_{ij}\chi_j + d_i}{\chi_i} \geq 0, \\
\alpha^* - w_{jj} &= \frac{w_{ji}\chi_i + d_j}{\chi_j} \geq 0, \\
(\alpha^* - w_{ii})(\alpha^* - w_{jj}) - w_{ij}w_{ji} &= \frac{w_{ji}\chi_i d_i + w_{ij}\chi_j d_j + d_i d_j}{\chi_i \chi_j} \geq 0,
\end{aligned} \tag{5.10}$$

where $d_i = \sum_{\ell \in \mathcal{N}_i, \ell \neq j} w_{i\ell} x_\ell(k)$ and $d_j = \sum_{\ell \in \mathcal{N}_j, \ell \neq i} w_{j\ell} x_\ell(k)$. Thus, the 2-by-2 matrix in (5.9) is positive semidefinite, so that $f(\chi) \geq f(x(k+1))$ for all $x_i(k+1)$ and $x_j(k+1)$ satisfying (5.3), as desired. $\square$

Lemmas 5.1 and 5.2 call for a few remarks. First, at each time $k \in \mathbb{Z}_{\geq 0}$, the pair of gossiping nodes, say, nodes i and j, can let $x_i(k+1) = \chi_i$ and $x_j(k+1) = \chi_j$, where $(\chi_i, \chi_j) \gg 0$ is from Lemma 5.1. Second, as suggested in the proof of Lemma 5.1, χ_i and χ_j, which are both positive, may be computed by either node i or node j from

$$\begin{aligned} w_{ii} + \frac{w_{ij}\chi_j + \sum_{\ell \in \mathcal{N}_i, \ell \neq j} w_{i\ell} x_\ell(k)}{\chi_i} \\ = w_{jj} + \frac{w_{ji}\chi_i + \sum_{\ell \in \mathcal{N}_j, \ell \neq i} w_{j\ell} x_\ell(k)}{\chi_j} \end{aligned} \tag{5.11}$$

and

$$\chi_i^2 + \chi_j^2 = x_i^2(k) + x_j^2(k) \tag{5.12}$$

using, for example, the bisection method. Third, if $x(k) \gg 0$, then $x(k+1) \gg 0$, implying that if the initial estimates satisfy $x(0) \gg 0$, then all future estimates satisfy $x(k) \gg 0$ $\forall k \in \{1, 2, \ldots\}$. Finally, the above remarks suggest a gossip algorithm, with which the sequence $(J(0), J(1), J(2), \ldots)$ is non-decreasing and is, in fact, incrementally maximized by the gossiping nodes, according to Lemma 5.2. A complete description of this gossip algorithm, including its communication and computation aspects, is given below:

Algorithm 5.1 (Gossip Algorithm)**.**

Initialization:

1. Each node $i \in \mathcal{V}$ creates a variable $x_i \in \mathbb{R}$ and initializes it: $x_i \leftarrow$ random$(0, 1)$.

Operation: At each iteration:

2. A node, say, node $i \in \mathcal{V}$, initiates the iteration and selects a neighbor, say, node $j \in \mathcal{N}_i$, to gossip.

3. Node i transmits a message to every node $\ell \in \mathcal{N}_i, \ell \neq j$, requesting their x_ℓ's.
4. Node j transmits a message to every node $\ell \in \mathcal{N}_j, \ell \neq i$, requesting their x_ℓ's.
5. Node j transmits $\sum_{\ell \in \mathcal{N}_j, \ell \neq i} w_{j\ell} x_\ell$, w_{jj}, and x_j to node i.
6. Node i computes $(\chi_i, \chi_j) \gg 0$ from (5.11) and (5.12).
7. Node i updates x_i: $x_i \leftarrow \chi_i$.
8. Node i transmits χ_j to node j.
9. Node j updates x_j: $x_j \leftarrow \chi_j$. ■

In Step 1 of Algorithm 5.1, random$(0, 1)$ generates a random number strictly between 0 and 1, thus ensuring the required $x(0) \gg 0$. In Step 2, how nodes i and j are selected to gossip can be realized either randomly (e.g., equiprobably) or deterministically (e.g., periodically).

Remark 5.1. Note that when $\mathcal{G}$ is a complete graph and W is its adjacency matrix, Algorithm 5.1 becomes a pairwise equalizing algorithm, with which every pair of gossiping nodes i and j simply equalize their state variables while maintaining their sum-of-squares, i.e., $x_i(k+1) = x_j(k+1) = \sqrt{\frac{x_i(k)^2 + x_j(k)^2}{2}}$.

5.4 Analysis of Gossip Algorithm

In this section, we analyze the convergence of Algorithm 5.1 and characterize its convergence rate. The analysis is carried out under Assumption 5.1 and Assumption 5.2.

In addition, similar to Chapter 4.4.2, let $D \in \{1, 2, \ldots, N-1\}$ be the diameter of graph $\mathcal{G}$, $\lambda_N \leq \lambda_2 < \lambda^*$ be the smallest and second largest eigenvalue of W as before, and the constant $\bar{w}$, $\bar{w}'$, $\underline{w}$, β be as defined in (4.22)–(4.25).

Moreover, let $x(0) \gg 0$ be given and let $c = \|x(0)\|$. Thus, by Algorithm 5.1,

$$\|x^*\| = \|x(k)\| = c, \quad \forall k \in \mathbb{Z}_{\geq 0}, \tag{5.13}$$

$$0 \leq x_i(k) \leq c, \quad \forall i \in \mathcal{V},\ \forall k \in \mathbb{Z}_{\geq 0}. \tag{5.14}$$

The following lemma characterizes the convergence rate of the gossip algorithm.

Theorem 5.1. *Consider the network modeled in Chapter 5.2, and let Assumptions 5.1 and 5.2 hold. Suppose Algorithm 5.1 is used and the initial state satisfies $x(0) \gg 0$. Then, for each $k \in \mathbb{Z}_{\geq 0}$,*

$$V(x(k)) \leq \frac{V(x(0))}{V(x(0))\gamma' \left\lfloor \frac{k}{B} \right\rfloor + 1}, \tag{5.15}$$

where

$$\begin{aligned}
\gamma' &= \min\{\gamma'_1, \gamma'_2\} > 0, \\
\gamma'_1 &= \frac{2\underline{w}/(B^2c^2N\beta^{2D^2+2D})}{(\beta^{\frac{D^2+D}{2}}(\lambda^* - \lambda_N) + 4\bar{w}D\sqrt{N})^2}, \\
\gamma'_2 &= \frac{8\underline{w}(\bar{w}+\bar{w}')^2\left(\sqrt{1+\frac{\underline{w}(\bar{w}+2\bar{w}')/(\bar{w}+\bar{w}')}{4(2D+1)\beta^{D^2+D}N}} - 1\right)^2}{(\bar{w}+2\bar{w}')^2B^2c^2(\lambda^* - \lambda_N)}.
\end{aligned} \tag{5.16}$$

In addition,

$$\|x(k) - x^*\| \leq \sqrt{\frac{4V(x(0))}{(\lambda^* - \lambda_2)(V(x(0))\gamma' \left\lfloor \frac{k}{B} \right\rfloor + 1)}}. \tag{5.17}$$

To prove Theorem 5.1, we first establish the following lemmas:

Lemma 5.3. *Suppose Assumption 5.1 holds. For each $k \in \mathbb{Z}_{\geq 0}$, if $\|x(k)\| = \|x^*\|$ and $x(k) > 0$, then*

$$\|x(k) - x^*\| \leq \sqrt{\frac{4V(x(k))}{\lambda^* - \lambda_2}}. \tag{5.18}$$

Proof. Replacing t by k in the proof of Lemma 4.4 , we obtain the (5.18). $\square$

Similar to the proof in Chapter 4.4.2, we focus on $V(x(k))$ in the following lemmas:

Lemma 5.4. *Consider the setup of Theorem 5.1. Then, for each $k \in \mathbb{Z}_{\geq 0}$,*

$$V(x(k)) - V(x(k+1)) \geq \underline{w}\|x(k+1) - x(k)\|^2.$$

Proof. Let $k \in \mathbb{Z}_{\geq 0}$ be given and let nodes i and j be the pair of gossiping nodes at time k. In addition, let $\tilde{x} = x(k+1) - x(k)$ and $\tilde{x}_\ell$ denote entry ℓ of $\tilde{x}$ $\forall \ell \in \mathcal{V}$. Then, in view of (4.14), (5.10), and Assumption 5.1, we can write

$$\begin{aligned}
&V(x(k)) - V((k+1)) \\
&\quad= \begin{bmatrix}\tilde{x}_i & \tilde{x}_j\end{bmatrix} \begin{bmatrix}\alpha^* - w_{ii} & -w_{ij} \\ -w_{ji} & \alpha^* - w_{jj}\end{bmatrix} \begin{bmatrix}\tilde{x}_i \\ \tilde{x}_j\end{bmatrix} \\
&\quad\geq \underline{w} \begin{bmatrix}\tilde{x}_i & \tilde{x}_j\end{bmatrix} \begin{bmatrix}\frac{x_j(k+1)}{x_i(k+1)} & -1 \\ -1 & \frac{x_i(k+1)}{x_j(k+1)}\end{bmatrix} \begin{bmatrix}\tilde{x}_i \\ \tilde{x}_j\end{bmatrix} \\
&\quad= \underline{w}\frac{(x_i(k+1)x_j(k) - x_j(k+1)x_i(k))^2}{x_i(k+1)x_j(k+1)} \\
&\quad\geq \underline{w}\big((x_i(k+1) - x_i(k))^2 + (x_j(k+1) - x_j(k))^2\big) \\
&\quad= \underline{w}\|x(k+1) - x(k)\|^2.
\end{aligned}$$

$\square$

Next, for convenience, let

$$\underline{x}(k) = \min_{i \in \mathcal{V}} x_i(k) > 0, \quad \forall k \in \mathbb{Z}_{\geq 0}, \tag{5.19}$$

$$\bar{x}_i(k) = \max_{j \in \mathcal{N}_i} x_j(k) \leq c, \quad \forall i \in \mathcal{V},\ \forall k \in \mathbb{Z}_{\geq 0}, \tag{5.20}$$

$$z_i(k) = \frac{y_i(k)}{x_i(k)}, \quad \forall i \in \mathcal{V},\ \forall k \in \mathbb{Z}_{\geq 0}, \tag{5.21}$$

$$\tilde{z}_i(k) = z_i(k) - \lambda^*, \quad \forall i \in \mathcal{V},\ \forall k \in \mathbb{Z}_{\geq 0}. \tag{5.22}$$

Since $x(0) \gg 0$ and due to Algorithm 5.1, $x(k) \gg 0$ $\forall k \in \mathbb{Z}_{\geq 0}$. Thus, the inequality in (5.19) holds. Therefore, $z_i(k)$ and $\tilde{z}_i(k)$ in (5.21) and (5.22) are well-defined. Because of (5.14), inequality in (5.20) holds.

Lemma 5.5. *Consider the setup of Theorem 5.1. For each $k \in \mathbb{Z}_{\geq 0}$, if $k_0 \in \{k, k+1, \dots B-1\}$, then*

$$x_i(k) - B\theta \leq x_i(k_0+1) \leq x_i(k) + B\theta, \quad \forall i \in \mathcal{V}, \tag{5.23}$$

where

$$\theta = \sqrt{\frac{2(V(x(k)) - V(x(k+B)))}{\underline{w}}} \geq 0. \tag{5.24}$$

Proof. Let $k \in \mathbb{Z}_{\geq 0}$ be given. Due to Algorithm 5.1, $V(x(k)) \geq V(x(k+1)) \geq \cdots \geq V(x(k+B))$ $\forall k \in \mathbb{Z}_{\geq 0}$, so that θ is well-defined and satisfies $\theta \geq 0$. Let $k_0 \in \{k, k+1, \dots B-1\}$ be given. By Lemma 5.4, $\forall m \in \{k, k+1, \dots, k_0\}$,

$$\begin{aligned} V(x(k)) - V(x(k+B)) &\geq V(x(m)) - V(x(m+1)) \\ &\geq \frac{1}{2}\underline{w}\|x(m) - x(m+1)\|^2 \\ &\geq \frac{1}{2}\underline{w}|x_i(m) - x_i(m+1)|^2, \quad \forall i \in \mathcal{V}. \end{aligned} \tag{5.25}$$

This, together with (5.24), implies that $\forall i \in \mathcal{V}$,

$$\begin{aligned} &|x_i(k_0+1) - x_i(k)| \\ &\quad \leq \sum_{m \in \{k, k+1, \dots, k_0\}} |x_i(m+1) - x_i(m)| \\ &\quad \leq (k_0 - k + 1)\theta \leq B\theta. \end{aligned} \tag{5.26}$$

Combining (5.25) and (5.26) yields (5.24). □

Lemma 5.6. *Consider the setup of Theorem 5.1. For each $k \in \mathbb{Z}_{\geq 0}$,*

$$V(x(k)) - V(x(k+B))$$

$$\geq \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\underline{x}(k)V(x(k)) + 2\bar{w}c^3D)^2}V(x(k))^2. \tag{5.27}$$

Proof. Let $k \in \mathbb{Z}_{\geq 0}$ be given. In addition, let $p \in \arg\max_{i\in\mathcal{V}} \tilde{z}_i(k)$ and $q \in \arg\min_{i\in\mathcal{V}, i\neq p} \tilde{z}_i(k)$. Then, either $\tilde{z}_p(k) = \tilde{z}_q(k)$ or $\tilde{z}_p(k) > \tilde{z}_q(k)$. First, suppose $\tilde{z}_p(k) = \tilde{z}_q(k)$. Then, $\tilde{z}_i(k)\ \forall i \in \mathcal{V}$ are equal, and so do $z_i(k)\ \forall i \in \mathcal{V}$. Let $z_i(k) = \lambda \in \mathbb{R}$, $\forall i \in \mathcal{V}$. This, together with (5.21) and (4.1), implies that $\lambda x(k) = y(k) = Wx(k)$, so that λ is an eigenvalue of W with $x(k)$ being its eigenvector. Since $x(k) \gg 0$ and by the Perron-Frobenius theorem, $\lambda = \lambda^*$ and $x(k) = x^*$. Thus, $V(x(k)) = 0$. Since $V(x(k)) - V(x(k+B)) \geq 0$ by Algorithm 5.1, (5.27) holds.

Next, suppose $\tilde{z}_p(k) > \tilde{z}_q(k)$. Then, since graph $\mathcal{G}$ is connected, there exists a shortest path between nodes p and q, whose length is less than or equal to D. Thus, there exists $\{r, s\} \in \mathcal{E}$ lying on that shortest path such that $\tilde{z}_r(k) - \tilde{z}_s(k) \geq \frac{1}{D}(\tilde{z}_p(k) - \tilde{z}_q(k)) > 0$. By the Perron-Frobenius theorem and definition of $\tilde{z}_p(k)$ and $\tilde{z}_q(k)$, we have $\tilde{z}_p(k) > 0$ and $\tilde{z}_q(k) < 0$. It follows that

$$\tilde{z}_r(k) - \tilde{z}_s(k) > -\frac{1}{D}\tilde{z}_q(k) > 0. \tag{5.28}$$

Since (5.13), (4.14) and (5.28),

$$\begin{aligned} V(x(k)) &= \frac{1}{2}(x^{*T}Wx^* - x(k)^TWx(k)) \\ &= \frac{1}{2}\Big(\lambda^* c^2 - \sum_{i\in\mathcal{V}} z_i(k)x_i(k)^2\Big) \\ &= \frac{1}{2}\Big(\lambda^* c^2 - \sum_{i\in\mathcal{V}} (\tilde{z}_i(k) + \lambda^*)x_i(k)^2\Big) \\ &= -\frac{1}{2}\Big(\sum_{i\in\mathcal{V}} \tilde{z}_i(k)x_i(k)^2\Big) \\ &\leq -\frac{1}{2}\tilde{z}_q(k)c^2 \leq \frac{1}{2}c^2D(\tilde{z}_r(k) - \tilde{z}_s(k)). \end{aligned} \tag{5.29}$$

By Assumption 5.2, there exists $k_0 \in \{k, k+1, \ldots, k+B-1\}$ such that nodes r and s gossip at time k_0. Thus, from Algorithm 5.1,

$$z_r(k_0+1) = z_s(k_0+1). \tag{5.30}$$

Since $k_0 \in \{k, k+1, \ldots, k+B-1\}$, by Lemma 5.5, (5.23) holds. We next show that

$$\theta \geq \frac{\underline{x}(k)^2}{B(\underline{x}(k) + 4\bar{w}c/(\tilde{z}_r(k) - \tilde{z}_s(k)))}. \tag{5.31}$$

Assume, to the contrary, that (5.31) does not hold. Note from (5.23) and (5.14) that if $\theta = 0$, then $x_i(k_0+1) = x_i(k) > 0\ \forall i \in \mathcal{V}$. Thus, $z_r(k_0+1) - z_s(k_0+1) = z_r(k) - z_s(k) = \tilde{z}_r(k) - \tilde{z}_s(k) > 0$, which contradicts (5.30). Now suppose $0 < \theta < \frac{\underline{x}(k)^2}{B(\underline{x}(k)+4\bar{w}c/(\tilde{z}_r(k)-\tilde{z}_s(k)))}$. This, along with (5.28), implies that $\theta < \frac{\underline{x}(k)}{B}$, so that $\underline{x}(k) - B\theta > 0$ and $x_i(k) - B\theta > 0\ \forall i \in \mathcal{V}$. This, together with (5.23), (5.13), (5.19), (5.22), (4.1), and (4.22), implies that

$$\begin{aligned}
\tilde{z}_r(k) - \tilde{z}_r(k_0+1) &= \frac{y_r(k)}{x_r(k)} - \frac{y_r(k_0+1)}{x_r(k_0+1)} \\
&= \frac{\sum_{j\in\mathcal{N}_r} w_{rj}x_j(k)}{x_r(k)} - \frac{\sum_{j\in\mathcal{N}_r} w_{rj}x_j(k_0+1)}{x_r(k_0+1)} \\
&\leq \frac{\sum_{j\in\mathcal{N}_r} w_{rj}x_j(k)}{x_r(k)} - \frac{\sum_{j\in\mathcal{N}_r} w_{rj}(x_j(k) - B\theta)}{x_r(k) + B\theta} \\
&= \frac{B\theta(\sum_{j\in\mathcal{N}_r} w_{rj}x_j(k) + x_r(k)\sum_{j\in\mathcal{N}_r} w_{rj})}{x_r(k)(x_r(k)+B\theta)} \\
&< \frac{2\bar{w}c}{\underline{x}(k)^2/(B\theta) + \underline{x}(k)} < \frac{\tilde{z}_r(k) - \tilde{z}_s(k)}{2}.
\end{aligned} \tag{5.32}$$

Similarly,

$$\begin{aligned}
\tilde{z}_s(k_0+1) - \tilde{z}_s(k) &= \frac{y_s(k_0+1)}{x_s(k_0+1)} - \frac{y_s(k)}{x_s(k)} \\
&= \frac{\sum_{j\in\mathcal{N}_s} w_{sj}x_j(k_0+1)}{x_s(k_0+1)} - \frac{\sum_{j\in\mathcal{N}_s} w_{sj}x_j(k)}{x_s(k)}
\end{aligned}$$

$$\begin{aligned}
&\le \frac{\sum_{j\in\mathcal{N}_s} w_{sj}(x_j(k)+B\theta)}{x_s(k)-B\theta} - \frac{\sum_{j\in\mathcal{N}_s} w_{sj}x_j(k)}{x_s(k)} \\
&= \frac{B\theta(\sum_{j\in\mathcal{N}_s} w_{sj}x_j(k) + x_s(k)\sum_{j\in\mathcal{N}_s} w_{sj})}{x_s(k)(x_s(k)-B\theta)} \\
&< \frac{2\bar{w}c}{\underline{x}(k)^2/(B\theta)-\underline{x}(k)} \le \frac{\tilde{z}_r(k)-\tilde{z}_s(k)}{2}. \qquad (5.33)
\end{aligned}$$

Adding both sides of (5.32) and (5.33), we obtain $\tilde{z}_r(k_0+1)-\tilde{z}_s(k_0+1)>0$. This, along with (5.22), implies that $z_r(k_0+1)-z_s(k_0+1) = \tilde{z}_r(k_0+1) - \tilde{z}_s(k_0+1) > 0$, which contradicts (5.30). Thus, (5.31) holds.

Now, replacing $\tilde{z}_r(k)-\tilde{z}_s(k)$ and θ in (5.31) with (5.29) and (5.24), we obtain

$$\begin{aligned}
&V(x(k)) - V(x(k+B)) \\
&\quad\ge \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\underline{x}(k)+2\bar{w}c^3D/(V(x(k))))^2} \\
&\quad= \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\underline{x}(k)V(x(k))+2\bar{w}c^3D)^2}V(x(k))^2,
\end{aligned}$$

as desired. □

Lemma 5.7. *Suppose Assumption 5.1 holds. For each $k\in\mathbb{Z}_{\ge 0}$, if $\|x(k)\|=c$, $x(k)>0$, and*

$$\underline{x}(k) < \frac{1}{\beta^{\frac{D^2+D}{2}}}\frac{c}{\sqrt{N}}, \qquad (5.34)$$

then there exists $\{u,v\}\in\mathcal{E}$ such that

$$\begin{aligned}
&w_{uv}x_u(k)^2 - \bar{w}\bar{x}_u x_v(k) - (|w_{uu}|+|w_{vv}|)x_u(k)x_v(k) \\
&\quad\ge \frac{\underline{w}c^2}{(2D+1)\beta^{D^2+D}N}. \qquad (5.35)
\end{aligned}$$

Proof. Replace t in the proof of Lemma 4.7 by k, we establish the lemma. □

Lemma 5.8. *Consider the setup of Theorem 5.1. For each $k \in \mathbb{Z}_{\geq 0}$, if* (5.34) *holds, then*

$$\begin{aligned} &V(x(k)) - V(x(k+B)) \\ &\quad \geq \Big(\sqrt{1 + \frac{\underline{w}(\bar{w} + 2\bar{w}')/(\bar{w} + \bar{w}')}{4(2D+1)\beta^{D^2+D}N}} - 1\Big)^2 \frac{4\underline{w}(\bar{w} + \bar{w}')^2 c^2}{(\bar{w} + 2\bar{w}')^2 B^2}. \end{aligned} \tag{5.36}$$

Proof. Let $k \in \mathbb{Z}_{\geq 0}$ be given and suppose (5.34) holds. By Lemma 5.7, there exists $\{u, v\} \in \mathcal{E}$ such that (5.35) holds. By Assumption 5.2, there exists $k_0 \in \{k, k+1, \ldots, k+B-1\}$ such that nodes u and v gossip at time k_0. From Algorithm 5.1,

$$y_v(k_0+1)x_u(k_0+1) - y_u(k_0+1)x_v(k_0+1) = 0. \tag{5.37}$$

In addition, since $k_0 \in \{k, k+1, \ldots, k+B-1\}$, (5.23) holds. Due to (5.20) and (5.23),

$$\begin{aligned} \bar{x}_u(k_0+1) &= \max_{j \in \mathcal{N}_u} x_j(k_0+1) \\ &\leq \max_{j \in \mathcal{N}_u} (x_j(k) + B\theta) = \bar{x}_u(k) + B\theta. \end{aligned}$$

This, together with Assumption 5.1, (4.1), (5.13), (5.23), (5.35), and (4.22), implies that

$$\begin{aligned} &y_v(k_0+1)x_u(k_0+1) - y_u(k_0+1)x_v(k_0+1) \\ &\quad = \Big(w_{vv}x_v(k_0+1) + \sum_{j \in \mathcal{N}_v} w_{jv}x_j(k_0+1)\Big)x_u(k_0+1) \\ &\qquad - \Big(w_{uu}x_u(k_0+1) + \sum_{j \in \mathcal{N}_u} w_{ju}x_j(k_0+1)\Big)x_v(k_0+1) \\ &\quad \geq (-|w_{vv}x_v(k_0+1)| + w_{uv}x_u(k_0+1))x_u(k_0+1) \\ &\qquad - (|w_{uu}|x_u(k_0+1) + \bar{w}\bar{x}_u(k_0+1))x_v(k_0+1) \\ &\quad = w_{uv}x_u(k_0+1)^2 - \bar{w}\bar{x}_u(k_0+1)x_v(k_0+1) \end{aligned}$$

$$\begin{aligned}
&- (|w_{uu}| + |w_{vv}|)x_u(k_0+1)x_v(k_0+1) \\
&\geq w_{uv}(x_u(k) - B\theta)^2 - \bar{w}(\bar{x}_u(k) + B\theta)(x_v(k) + B\theta) \\
&- (|w_{uu}| + |w_{vv}|)(x_u(k) + B\theta)(x_v(k) + B\theta) \\
&= w_{uv}x_u(k)^2 - \bar{w}\bar{x}_u(k)x_v(k) - (|w_{uu}| + |w_{vv}|)x_u(k)x_v(k) \\
&- B\theta(2w_{uv}x_u(k) + \bar{w}(\bar{x}_u(k) + x_v(k)) \\
&+ (|w_{uu}| + |w_{vv}|)(x_u(k) + x_v(k))) \\
&- B^2\theta^2(-w_{uv} + \bar{w} + |w_{uu}| + |w_{vv}|) \\
&\geq \frac{\underline{w}c^2}{(2D+1)\beta^{D^2+D}N} - 2B\theta c(w_{uv} + \bar{w} + |w_{uu}| + |w_{vv}|) \\
&- B^2\theta^2(-w_{uv} + \bar{w} + |w_{uu}| + |w_{vv}|) \\
&\geq \frac{\underline{w}c^2}{(2D+1)\beta^{D^2+D}N} - 4B\theta c(\bar{w} + \bar{w}') - B^2\theta^2(\bar{w} + 2\bar{w}').
\end{aligned}$$

This, along with (5.37) and (5.24), implies that

$$\theta \geq \Big(\sqrt{1 + \frac{\underline{w}(\bar{w} + 2\bar{w}')/(\bar{w} + \bar{w}')}{4(2D+1)\beta^{D^2+D}N}} - 1\Big)\frac{2(\bar{w} + \bar{w}')c}{(\bar{w} + 2\bar{w}')B}.$$

Due again to (5.24), we obtain (5.36). □

With the above lemmas in hand, we now prove the Theorem 5.1.

Proof of Theorem 5.1. First, we show that $\forall k \in \mathbb{Z}_{\geq 0}$,

$$V(x(k)) - V(x(k+B)) \geq \gamma' V(x(k))^2, \tag{5.38}$$

where γ' is as defined in Theorem 5.1 and satisfies

$$\frac{1}{8V(x(k))} \geq \gamma' > 0. \tag{5.39}$$

Let $k \in \mathbb{Z}_{\geq 0}$ be given. Due to (5.13) and the Rayleigh quotient,

$$V(x(k)) \leq \frac{1}{2}(\lambda^* c^2 - \lambda_N c^2). \tag{5.40}$$

Because of (5.40) and since $\lambda^* > \lambda_N$,

$$0 < \frac{4V(x(k))^2}{(\lambda^* - \lambda_N)^2 c^4} \leq 1. \tag{5.41}$$

Consider the following two cases: (i) $\underline{x}(k) \geq \frac{1}{\beta^{\frac{D^2+D}{2}}}\frac{c}{\sqrt{N}}$; (ii) $\underline{x}(k) < \frac{1}{\beta^{\frac{D^2+D}{2}}}\frac{c}{\sqrt{N}}$. For case (i), due to Lemma 5.6 and (5.40) and since $\lambda^* > \lambda_N$,

$$\begin{aligned}
&V(x(k)) - V(x(k+B)) \\
&\quad\geq \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\underline{x}(k)V(x(k)) + 2\bar{w}c^3D)^2}V(x(k))^2 \\
&\quad\geq \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\frac{1}{2}\underline{x}(k)(\lambda^* - \lambda_N)c^2 + 2\bar{w}c^3D)^2}V(x(k))^2. \\
&\quad\geq \underbrace{\frac{2\underline{w}/(B^2c^2N\beta^{2D^2+2D})}{(\beta^{\frac{D^2+D}{2}}(\lambda^* - \lambda_N) + 4\bar{w}D\sqrt{N})^2}}_{\gamma_1'}V(x(k))^2.
\end{aligned} \tag{5.42}$$

From (5.42), (4.22), (4.25), and (4.24), and because $\underline{x}(k) \leq c$, $\lambda^* > \lambda_N$, $B \geq 1$, $D \geq 1$, we have

$$\begin{aligned}
0 < \gamma_1' &\leq \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(\frac{1}{2}\underline{x}(k)(\lambda^* - \lambda_N)c^2 + 2\bar{w}c^3D)^2} \\
&\leq \frac{\underline{w}\ \underline{x}(k)^4}{2B^2(2\underline{x}(k)(\lambda^* - \lambda_N)c^2\bar{w}c^3D)} \\
&\leq \frac{1}{4(\lambda^* - \lambda_N)c^2}.
\end{aligned} \tag{5.43}$$

Due to (5.40) and (5.43),

$$0 < \gamma_1' \leq \frac{1}{8V(x(k))}. \tag{5.44}$$

For case (ii), due to Lemma 5.8 and (5.41),

$$\begin{aligned}
&V(x(k)) - V(x(k+B)) \\
&\quad\geq \underbrace{\frac{8\underline{w}(\bar{w}+\bar{w}')^2\left(\sqrt{1+\frac{\underline{w}(\bar{w}+2\bar{w}')/(\bar{w}+\bar{w}')}{4(2D+1)\beta^{D^2+D}N}} - 1\right)^2}{(\bar{w}+2\bar{w}')^2B^2c^2(\lambda^* - \lambda_N)}}_{\gamma_2'}V(x(k))^2.
\end{aligned} \tag{5.45}$$

Clearly, $\gamma_2' > 0$. Combining this with (5.42), (5.45), (5.44), and (5.16), we obtain (5.38) and (5.39).

Next, we show by induction that $\forall k' \in \mathbb{Z}_{\geq 0}$,

$$V(x(k'B)) \leq \frac{V(x(0))}{V(x(0))\gamma' k' + 1}. \tag{5.46}$$

First, (5.46) holds for $k' = 0$. Next, suppose (5.46) holds for some $k_0' \geq 0$, i.e.,

$$V(x(k_0'B)) \leq \frac{V(x(0))}{V(x(0))\gamma' k_0' + 1}. \tag{5.47}$$

Due to (5.38), $V(x(k_0'B)) \leq \frac{1}{8\gamma'} < \frac{1}{2\gamma'}$. This, along with (5.38) and (5.47), implies that

$$\begin{aligned}
&V(x((k_0'+1)B)) \\
&\quad \leq V(x(k_0'B)) - \gamma' V(x(k_0'B))^2 \\
&\quad \leq \frac{V(x(0))}{V(x(0))\gamma' k_0' + 1} - \gamma'(\frac{V(x(0))}{V(x(0))\gamma' k_0' + 1})^2 \\
&\quad = \frac{V(x(0))(V(x(0))\gamma' k_0' + 1) - V(x(0))^2\gamma'}{(V(x(0))\gamma' k_0' + 1)^2} \\
&\quad = \frac{V(x(0))}{V(x(0))\gamma'(k_0'+1) + 1} \\
&\qquad - \frac{V(x(0))^3\gamma'^2 + 2V(x(0))^2\gamma'}{(V(x(0))\gamma' k_0' + 1)^2(V(x(0))\gamma'(k_0'+1)+1)} \\
&\quad \leq \frac{V(x(0))}{V(x(0))\gamma'(k_0'+1) + 1}.
\end{aligned}$$

It follows that (5.46) holds for $k_0' + 1$. Therefore, (5.46) holds.

Finally, let $k \in \mathbb{Z}_{\geq 0}$ be given. Due to Algorithm 5.1 and definition of $\lfloor \cdot \rfloor$,

$$V(x(k)) \leq V(x(\left\lfloor \tfrac{k}{B} \right\rfloor B)). \tag{5.48}$$

Applying (5.46) to (5.48), we obtain (5.15). From (5.15) and Lemma 5.3, we obtain (5.17). □

Remark 5.2. A slightly modified version of Algorithm 5.1 can used with Assumption 4.1. To see this, suppose $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is as given in Chapter 4.2 and Assumption 4.1 holds. Let $W' = W$ and $\mathcal{G}' = \{\mathcal{V}', \mathcal{E}'\}$, where $\mathcal{V}' = \mathcal{V}$ and $\mathcal{E}' = \{\{i, j\} : \{i, j\} \in \mathcal{E} \quad \text{and} \quad w_{ij} > 0\}$. Then, $\mathcal{G}'$ is connected since W is irreducible. Thus, W' satisfies Assumption 5.1, so that Algorithm 5.1 can be used for $\mathcal{G}'$ and W' to find the eigenvector x^* of W', which is also the eigenvector of W. Therefore, we can let step 2 of Algorithm 5.1 be such that node i only picks its neighbor $j \in \mathcal{N}_i$ to gossip if $w_{ij} > 0$. This algorithm is always well-defined since W satisfies Assumption 4.1.

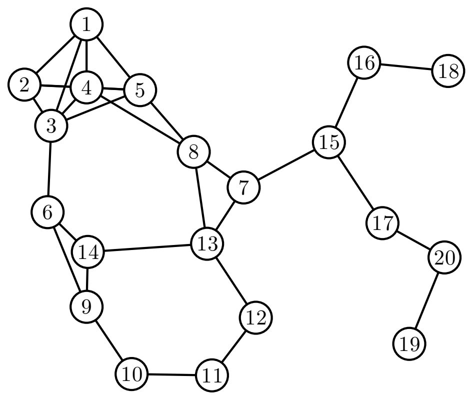
5.5 Simulation Results

Consider a network with $N = 20$ nodes, modeled as an undirected, connected graph $\mathcal{G}$, whose topology is shown in Figure 5.1(a). Suppose W is the adjacency matrix of $\mathcal{G}$, whose entries satisfy Assumption 5.1. Also suppose Algorithm 5.1 is used, where each node $i \in \mathcal{V}$ lets its initial state satisfying $x_i(0) > 0$, and each pair of nodes i and j with $\{i, j\} \in \mathcal{E}$ is selected to gossip periodically, so that B in Assumption 5.2 can be taken as $B = |\mathcal{E}|$.

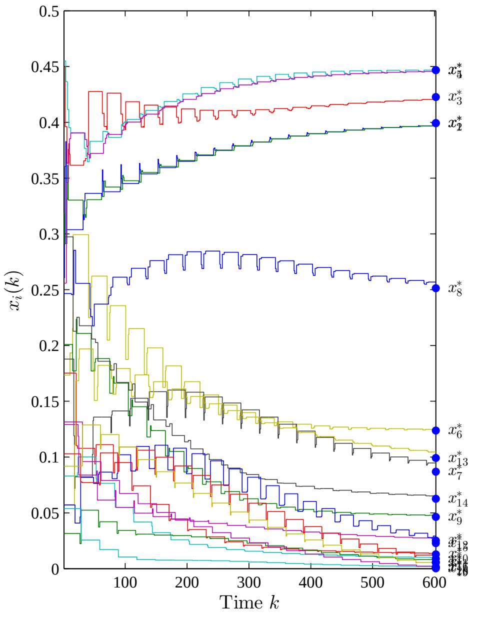
Figures 5.1(b) and 5.1(c) display the simulation result. Specifically, Figure 5.1(b) shows that node i's estimate $x_i(k)$ asymptotically converges to x_i^* $\forall i \in \mathcal{V}$. Figure 5.1(c) shows that the value of $x(k)^T W x(k)$increases monotonically and converges to $x^{*T} W x^*$. Again, the result agrees with expectation.

5.6 Conclusion

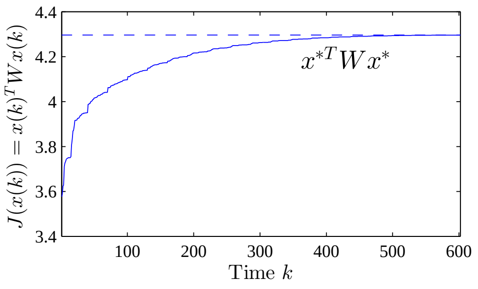
In this chapter, we have developed a gossip algorithm, with which nodes in an undirected and connected graph can compute the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue, with only partial information about the graph and the matrix. In addition, we have derived a lower bound on its convergence rate under a slightly more restrictive condition.



(a) A 20-node graph.



(b) Node i's estimates $x_i(k)$of $x_i^* \ \forall i \in \mathcal{V}$.



(c) Value of $x(k)^T W x(k)$.

Figure 5.1: Performance of gossip algorithm.

Chapter 6 Conclusions

6.1 Overall Summary

In this dissertation, we have developed a novel collection of control theory-based distributed algorithms, which enable nodes in a graph (representing, for example, a communication/social/transportation/power network) to cooperatively compute several quantities of common interest, with only local interaction and without any centralized coordination. These algorithms can be used to: (i) estimate the spectrum of a graph, thus allowing the nodes to infer about the graph structure; (ii) calculate the solution to a general system of linear equations that arise in a variety of ways; and (iii) compute the Perron-Frobenius eigenvector, thus allowing the nodes to determine their eigenvector centrality representing their relative importance in the graph.

More specifically, in Chapter 2, a two-stage distributed algorithm has been constructed, with which nodes can jointly estimate the spectrum of a matrix associated with the graph. In the first stage, the algorithm uses a discrete-time linear iteration and the Cayley-Hamilton theorem to convert the problem into one of solving a set of linear equations, where each equation is known to a node. In the second stage, if the nodes happen to know that said matrix is cyclic, the algorithm uses a Lyapunov approach to asymptotically solve the equations with an exponential rate of convergence. If they do not know whether said matrix is cyclic, the algorithm uses a random perturbation approach and a structural controllability result to approximately solve the equations with an error that can be made small.

In Chapter 3, a continuous-time distributed algorithm has been developed that allows nodes in an undirected and connected graph to collaboratively solve a general system of linear equations, where the only assumption is that each equation is known to at least one node. We have shown that the algorithm enables the nodes to asymptotically agree on a solution when there are infinitely many solutions, determine the solution when there is exactly one, and discover that no solution exists when there are none. In addition, we have proved that the algorithm is globally exponentially convergent, derived an explicit lower bound on its convergence rate, and shown that under certain conditions, the larger the graph's algebraic connectivity, or the further away from being singular the system of equations, the larger this lower bound.

In Chapter 4, a class of continuous-time distributed algorithms have been devised, which enable each node i in an undirected and connected graph to compute the ith entry of the Perron-Frobenius eigenvector of a symmetric, Metzler, and irreducible matrix associated with the graph, as well as the corresponding eigenvalue, when node i knows only row i of the matrix. We have shown that each continuous-time distributed algorithm in the class is a nonlinear networked dynamical system with a skew-symmetric structure, whose state is guaranteed to stay on a sphere, remain nonnegative, and converge asymptotically to the said eigenvector at an $O(\frac{1}{t})$ rate. In addition, in Chapter 5, we have shown that the same idea that yields the continuous-time algorithms can be extended to a discrete-time setting, leading to an asynchronous gossip algorithm for computing the Perron-Frobenius eigenvector, which has been proven to be asymptotically convergent at an $O(\frac{1}{k})$ rate under a mild assumption on the gossiping pattern.

6.2 Future Work

In our opinion, distributed computation over networks will remain a vibrant area of research for years to come. In terms of the three topics addressed in this dissertation, we see the following possible future work:

- *Making the graph spectrum estimation algorithm converge faster in large networks*: Although the two-stage algorithm we developed in Chapter 2 is capable of estimating, at least in theory, the spectrum of a graph of arbitrary size as long as the required assumptions are met, its actual convergence rate may decrease substantially as the size of the graph grows. Indeed, this phenomenon has been observed in our simulation results. An explanation for the phenomenon is that the matrix constructed in the first stage of the algorithm tends to be poorly conditioned or nearly singular in a large network, triggering a very slow convergence rate in the second stage. Therefore, a possible future research direction is to come up with a brand new distributed algorithm, which completely sidesteps the need to form a potentially poorly-conditioned or nearly-singular matrix, or quickly solves the resulting linear equations despite the matrix having these issues.

- *Extension of the continuous-time eigenvector centrality computation algorithm to handle non-symmetric matrices and directed graphs*: The class of continuous-time distributed algorithms we developed in Chapter 4 is able to compute the Perron-Frobenius eigenvector of *any* symmetric, Metzler, and irreducible matrix associated with *any* undirected graph. In particular, the symmetricity allows us to introduce a Lyapunov function candidate for algorithm derivation and analysis. Given that non-symmetric matrices and directed graphs not uncommon in applications, another worthy future research direction is to extend our results to the following two cases: (i) the graph is still undirected but the matrix may not be symmetric; and (ii) the graph is directed so that the matrix is necessarily non-symmetric. For case (i), our simulation results indicate that the continuous-time algorithms would still converge to the right point, suggesting that what is needed may just be a new way of establishing their convergence, using perhaps a new Lyapunov function candidate. For case (ii), a reasonable starting point is to consider forcing the state to remain on the unit sphere and nonnegative orthant despite having directed information flows, and then trying to drive the state toward the eigenvector.

- *Eliminating the "neighboring nodes must gossip sufficiently often" assumption needed by the gossip eigenvector centrality computation algorithm*: The gossip algorithm we developed in Chapter 5 is guaranteed to converge if there exists a finite constant B such that every pair of neighboring nodes gossip at least once per B iterations. While this assumption is often met in practice, it is of interest to see whether the gossip algorithm would still manage to converge without the assumption. In distributed consensus, removal of such an assumption has been shown to be possible. Thus, it is reasonable to conjecture that the same can be done for the gossip algorithm.

- *Analysis of the algorithms in the presence of communication delays, errors, and quantization*: In this dissertation, for simplicity we have assumed that all communications between neighboring nodes are ideal, i.e., there are no delays and errors, nor quantization. Since internode communications are not ideal in many real networks (especially wireless networks), one possible future research thrust is to investigate the impact of such non-idealities on algorithm performance, such as quantifying how the size of communication delays and errors and the level of quantization affect the steady-state accuracy of the algorithm estimates.

Bibliography

[1] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[2] J. Cortés, "Finite-time convergent gradient flows with applications to network consensus," *Automatica*, vol. 42, no. 11, pp. 1993–2000, 2006.

[3] A. Tahbaz-Salehi and A. Jadbabaie, "Small world phenomenon, rapidly mixing Markov chains, and average consensus algorithms," in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 276–281.

[4] R. Olfati-Saber, "Ultrafast consensus in small-world networks," in *Proc. American Control Conference*, Portland, OR, 2005, pp. 2371–2378.

[5] J. Wang and N. Elia, "Consensus over networks with dynamic channels," in *Proc. American Control Conference*, Seattle, WA, 2008, pp. 2637–2642.

[6] Y. Hatano and M. Mesbahi, "Agreement over random networks," *IEEE Transactions on Automatic Control*, vol. 50, no. 11, pp. 1867–1872, 2005.

[7] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[8] A. Tahbaz-Salehi and A. Jadbabaie, "Necessary and sufficient conditions for consensus over random independent and identically distributed switching graphs," in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 4209–4214.

[9] S. Di Cairano, A. Pasini, A. Bemporad, and R. M. Murray, "Convergence properties of dynamic agents consensus networks with broken links," in *Proc. American Control Conference*, Seattle, WA, 2008, pp. 1362–1367.

[10] R. Olfati-Saber and R. M. Murray, “Consensus protocols for networks of dynamic agents,” in *Proc. American Control Conference*, Denver, CO, 2003, pp. 951–956.

[11] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[12] R. Olfati-Saber and J. S. Shamma, “Consensus filters for sensor networks and distributed sensor fusion,” in *Proc. IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, 2005, pp. 6698–6703.

[13] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, “Dynamic consensus for mobile networks,” in *Proc. IFAC World Congress*, Prague, Czech Republic, 2005.

[14] D. Kempe, A. Dobra, and J. Gehrke, “Gossip-based computation of aggregate information,” in *Proc. IEEE Symposium on Foundations of Computer Science*, Cambridge, MA, 2003, pp. 482–491.

[15] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[16] D. S. Scherber and H. C. Papadopoulos, “Distributed computation of averages over ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 776–787, 2005.

[17] D. B. Kingston and R. W. Beard, “Discrete-time average-consensus under switching network topologies,” in *Proc. American Control Conference*, Minneapolis, MN, 2006, pp. 3551–3556.

[18] A. Olshevsky and J. N. Tsitsiklis, “Convergence rates in distributed consensus and averaging,” in *Proc. IEEE Conference on Decision and Control*, San Diego, CA, 2006, pp. 3387–3392.

[19] L. Xiao, S. Boyd, and S.-J. Kim, “Distributed average consensus with least-mean-square deviation,” *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[20] F. Fagnani and S. Zampieri, “Randomized consensus algorithms over large scale networks,” *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 634–649, 2008.

[21] M. Zhu and S. Martínez, "Dynamic average consensus on synchronous communication networks," in *Proc. American Control Conference*, Seattle, WA, 2008, pp. 4382–4387.

[22] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 33–55, 2009.

[23] B. N. Oreshkin, M. J. Coates, and M. G. Rabbat, "Optimization and analysis of distributed averaging with short node memory," *IEEE Transactions on Signal Processing*, vol. 58, no. 5, pp. 2850–2865, 2010.

[24] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[25] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[26] M. Cao, D. A. Spielman, and A. S. Morse, "A lower bound on convergence of a distributed network consensus algorithm," in *Proc. IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, 2005, pp. 2356–2361.

[27] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[28] S. Sundaram and C. N. Hadjicostis, "Finite-time distributed consensus in graphs with time-invariant topologies," in *Proc. American Control Conference*, New York, NY, 2007, pp. 711–716.

[29] A. Olshevsky and J. N. Tsitsiklis, "On the nonexistence of quadratic Lyapunov functions for consensus algorithms," *IEEE Transactions on Automatic Control*, vol. 53, no. 11, pp. 2642–2645, 2008.

[30] A. Tahbaz-Salehi and A. Jadbabaie, "Consensus over ergodic stationary graph processes," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 225–230, 2010.

[31] A. Nedić, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[32] J. N. Tsitsiklis, “Problems in decentralized decision making and computation,” Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, 1984.

[33] M. Jelasity and A. Montresor, “Epidemic-style proactive aggregation in large overlay networks,” in *Proc. IEEE International Conference on Distributed Computing Systems*, Tokyo, Japan, 2004, pp. 102–109.

[34] A. Montresor, M. Jelasity, and O. Babaoglu, “Robust aggregation protocols for large-scale overlay networks,” in *Proc. IEEE/IFIP International Conference on Dependable Systems and Networks*, Florence, Italy, 2004, pp. 19–28.

[35] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Randomized gossip algorithms,” *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[36] M. Cao, D. A. Spielman, and E. M. Yeh, “Accelerated gossip algorithms for distributed computation,” in *Proc. Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2006, pp. 952–959.

[37] J.-Y. Chen, G. Pandurangan, and D. Xu, “Robust computation of aggregates in wireless sensor networks: Distributed randomized algorithms and analysis,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, pp. 987–1000, 2006.

[38] C. C. Moallemi and B. Van Roy, “Consensus propagation,” *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4753–4766, 2006.

[39] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. M. Murray, “Asynchronous distributed averaging on communication networks,” *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 512–520, 2007.

[40] V. Borkar and P. P. Varaiya, “Asymptotic agreement in distributed estimation,” *IEEE Transactions on Automatic Control*, vol. 27, no. 3, pp. 650–655, 1982.

[41] J. N. Tsitsiklis and M. Athans, “Convergence and asymptotic agreement in distributed decision problems,” *IEEE Transactions on Automatic Control*, vol. 29, no. 1, pp. 42–50, 1984.

[42] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.

[43] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, 2005, pp. 2996–3000.

[44] M. Cao, A. S. Morse, and B. D. O. Anderson, "Coordination of an asynchronous multi-agent system via averaging," in *Proc. IFAC World Congress*, Prague, Czech Republic, 2005.

[45] L. Fang and P. J. Antsaklis, "Information consensus of asynchronous discrete-time multi-agent systems," in *Proc. American Control Conference*, Portland, OR, 2005, pp. 1883–1888.

[46] L. Fang, P. J. Antsaklis, and A. Tzimas, "Asynchronous consensus protocols: Preliminary results, simulations and open questions," in *Proc. IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, 2005, pp. 2194–2199.

[47] L. Fang and P. J. Antsaklis, "On communication requirements for multi-agent consensus seeking," in *Networked Embedded Sensing and Control*, ser. Lecture Notes in Control and Information Sciences, P. J. Antsaklis and P. Tabuada, Eds. Berlin, Germany: Springer-Verlag, 2006, vol. 331, pp. 53–67.

[48] C. Y. Tang and J. Lu, "Controlled hopwise averaging: Bandwidth/energy-efficient asynchronous distributed averaging for wireless networks," in *Proc. American Control Conference*, St. Louis, MO, 2009, pp. 1561–1568.

[49] J. Lu and C. Y. Tang, "Convergence rate of controlled hopwise averaging on various graphs," in *Proc. IEEE Conference on Decision and Control*, Orlando, FL, 2011, pp. 4290–4295.

[50] ——, "Controlled hopwise averaging and its convergence rate," *IEEE Transactions on Automatic Control*, vol. 57, no. 4, pp. 1005–1012, 2012.

[51] S. Martinez, J. Cortes, and F. Bullo, "Motion coordination with distributed information," *IEEE Control Systems Magazine*, vol. 27, no. 4, pp. 75–88, 2007.

[52] J. Fax and R. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1465–1476, 2004.

[53] W. Ren and R. Beard, "Distributed consensus in multi-vehicle cooperative control,," in *Proc. Springer-Verlag*, London, 2008.

[54] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[55] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, "Flocking in fixed and switching networks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 863–868, 2007.

[56] J. Cortés, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, no. 3, pp. 726–737, 2008.

[57] D. Mosk-Aoyama and D. Shah, "Computing separable functions via gossip," in *Proc. ACM Symposium on Principles of Distributed Computing*, Denver, CO, 2006, pp. 113–122.

[58] A. Tahbaz-Salehi and A. Jadbabaie, "A one-parameter family of distributed consensus algorithms with boundary: From shortest paths to mean hitting times," in *Proc. IEEE Conference on Decision and Control*, San Diego, CA, 2006, pp. 4664–4669.

[59] D. Bauso, L. Giarré, and R. Pesenti, "Non-linear protocols for optimal distributed consensus in networks of dynamic agents," *Systems & Control Letters*, vol. 55, no. 11, pp. 918–928, 2006.

[60] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proc. IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, 2005, pp. 8179–8184.

[61] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Approximate distributed Kalman filtering in sensor networks with quantifiable performance," in *Proc. International Symposium on Information Processing in Sensor Networks*, Los Angeles, CA, 2005, pp. 133–139.

[62] ——, “Distributed sensor fusion using dynamic consensus,” in *Proc. IFAC World Congress*, Prague, Czech Republic, 2005.

[63] R. Olfati-Saber, “Distributed Kalman filtering for sensor networks,” in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 5492–5498.

[64] S. Roy, A. Saberi, and K. Herlugson, “A control-theoretic perspective on the design of distributed agreement protocols,” in *Proc. American Control Conference*, Portland, OR, 2005, pp. 1672–1679.

[65] S. Roy, K. Herlugson, and A. Saberi, “A control-theoretic approach to distributed discrete-valued decision-making in networks of sensing agents,” *IEEE Transactions on Mobile Computing*, vol. 5, no. 8, pp. 945–957, 2006.

[66] S. Sundaram and C. N. Hadjicostis, “Distributed consensus and linear functional calculation in networks: An observability perspective,” in *Proc. International Conference on Information Processing in Sensor Networks*, Cambridge, MA, 2007, pp. 99–108.

[67] J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, “Distributed anonymous discrete function computation,” *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2276–2289, 2011.

[68] K. A. Lehmann and M. Kaufmann, “Decentralized algorithms for evaluating centrality in complex networks,” University of Tübingen, Tübingen, Germany, Technical Report, 2003.

[69] W. Wang and C. Y. Tang, “Distributed computation of node and edge betweenness on tree graphs,” in *Proc. IEEE Conference on Decision and Control*, Florence, Italy, 2013, pp. 43–48.

[70] ——, “Distributed computation of classic and exponential closeness on tree graphs,” in *Proc. American Control Conference*, Portland, OR, 2014, pp. 2090–2095.

[71] ——, “Distributed estimation of betweenness centrality,” in *Proc. Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2015, pp. 250–257.

[72] ——, “Distributed estimation of closeness centrality,” in *Proc. IEEE Conference on Decision and Control*, Osaka, Japan, 2015, pp. 4860–4865.

[73] C. Li and Z. Qu, "Distributed estimation of algebraic connectivity of directed networks," *Systems & Control Letters*, vol. 62, no. 6, pp. 517–524, 2013.

[74] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," in *Proc. ACM Symposium on Theory of Computing*, 2004, pp. 561–568.

[75] H. Ishii and R. Tempo, "Distributed randomized algorithms for the PageRank computation," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 1987–2002, 2010.

[76] A. D. Sarma, A. R. Molla, G. Pandurangan, and E. Upfal, "Fast distributed PageRank computation," in *Proc. International Conference on Distributed Computing and Networking*, 2013, pp. 11–26.

[77] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity for mobile sensor networks," *Automatica*, vol. 46, no. 2, pp. 390–396, 2010.

[78] R. Aragues, G. Shi, D. V. Dimarogonas, C. Sagues, and K. H. Johansson, "Distributed algebraic connectivity estimation for adaptive event-triggered consensus," in *Proc. American Control Conference*, Montreal, Canada, 2012, pp. 32–37.

[79] C. Li and Z. Qu, "Distributed estimation of algebraic connectivity of directed networks," *Systems & Control Letters*, vol. 62, no. 6, pp. 517–524, 2013.

[80] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. International Symposium on Information Processing in Sensor Networks*, Los Angeles, CA, 2005, pp. 63–70.

[81] ——, "A space-time diffusion scheme for peer-to-peer least-squares estimation," in *Proc. International Conference on Information Processing in Sensor Networks*, Nashville, TN, 2006, pp. 168–176.

[82] J. Lu and C. Y. Tang, "Distributed asynchronous algorithms for solving positive definite linear equations over networks—Part I: Agent networks," in *Proc. IFAC Workshop on Estimation and Control of Networked Systems*, Venice, Italy, 2009, pp. 252–257.

[83] ——, “Distributed asynchronous algorithms for solving positive definite linear equations over networks—Part II: Wireless networks,” in *Proc. IFAC Workshop on Estimation and Control of Networked Systems*, Venice, Italy, 2009, pp. 258–263.

[84] S. Mou and A. S. Morse, “A fixed-neighbor, distributed algorithm for solving a linear algebraic equation,” in *Proc. European Control Conference*, Zurich, Switzerland, 2013, pp. 2269–2273.

[85] J. Liu, S. Mou, and A. S. Morse, “An asynchronous distributed algorithm for solving a linear algebraic equation,” in *Proc. IEEE Conference on Decision and Control*, Florence, Italy, 2013, pp. 5409–5414.

[86] S. Mou, J. Liu, and A. S. Morse, “A distributed algorithm for solving a linear algebraic equation,” *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 2863–2878, 2015.

[87] J. Lu and C. Y. Tang, “A distributed algorithm for solving positive definite linear equations over networks with membership dynamics,” *IEEE Transactions on Control of Network Systems*, 2016, to appear.

[88] S. T. Cady, A. D. Domínguez-García, and C. N. Hadjicostis, “A distributed generation control architecture for islanded AC microgrids,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1717–1735, 2015.

[89] A. Cherukuri and J. Cortes, “Initialization-free distributed coordination for economic dispatch under varying loads and generator commitment,” *Automatica*, vol. 74, pp. 183–193, 2016.

[90] A. Cherukuri and J. Cortés, “Decentralized Nash equilibrium seeking by strategic generators for DC optimal power flow,” in *Proc. Conference on Information Sciences and Systems*, 2017, pp. 1–6.

[91] J. Cai, D. Kim, R. Jaramillo, J. E. Braun, and J. Hu, “A general multi-agent control approach for building energy system optimization,” *Energy and Buildings*, vol. 127, pp. 337–351, 2016.

[92] X. Hou, Y. Xiao, J. Cai, J. Hu, and J. E. Braun, “Distributed model predictive control via proximal Jacobian ADMM for building control applications,” in *Proc. American Control Conference*, 2017, pp. 37–43.

[93] M. G. Rabbat and R. D. Nowak, "Distributed optimization in sensor networks," in *Proc. International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, 2004, pp. 20–27.

[94] P. Wan and M. D. Lemmon, "Event-triggered distributed optimization in sensor networks," in *Proc. International Conference on Information Processing in Sensor Networks*, 2009, pp. 49–60.

[95] A. Nedić and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[96] A. Nedić, D. P. Bertsekas, and V. S. Borkar, "Distributed asynchronous incremental subgradient methods," in *Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications*, D. Butnariu, Y. Censor, and S. Reich, Eds. Amsterdam, Holland: Elsevier, 2001, pp. 381–407.

[97] A. Nedić and D. P. Bertsekas, "Convergence rate of incremental subgradient algorithms," in *Stochastic Optimization: Algorithms and Applications*, S. P. Uryasev and P. M. Pardalos, Eds. Norwell, MA: Kluwer Academic Publishers, 2001, pp. 223–264.

[98] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 798–808, 2005.

[99] S.-H. Son, M. Chiang, S. R. Kulkarni, and S. C. Schwartz, "The value of clustering in distributed estimation for sensor networks," in *Proc. International Conference on Wireless Networks, Communications and Mobile Computing*, Maui, HI, 2005, pp. 969–974.

[100] B. Johansson, M. Rabi, and M. Johansson, "A simple peer-to-peer algorithm for distributed optimization in sensor networks," in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 4705–4710.

[101] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Stochastic incremental gradient descent for estimation in sensor networks," in *Proc. Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, 2007, pp. 582–586.

[102] ——, "Incremental stochastic subgradient algorithms for convex optimization," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 691–717, 2009.

[103] A. Nedić and A. Ozdaglar, "On the rate of convergence of distributed subgradient methods for multi-agent optimization," in *Proc. IEEE Conference on Decision and Control*, New Orleans, LA, 2007, pp. 4711–4716.

[104] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 4185–4190.

[105] I. Lobel and A. Ozdaglar, "Convergence analysis of distributed subgradient methods over random networks," in *Proc. Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2008, pp. 353–360.

[106] A. Nedić, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "Distributed subgradient methods and quantization effects," in *Proc. IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 4177–4184.

[107] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[108] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Asynchronous gossip algorithms for stochastic optimization," in *Proc. IEEE Conference on Decision and Control*, Shanghai, China, 2009, pp. 3581–3586.

[109] ——, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, 2010.

[110] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2012.

[111] J. Lu and C. Y. Tang, "Zero-gradient-sum algorithms for distributed convex optimization: The continuous-time case," in *Proc. American Control Conference*, San Francisco, CA, 2011, pp. 5474–5479.

[112] J. Lu, C. Y. Tang, P. R. Regier, and T. D. Bow, “A gossip algorithm for convex consensus optimization over networks,” in *Proc. American Control Conference*, Baltimore, MD, 2010, pp. 301–308.

[113] ——, “Gossip algorithms for convex consensus optimization over networks,” *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2917–2923, 2011.

[114] S. Nikookhoy, J. Lu, and C. Y. Tang, “Distributed convex optimization with identical constraints,” in *Proc. IEEE Conference on Decision and Control*, Orlando, FL, 2011, pp. 2926–2931.

[115] T. T. Doan and C. Y. Tang, “Continuous-time constrained distributed convex optimization,” in *Proc. Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2012.

[116] W. Chen and W. Ren, “Event-triggered zero-gradient-sum distributed consensus optimization over directed networks,” *Automatica*, vol. 65, pp. 90–97, 2016.

[117] J. Wang and N. Elia, “Control approach to distributed optimization,” in *Proc. Allerton Conference on Communication, Control and Computation*, 2010, pp. 557–561.

[118] ——, “A control perspective for centralized and distributed convex optimization,” in *Proc. IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 3800–3805.

[119] J. C. Duchi, A. Agarwal, and M. J. Wainwright, “Dual averaging for distributed optimization: Convergence analysis and network scaling,” *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[120] B. Gharesifard and J. Cortés, “Distributed continuous-time convex optimization on weight-balanced digraphs,” *IEEE Transactions on Automatic Control*, vol. 59, no. 3, pp. 781–786, 2014.

[121] M. E. J. Newman, *Networks: An Introduction.* New York, NY: Oxford University Press, 2010.

[122] F. R. K. Chung, *Spectral Graph Theory.* American Mathematical Society, 1997.

[123] T. Sahai, A. Speranzon, and A. Banaszuk, "Hearing the clusters of a graph: A distributed algorithm," *Automatica*, vol. 48, no. 1, pp. 15–24, 2012.

[124] M. Franceschelli, A. Gasparri, A. Giua, and C. Seatzu, "Decentralized estimation of Laplacian eigenvalues in multi-agent systems," *Automatica*, vol. 49, no. 4, pp. 1031–1036, 2013.

[125] T.-M. D. Tran and A. Y. Kibangou, "Distributed estimation of graph Laplacian eigenvalues by the alternating direction of multipliers method," in *Proc. IFAC World Congress*, Cape Town, South Africa, 2014, pp. 5526–5531.

[126] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control.* Upper Saddle River, NJ: Prentice Hall, 1996.

[127] C.-T. Lin, "Structural controllability," *IEEE Transactions on Automatic Control*, vol. 19, no. 3, pp. 201–208, 1974.

[128] S. U. Pillai, T. Suel, and S. Cha, "The Perron-Frobenius theorem: Some of its applications," *IEEE Signal Processing Magazine*, vol. 22, no. 2, pp. 62–75, 2005.

[129] S. A. Grandhi, R. Vijayan, and D. J. Goodman, "Distributed power control in cellular radio systems," *IEEE Transactions on Communications*, vol. 42, no. 234, pp. 226–228, 1994.

[130] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[131] S. Kamvar, T. Haveliwala, and G. Golub, "Adaptive methods for the computation of PageRank," *Linear Algebra and its Applications*, vol. 386, pp. 51–65, 2004.

[132] M. Yang and C. Y. Tang, "Distributed estimation of graph spectrum," in *Proc. American Control Conference*, Chicago, IL, 2015, pp. 2703–2708.

[133] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[134] R. S. Varga, *Matrix Iterative Analysis*, 2nd ed. Berlin, Germany: Springer-Verlag, 2009.

[135] R. A. Horn and C. R. Johnson, *Matrix Analysis*. New York, NY: Cambridge University Press, 2012.