applied sciences

MDPI

Article

Concensus-Based ALADIN Method to Faster the Decentralized Estimation of Laplacian Spectrum

Thi-Minh-Dung Tran [1,*], Luu Ngoc An [1] and Ngoc Chi Nam Doan [2]

[1] The Faculty of Electrical Engineering, The University of Danang—University of Science and Technology, 54 Nguyen Luong Bang Street, Danang City 550000, Vietnam; lnan@dut.udn.vn

[2] Manufacturing Execution and Control Group, Singapore Institute of Manufacturing Technology, Singapore 138634, Singapore; doanncn@SIMTech.a-star.edu.sg

* Correspondence: ttmdung@dut.udn.vn; Tel.: +84-236-735-112

Received: 13 June 2020; Accepted: 13 July 2020; Published: 13 August 2020

Abstract: With the upcoming fifth Industrial Revolution, humans and collaborative robots will dance together in production. They themselves act as an agent in a connected world, understood as a multi-agent system, in which the Laplacian spectrum plays an important role since it can define the connection of the complex networks as well as depict the robustness. In addition, the Laplacian spectrum can locally check the controllability and observability of a dynamic controlled network, etc. This paper presents a new method, which is based on the Augmented Lagrange based Alternating Direction Inexact Newton (ALADIN) method, to faster the convergence rate of the Laplacian Spectrum Estimation via factorization of the average consensus matrices, that are expressed as Laplacian-based matrices problems. Herein, the non-zero distinct Laplacian eigenvalues are the inverse of the stepsizes $\{\alpha_t, t = 1, 2, \ldots\}$ of those matrices. Therefore, the problem now is to carry out the agreement on the stepsize values for all agents in the given network while ensuring the factorization of average consensus matrices to be accomplished. Furthermore, in order to obtain the entire Laplacian spectrum, it is necessary to estimate the relevant multiplicities of these distinct eigenvalues. Consequently, a non-convex optimization problem is formed and solved using ALADIN method. The effectiveness of the proposed method is evaluated through the simulation results and the comparison with the Lagrange-based method in advance.

Keywords: multi-agent systems; laplacian eigenvalues; augmented Lagrange based Alternating Direction Inexact Newton (ALADIN) method; consensus algorithms; Alternating Direction of Multipliers Method (ADMM)

1. Introduction

Leaders around the world obviously prefer the present era of connectivity as the Fourth Industrial Revolution [2]. Industry 4.0 has been significantly contributing to the transformation of many industries such as transportation, manufacturing, health-care, agriculture, etc. via enabling data transmission and integration between disciplines. Today, we are in the fourth one, a generation of connection between our physical, digital, social, and biological worlds. In particular, data and information from these different areas have been made available and have been connected in complex and dense networks. For better integration and utilization, control aspect of these complex networks, researchers from different communities have brought different contributions varying from topology inference to control strategy, deputizing for interacting systems, which are modeled by graphs, whose vertices represent the components of the system while edges stand for the interactions between these components.

In the last decade there has been dramatic increasing number of publications in the cooperative control of multi-agent systems. In control of multi-agent systems, the performance of the whole system

depends on both structure and the connections between individuals of the systems. Here, the total connections can be defined by the graph Laplacian matrix, and its spectrum is involved in some useful properties of the network system [3,4]. For instance, the second smallest graph Laplacian eigenvalue, i.e., the so-called algebraic connectivity of the graph, has the main role in the convergence time of various distributed algorithms as well as the performance and robustness of dynamical systems [5]. Conceptually, agents share their information with each other to achieve common objectives, relative position information, or common control algorithms. This is called consensus problem [6,7], where a group of agents' approaches average consensus in an undirected network under simple linear iteration scheme.

It is well known that in a multi-agent system, consensus is achieved if and only if the network is connected (the algebraic connectivity) being strictly greater than zero [8]. On the other hand, the largest Laplacian eigenvalue is an important factor to decide the stability of the system. For example, minimizing the spectral radius in [9] leads to maximize the robustness of the network to time delays under a linear consensus protocol. Furthermore, to speed up consensus algorithms, the optimal Laplacian-based consensus matrix is obtained with a stepsize which is the inverse of the sum of the smallest and the largest non-zero graph Laplacian eigenvalue [10]. Moreover, the authors in [11,12] have proved that the spectrum of the Laplacian matrix can be used to design consensus matrices to obtain average consensus in finite number of steps.

In order to investigate network efficiency, structural robustness of a network which is related to its performance despite changes in the network topology [13] has been also studied. The concept of natural connectivity as a spectral measure of robustness was introduced in [14]. It is expressed in mathematical form as the average eigenvalue of the adjacency matrix of the graph representing the network topology. The Laplacian spectrum $sp(\mathbf{L}) = \{\lambda_1^{m_1}, \ldots, \lambda_i, \ldots\}$ can also be employed to compute the robustness indices, for instance, the number of spanning trees and the effective graph resistance (Kirchhoff index) [15]. The smaller (or greater) the Kirchhoff index (or the number of spanning trees) is, the more robust the network becomes. In addition, it has been pointed out that adding an edge strictly decreases the Kirchhoff index and hence increases the robustness. In [16], the authors have proposed a method to monitor collaboratively the robustness of the networks partitioned into sub-networks by Kirchhoff index $\mathcal{R}_{\mathbf{L}} = N \sum_{i=2}^{D+1} \frac{m_i}{\lambda_i}$. Here, an Alternating Direction of Multipliers Method (ADMM)-based algorithm was employed to perform the factorization of the averaging matrix and to compute the average degree of the network concurrently. However, the main point in this work was the reformulation into the convex optimization problem, which is convenient to make use of the ADMM method to solve the problem. In addition to that, the impact of the Laplacian spectrum into power systems is expressed through energy management in smart grids [17] and the determination of the grid robustness against low frequency disturbance in [18]. In this work, in the framework of spectral graph theory, the authors reveal that the decomposition of frequency signal along scaled Laplacian spectrum when the damping-inertia ratios are uniform across buses not only makes the system respond faster but also helps lower the system nadir after a disturbance. In dynamic network systems, the spectrum of Laplacian matrix can also be utilized for locally checking the controllability and the observability [19].

From a short literature survey above, it is obvious that the Laplacian spectrum plays an important role in many fields. For instance, Laplacian spectrum can be used to design consensus matrices [11,12], to compute these robustness indices [15–18], or to check the controllability and the observability [19]. Hence, it is desirable to have an efficient method for monitoring the Laplacian spectrum of a dynamics network system.

One thing to remark here is that if the global network topology is known in a-priori, the Laplacian matrix can be easily deduced. However, implementing a centralized structure is an expensive task due to the high computational cost, the heavy communication infrastructure aspect and the problem from large dimensionality. Additionally, if there is a failure problem from one point, it will affect the whole network. Therefore, our study is restricted to the assumption that the network topology

(represented by the Laplacian Matrix) is unknown at the first glance. A dominant contribution of this paper is the possibility of implementing this monitoring scheme in a decentralized manner.

In this paper, we present an Augmented Lagrangian based Alternating Direction Inexact Newton (ALADIN) method to estimate the Laplacian spectrum in decentralized scheme for dynamic controlled networks. The key feature of this paper is the direct solution to non-convex optimization for Laplacian spectrum estimation using ALADIN method. To simplify, the scope of this study is restricted to networks performing noise-free as well as the number of the agents N in the network in known in-priori by using random walk algorithm [20]. The network is modeled, then Laplacian eigenvalues and average consensus are retrieved respectively. Since the Laplacian spectrum matrix is not directly computable for undetermined network topology, the decentralized estimation of the Laplacian spectrum has been introduced with three main approaches in the recent literature: Fast Fourier Transform (FFT)-based methods [21,22], local eigenvalue decomposition of given observability-based matrices [23], and distributed factorization of the averaging matrix $\mathbf{J}_N = \frac{1}{N}\mathbf{1}\mathbf{1}^T$ [24]. FFT-based methods require a specific protocol. However, they do not make use of the available measurements coming from the consensus protocol. On the other hand, the method in [23] allows using the transient of the average consensus protocol but for several consecutive initial conditions. The distributed factorization of the averaging matrix in [24] yields the inverses of non-zero Laplacian eigenvalues and can be solved as a constrained consensus problem. The Laplacian eigenvalues can be deduced as the inverse of the stepsizes in each estimating factor, where these factors are constrained to be structured as Laplacian based consensus matrices. In [1], authors have applied a gradient descent algorithm to solve this optimization problem in which only local minima was guarantees accompany with slow convergence rate. In order to solve this annoying issue, in [16,24], the authors have introduced an interesting way by reformulating a non-convex optimization problem in [1] into convex one and solved by applying an ADMM-based method. However, this is an indirect approach obtaining by an adequate re-parameterization. In this paper, we inherit the idea in [1] to form the non-convex optimization for decentralized estimation of Laplacian spectrum and then directly solve it using the ALADIN method that was proposed by [25]. The proposed approach is then evaluated with two network structures for performance evaluation in comparison with gradient descent method [1].

In this paper, we firstly introduce the background of average consensus and state the problem in Section 2, then present the distributed estimation of Laplacian spectrum in Section 3. The structure of this section can be illustrated as in Figure 1. Before concluding the paper, the simulation results are described in Section 4 to evaluate the efficiency of the proposed method.

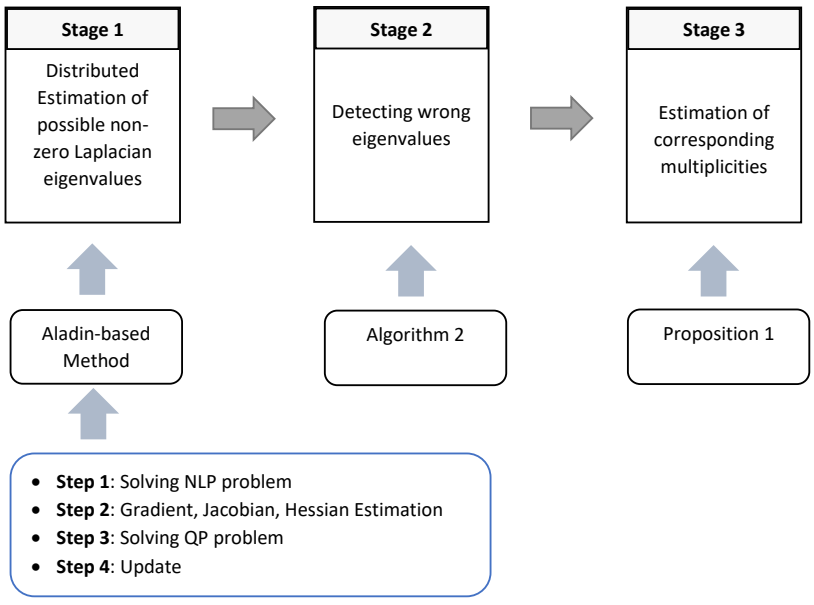

Figure 1. Structure of Section 3.

2. Background and Problem Statement

Consider a dynamic network, in which interconnection is represented by $G(V, E)$, an undirected graph with components' set V and links' set E, consisting of $N = |V|$ nodes, let us denote by $\mathcal{N}_i = \{j \in V : (i, j) \in E\}$ the set of neighbors of node i and $d_i = |\mathcal{N}_i|$ its degree. Interactions between nodes can be captured by the Laplacian matrix $\mathbf{L} \in \Re^N$ with entries $l_{ii} = d_i$, $l_{ij} = -1$ if $j \in \mathcal{N}_i$ and $l_{ij} = 0$ elsewhere. Denote the Laplacian spectrum by $sp(\mathbf{L}) = \{\lambda_1^{m_1}, \lambda_2^{m_2}, \ldots, \lambda_{D+1}^{m_{D+1}}\}$, where the different Laplacian eigenvalues are in increasing order $0 = \lambda_1 < \lambda_2 < \ldots < \lambda_{D+1}$ and superscripts stand for multiplicities $m_i = m(\lambda_i)$, while $\mathbf{S}_2 = \{\lambda_2, \ldots, \lambda_{D+1}\}$ stands for the set of the non-zero distinct Laplacian eigenvalues.

2.1. Average Consensus

For each node $i \in V$, let $x_i(t)$ denotes the value of node i at timestep t. Define $\mathbf{x}(t) = [x_1(t), x_2(t), ..., x_N(t)]^T$, where N is the number of nodes in the network. Average consensus algorithms can be achieved by using the following linear iteration scheme as follows:

$$\mathbf{x}(t) = (\mathbf{I}_N - \alpha \mathbf{L})\mathbf{x}(t-1), \tag{1}$$

where α is an appropriately selecting stepsize [26], by which all nodes converge asymptotically to the same value $\bar{x}$ that is the average of the initial ones $\bar{x}\mathbf{1} = \lim_{t\to\infty} \mathbf{x}(t) = \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{x}(0)$.

On the other hand, it has been shown in [11,12] that the average consensus matrix can be factored as

$$\prod_{t=D}^{1} \mathbf{W}_t = \frac{1}{N}\mathbf{1}\mathbf{1}^T, \tag{2}$$

where $\mathbf{W}_t = \vartheta_t \mathbf{I}_N + \alpha_t \mathbf{L}$, ϑ_t and α_t being parameters to be designed. In [12], the solution was given by $\vartheta_t = 1$ and $\alpha_t = -\frac{1}{\lambda_{t+1}}$, λ_t being a non-zero Laplacian eigenvalue. Owing to the above factorization, average consensus can then be reached in D steps, D being the number of distinct non-zero Laplacian eigenvalues:

$$\bar{\mathbf{x}} = \mathbf{x}(D) = \prod_{t=D}^{1} \mathbf{W}_t \mathbf{x}(0) = \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{x}(0) \text{ for all } \mathbf{x}(0) \in \Re^N. \tag{3}$$

2.2. Problem Statement

It can be noted that by factorizing the average consensus matrix, while constraining the factor matrices to be in the form $\mathbf{I}_N - \alpha_t \mathbf{L}$, the eigenvalues of the Laplacian matrix as the inverse of α_t can be deduced. The uniqueness has been proved in [1].

Lemma 1. *[1] Let $\lambda_2, \cdots, \lambda_{D+1} \neq 0$ be the D distinct non-zero eigenvalues of the graph Laplacian matrix $\boldsymbol{L}$, then, up to permutation, the sequence $\{\alpha_i\}_{i=1,\cdots,D}$, with $\alpha_i = \frac{1}{\lambda_{i+1}}$, $i = 1, 2, \cdots, D$, is the unique sequence allows getting the minimal factorization of the average consensus matrix as $\frac{1}{N}\mathbf{1}\mathbf{1}^T = \prod_{i=1}^{D}(\boldsymbol{I}_N - \alpha_i \boldsymbol{L})$.*

Therefore, in order to implement the proposed method, the knowledge of the network should be known. Meaning that, the number of the components N of the given network should be known by adding a learning mechanism as a configuration step. Practically, in most systems where communications are involved, learning sequences are used for communication channel identification or for synchronization. In [20], the authors have proposed a method using random walks to estimate the global properties of large connected undirected graphs such as number of vertices, edges, etc. However, it is not in the scope of this paper. Indeed, assuming that the number of agents N is known in a-priori, a consensus protocol in [10] is to be uploaded to each agent to compute the average consensus value $\bar{x}$. The main task in our study is to estimate the whole Laplacian spectrum.

3. Distributed Estimation of Laplacian Spectrum

Given an initial input-output pair $\{\mathbf{x}(0), \bar{\mathbf{x}}\}$, with $\bar{\mathbf{x}} = \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{x}(0)$, the matrix factorization problem (3) is equivalent to minimize the cost function $E(\mathbf{W}) = \|\mathbf{x}(D) - \bar{\mathbf{x}}\|^2$ that can also be rewritten as follows:

$$E(\mathbf{W}) = \left\| \prod_{t=D}^{1} \mathbf{W}_t \mathbf{x}(0) - \bar{\mathbf{x}} \right\|^2, \tag{4}$$

where D is the number of steps before reaching average consensus and $\mathbf{W}_t = \mathbf{I}_N - \boldsymbol{\alpha}_t \mathbf{L}$.

Note that there is no need for a central node to set the initial input-output pair. Indeed, such a pair can be obtained after running a standard average consensus algorithm. Each node keeps in memory its own initial value and the consensus value.

Solving this factorization problem consists in finding the sequence of stepsize $\{\boldsymbol{\alpha}_t\}_{t=1,\cdots,D}$. It is obvious that $\boldsymbol{\alpha}_t$ are global parameters. To relax these constraints, define the factor matrices as $\mathbf{W}_t = \mathbf{I}_N - \boldsymbol{\Lambda}_t \mathbf{L}$, where $\boldsymbol{\Lambda}_t = diag(\boldsymbol{\alpha}_t)$, $\boldsymbol{\alpha}_t = [\alpha_{t,1}, \alpha_{t,2}, \ldots, \alpha_{t,N}]$, $t = 1, 2, \ldots, D$. The problem above can be reformulated as a constrained consensus problem, that is to compute the sequence of stepsize $\{\boldsymbol{\alpha}_t\}$ so that $\alpha_{t,1} = \alpha_{t,2} = \ldots = \alpha_{t,N}$. Moreover, in Section 2.1, D is denoted as number of non-zero distinct Laplacian eigenvalues. However, in this work, Laplacian matrix is assumed to be not-known in-a-priori. Therefore, the authors have assigned D as $h = N - 1$ since N can be estimated in the configuration step through the Random Walk Algorithm proposed in [20]. Furthermore, the Laplacian Spectrum estimation procedure is divided in following stages:

- *Stage 1*: Distributed estimation of the set of non-zero Laplacian eigenvalues $\mathcal{S}_1 = \{\lambda_1, \lambda_2, \ldots, \lambda_h\}$, composing of the set of D non-zero distinct Laplacian eigenvalues $\mathcal{S}_2 = \{\lambda_1, \lambda_2, \ldots, \lambda_D\}$.
- *Stage 2*: Eliminating the wrong eigenvalues in the set $\mathcal{S}_1$ to obtain the set $\mathcal{S}_2$
- *Stage 3*: Estimating the multiplicities $\mathbf{m}$ corresponding to each eigenvalues in the set $\mathcal{S}_2$ to achieve the whole Laplacian spectrum $sp(\mathbf{L})$.

3.1. Distributed Estimation of Non-Zero Laplacian Eigenvalues

For distributively carrying out the factorization of the average consensus matrix as factors of Laplacian based consensus matrices, the idea is to minimize the disagreement between neighbors on the value of $\boldsymbol{\alpha}_t$ while ensuring that the factorization of the average consensus matrix is achieved. Such a factorization is assessed by constraining the values of the nodes after h iterations of the consensus algorithm to be equal to the average of the initial values:

$$\begin{aligned} \min_{\boldsymbol{\alpha}_t \in R^{N\times 1}, t=1,2,\ldots,h} \quad & \frac{1}{2} \sum_{t=1}^{h} \sum_{i\in V} \sum_{j\in\mathcal{N}_i} (\alpha_{t,j} - \alpha_{t,i})^2 \\ \text{subject to} \quad & \mathbf{x}(h) = \bar{\mathbf{x}} \end{aligned} \tag{5}$$

or, it can be rewritten in the following form:

$$\begin{aligned} \min_{\boldsymbol{\alpha}_t \in R^{N\times 1}} \quad & \frac{1}{2} \sum_{t=1}^{h} \boldsymbol{\alpha}_t^T \mathbf{L} \boldsymbol{\alpha}_t. \\ \text{subject to} \quad & \mathbf{x}(h) = \bar{\mathbf{x}} \end{aligned} \tag{6}$$

This optimization has been solved by applying Augmented Lagrange Method [1]. However, the disadvantage of this method is the slow convergence rate due to the fact that it is a non-convex optimization problem. To overcome this unexpected issue, the authors have suggested an interesting variant by converting the non-convex function into the convex one. By that, the optimization can be easily and effectively solved by Alternating Direction of Multipliers Methods

(ADMM) [16,27]. In this paper, we proposed a method that can solve a non-convex problem effectively by employing ALADIN method, which is described as following:

(1) **Step 1:** ALADIN solves in parallel a sequence of equality-constrained non-linear problems (NLP) by introducing an augmented variables $\mathbf{y}$ as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha}_t \in \mathbf{R}^{N\times 1}, t=1,2,\ldots,h} & \frac{1}{2}\boldsymbol{\alpha}_t^T \mathbf{L}\boldsymbol{\alpha}_t + \boldsymbol{\lambda}^T(\boldsymbol{\alpha}_t - \mathbf{y}_t) + \frac{\rho}{2}\|(\boldsymbol{\alpha}_t - \mathbf{y}_t)\|^2_{\boldsymbol{\Sigma}_t} \\ \text{subject to } & \mathbf{x}(h) - \bar{\mathbf{x}} = 0 | \boldsymbol{\beta} \\ & \mathbf{y}_{t,i} = \mathbf{y}_{t,j} \;\; i = 1,\ldots,N; j \in \mathcal{N}_i \quad (\mathcal{C}) \end{aligned} \tag{7}$$

where ρ, β are penalty parameter and multiplier of the equality constraint, respectively. $\mathcal{C} = \{\mathbf{y}_t : y_{t,i} = y_{t,j}, i = 1,\ldots,N; j \in \mathcal{N}_i\}$.

One thing to note here is that with the given initial information $x_i(0), i = 1,2,\ldots,N$, running a standard consensus algorithm can determine the average value $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$. In addition to that, the positive semi-definite scaling matrices Σ_t can be randomly initialized or even be an identity matrix $\mathbf{I}_N$. In this optimization problem (7), augmented variables $\mathbf{y}_t$ are introduced with respect to the constraint $\mathcal{C}$. However, this NLP is to be solved to define the variables $\boldsymbol{\alpha}_t$, hence, the constraint $\mathcal{C}$ is going to be relaxed.

The solution $\boldsymbol{\alpha}_t[k+1], \boldsymbol{\beta}[k+1]$ obtained from (7) is then used to check the stopping criteria the the next steps of the ALADIN-based algorithm procedure with k being an iteration of the optimization process. Herein, if $\frac{1}{2}\sum_{t=1}^{h}\sum_{i\in V}\sum_{j\in\mathcal{N}_i}(\alpha_{t,j} - \alpha_{t,i})^2 < \epsilon$ and $\rho\|\Sigma_t(\boldsymbol{\alpha}_t - \mathbf{y}_t)\|_1 \leq \epsilon$, then one can get $\boldsymbol{\alpha}_t^*$ as well as the Algorithm stops.

(2) **Step 2:** The Gradients, Jacobian matrices, and Hessian matrices are estimated for the next quadratic programming (QP) subproblems the as follows:

$$\mathbf{g}_t = \frac{\partial}{\partial \boldsymbol{\alpha}_t}\{\frac{1}{2}\boldsymbol{\alpha}_t^T\mathbf{L}\boldsymbol{\alpha}_t + \boldsymbol{\beta}_t^T(\mathbf{x}(h) - \bar{\mathbf{x}})\} \tag{8}$$

$$\mathbf{C}_t = \frac{\partial}{\partial \boldsymbol{\alpha}_t}(\mathbf{x}(h) - \bar{\mathbf{x}})^T \tag{9}$$

$$\mathbf{B}_t = \frac{\partial^2}{\partial \boldsymbol{\alpha}_t^2}(\frac{1}{2}\boldsymbol{\alpha}_t^T\mathbf{L}\boldsymbol{\alpha}_t). \tag{10}$$

(3) **Step 3:** Analogically to inexact SQP method, the QP problem is solved to find the $\Delta\boldsymbol{\alpha}_t[k]$ and the affine multiplier $\lambda_{QP}[k]$ as follows:

$$\begin{aligned} \min_{\Delta\boldsymbol{\alpha}\in\mathbf{R}^{N\times h}, s\in\mathbf{R}^{N\times 1}, t=1,2,\ldots,h} & \sum_{t=1}^{h}\{\frac{1}{2}\Delta\boldsymbol{\alpha}_t^T\mathbf{B}_t\Delta\boldsymbol{\alpha}_t + \mathbf{g}_t^T\Delta\boldsymbol{\alpha}_t\} + \boldsymbol{\lambda}^T\mathbf{s} + \frac{\mu}{2}\|\mathbf{s}\|^2 \\ \text{subject to } & \sum_{t=1}^{h}(\boldsymbol{\alpha}_t + \Delta\boldsymbol{\alpha}_t - \mathbf{y}_t) = \mathbf{s} | \boldsymbol{\lambda}_{QP} \\ & \mathbf{C}_t\Delta\boldsymbol{\alpha}_t = 0, t = 1,2,\ldots,h | \eta \end{aligned} \tag{11}$$

where $\mathbf{s}$ is the slack variable, introduced into the QP sub-problem to attenuate the numerical reasons when the penalty parameter μ becomes large.

(4) **Step 4:** The final step is to update $\boldsymbol{\lambda}[k+1], \mathbf{y}_t[k+1], \mathbf{B}_t[k+1]$:

$$\boldsymbol{\lambda}[k+1] = \boldsymbol{\lambda}_{QP}[k] \tag{12}$$

$$\hat{\mathbf{y}}_t[k] = \boldsymbol{\alpha}_t[k] + \Delta\boldsymbol{\alpha}_t[k] \tag{13}$$

This update rule is relevant to the full-size step where $a_1 = a_2 = a_3 = 1$ for the steplength computation, which proposed in [25]. Then, projecting $\hat{\mathbf{y}}[k]$ on the constraint $\mathcal{C}$ to derive $\mathbf{y}[k+1]$.

The steps are repeated until the stopping criteria is satisfied.

In order to derive the distributed algorithm, let us take a closer look at each step of the proposed ALADIN-based method.

3.1.1. Implementation of Decoupled Nonlinear Problems

Firstly, in order to solve the decoupled NLP (7) for $t = 1, \ldots, h$, the Augmented Lagrange method is applied here. Hence, we introduce the Augmented Lagrange function with the Lagrange multiplier $\boldsymbol{\beta}$ and penalty parameter $\mathbf{c}$ as below:

$$\mathbf{H}_{1,t} = \frac{1}{2}\boldsymbol{\alpha}_t^T \mathbf{L}\boldsymbol{\alpha}_t + \boldsymbol{\lambda}^T(\boldsymbol{\alpha}_t - \mathbf{y}_t) + \frac{\rho}{2}\|(\boldsymbol{\alpha}_t - \mathbf{y}_t)\|_{\boldsymbol{\Sigma}_t}^2 + \beta^T(\mathbf{x}(h) - \bar{\mathbf{x}}) + \frac{c}{2}\|\mathbf{x}(h) - \bar{\mathbf{x}}\|_2^2 \tag{14}$$

The solution of this problem can be obtained by applying a gradient descent method iteratively:

$$\boldsymbol{\alpha}_t[k+1] = \boldsymbol{\alpha}_t[k] - b\frac{\partial \mathbf{H}_{1,t}}{\partial \boldsymbol{\alpha}_t} \tag{15}$$

$$\boldsymbol{\beta}[k+1] = \boldsymbol{\beta}[k] + c(\mathbf{x}(h) - \bar{\mathbf{x}}) \tag{16}$$

where b stands for stepsizes of the gradient descent method, which can be chosen by fix constants or be determined by deploying a line-search algorithms such as Wolfe Condition, Back-stracking, or Armijo ones in [28], while c dedicates for the penalty parameter of the Augmented Lagrange method.

Lemma 2. *The derivatives of this Lagrange function* (14) *is obtained as follows:*

$$\frac{\partial \boldsymbol{H}_{1,t}}{\partial \boldsymbol{\alpha}_t} = \boldsymbol{L}\boldsymbol{\alpha}_t + \boldsymbol{\lambda} + \rho\boldsymbol{\Sigma}_t(\boldsymbol{\alpha}_t - \boldsymbol{y}_t) - diag^{-1}(\boldsymbol{\alpha}_t)diag(\boldsymbol{x}_{t-1} - \boldsymbol{x}_t)\boldsymbol{\delta}_t - diag^{-1}(\boldsymbol{\alpha}_t)diag(\boldsymbol{x}_{t-1} - \boldsymbol{x}_t)\boldsymbol{e}_t \tag{17}$$

where $\boldsymbol{\delta}_h = \beta$, *and* $\boldsymbol{\delta}_t = \boldsymbol{W}_{t+1}\boldsymbol{\delta}_{t+1}$, *while* $\boldsymbol{e}_h = \mathbf{x}(h) - \bar{\mathbf{x}}$ *and* $\boldsymbol{e}_t = \boldsymbol{W}_{t+1}\boldsymbol{e}_{t+1}$.

The proof is showed in the Appendix A.

3.1.2. Implementation of the Coupling Quadratic Programming (QP)

Now, we apply the Karush–Kuhn–Tucker (KKT) conditions for solving the quadratic programming (QP) (11). The Augmented Lagrange Function is described as follows:

$$\begin{aligned}\mathbf{H}_2 = &\sum_{t=1}^{h}\left\{\frac{1}{2}\Delta\boldsymbol{\alpha}_t^T\mathbf{B}_t\Delta\boldsymbol{\alpha}_t + \mathbf{g}_t^T\Delta\boldsymbol{\alpha}_t + \boldsymbol{\lambda}_{QP}^T\Delta\boldsymbol{\alpha}_t\right\} + (\boldsymbol{\lambda}^T - \boldsymbol{\lambda}_{QP}^T)\mathbf{s} + \frac{\mu}{2}\|\mathbf{s}\|^2 \\ &+ \boldsymbol{\lambda}_{QP}^T\sum_{t=1}^{h}(\boldsymbol{\alpha}_t - \mathbf{y}_t) + \sum_{t=1}^{h}\eta_t^T\mathbf{C}_t\Delta\boldsymbol{\alpha}_t\end{aligned} \tag{18}$$

The KKT conditions, which are showed in the Appendix B yields a system of equations as follows:

$$\begin{cases} \mathbf{B}_t\Delta\alpha_t - \boldsymbol{\lambda}_{QP} + \mathbf{C}_t^T\eta_t &= -g_t, t = 1, 2, \ldots, h \\ \sum_{t=1}^{h}\Delta\boldsymbol{\alpha}_t - \frac{1}{\mu}\boldsymbol{\lambda}_{QP} &= -\frac{1}{\mu}\boldsymbol{\lambda} - \sum_{t=1}^{h}(\boldsymbol{\alpha}_t - \mathbf{y}_t) \\ \mathbf{C}_t\Delta\boldsymbol{\alpha}_t &= 0 \end{cases}$$

Solutions of this system of equations are $\Delta\alpha_t^*, \boldsymbol{\lambda}_{QP}^*, \eta_t^*$ respectively in the equivalent matrix form as follows:

$$\begin{pmatrix} \mathbf{B}_1 & \mathbf{0} & \dots & \mathbf{0} & \mathbf{I} & \mathbf{C}_1^T & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_2 & \dots & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{C}_2^T & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{B}_h & \mathbf{I} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{C}_h^T \\ \mathbf{I} & \mathbf{I} & \dots & \mathbf{I} & -\frac{1}{\mu}\mathbf{I} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{C}_1 & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_2 & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{C}_h & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \end{pmatrix} \begin{pmatrix} \Delta\alpha_1^* \\ \Delta\alpha_2^* \\ \vdots \\ \Delta\alpha_h^* \\ \boldsymbol{\lambda}_{QP}^* \\ \eta_1^* \\ \eta_2^* \\ \vdots \\ \eta_h^* \end{pmatrix} = \begin{pmatrix} -\mathbf{g}_1 \\ -\mathbf{g}_2 \\ \vdots \\ -\mathbf{g}_h \\ -\frac{\boldsymbol{\lambda}}{\mu} - \sum_{t=1}^{h}(\boldsymbol{\alpha}_t - \mathbf{y}_t) \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix} \tag{19}$$

This linear system can be solved by any linear solver. One thing to note here is the adaptive parameter μ. We can start with a quite small μ and adapt it during the optimization progress to relax the coupling conditions.

3.1.3. Implementation of an ADMM-Based Algorithm

Now, the update steps (12) and (13) are executed. Since the achieved $\hat{y}_{t,i}$ have to agree with the constraint ($\mathcal{C}$), we solve the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{y}_i \in \Re^{h\times 1}} \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \\ &\text{subject to} \quad \mathbf{y}_j = \mathbf{y}_i \;\; i = 1,\dots,N; j \in \mathcal{N}_i \quad (\mathcal{C}) \end{aligned}$$

To solve this optimization problem, an Alternating Direction Method of Multipliers (ADMM) in [16,27] can be employed by introducing an augmented parameters $\mathbf{z}_{ij}$. Hence, the optimization can be rewritten as follows:

$$\begin{aligned} &\min_{\mathbf{y}_t} \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \\ &\text{subject to} \quad \mathbf{y}_i = \mathbf{z}_{ij} \;\; i = 1,\dots,N; j \in \mathcal{N}_i \\ &\qquad\qquad \mathbf{z}_{ji} = \mathbf{z}_{ij} \end{aligned} \tag{20}$$

The Augmented Lagrange Function is defined as follows:

$$\mathbf{H}_3(\mathbf{y},\mathbf{z},\boldsymbol{\tau}) = \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 + \sum_{j\in\mathcal{N}_i} \boldsymbol{\tau}_{ij}^T(\mathbf{y}_i - \mathbf{z}_{ij}) + \frac{\nu}{2}\sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{z}_{ij}\|.$$

The ADMM solution acts in three steps repetitively until the tolerance achieves:

- Compute $\mathbf{y}_i$:

$$\mathbf{y}_i[p+1] = (1+\nu d_i)^{-1}\{\hat{y}_i[k] + \nu \sum_{j\in\mathcal{N}_i} \mathbf{z}_{ij}[p] - \sum_{j\in\mathcal{N}_i} \boldsymbol{\tau}_{ij}[p]\}. \tag{21}$$

- Compute $\mathbf{z}_{ij}$:

$$\mathbf{z}_{ij}[p+1] = \frac{\mathbf{y}_i[p+1] + \mathbf{y}_j[p+1]}{2} + \frac{\boldsymbol{\tau}_{ij}[p] + \boldsymbol{\tau}_{ji}[p]}{2\nu}. \tag{22}$$

- Lagrange multiplier update:

$$\boldsymbol{\tau}_{ij}[p+1] = \boldsymbol{\tau}_{ij}[p] + \nu(\mathbf{y}_i[p+1] - \mathbf{z}_{ij}[p+1]) \tag{23}$$

Herein, p is a iteration of the ADMM optimization process, then this output of this process is $\mathbf{y}_i[k+1] = \mathbf{y}_i^*[p+1]$.

The distributed algorithm is illustrated in Algorithm 1.

The convergence analysis of the ALADIN method has been studied clearly for both non-convex and convex optimization problem in [25]. Lemma 3 in [25] has been proven that with the cost function $f(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{t=1}^{h}\sum_{i\in V}\sum_{j\in\mathcal{N}_i}(\alpha_{t,j} - \alpha_{t,i})^2$ being twice continuously differentiable and letting $(\boldsymbol{\alpha}_t^*, \boldsymbol{\lambda}^*)$ for $t = 1, \ldots, h$ of problem (5) be a regular KKT point. On the other hand, the Hessian $\mathbf{B}_t = \frac{\partial^2}{\partial \boldsymbol{\alpha}_t^2}(\frac{1}{2}\boldsymbol{\alpha}_t^T\mathbf{L}\boldsymbol{\alpha}_t) + \rho\Sigma_t \succ 0$ obviously, since $\Sigma_t \succeq 0$. There exists constants χ_1, χ_2 such that for every point $\boldsymbol{\alpha}_t, \lambda$ satisfying the condition convergence of the decoupled minimization problem (7) have unique locally minimizers $\{\mathbf{y}_t, t = 1, \ldots, h\}$ that satisfy $\|\mathbf{y}_t - \boldsymbol{\alpha}_t\| \leq \chi_1\|\boldsymbol{\alpha}_t - \boldsymbol{\alpha}_t^*\| + \chi_2\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|$.

Moreover, if $\frac{1}{\mu} < \mathbf{0}(\|\mathbf{y}_t - \boldsymbol{\alpha}_t\|)$ when solving the QP (11), with the $((\boldsymbol{\alpha}_t^*, \boldsymbol{\lambda}^*))$ is a regular KKT point, then $\chi_1\|\boldsymbol{\alpha}_t - \boldsymbol{\alpha}_t^*\| + \chi_2\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\| \leq \frac{(\chi_1+\chi_2)\omega}{2}(\chi_1\|\boldsymbol{\alpha}_t - \boldsymbol{\alpha}_t^*\| + \chi_2\|\boldsymbol{\lambda} - \boldsymbol{\lambda}^*\|)^2$. This is sufficient to prove local quadratic convergence of the algorithm as χ_1, χ_2 are strictly positive constants. As a result, it is effectively applied to our proposed method since it is obviously an equality constrained non-convex optimization problem.

One thing to remark here is that in our study, the penalty parameters ρ, μ can be updated using the following rules:

$$\rho[k+1] = \begin{cases} \iota_\rho\rho[k] & \text{if } \rho[k] < \rho_{max} \\ \rho[k] & \text{elsewhere} \end{cases}$$

$$\mu[k+1] = \begin{cases} \iota_\mu\mu[k] & \text{if } \mu[k] < \mu_{max} \\ \mu[k] & \text{elsewhere} \end{cases}$$

where $\iota_\rho, \iota_\mu > 1$ and $\rho_{max} = 50, \mu_{max} = 30$ obtained by experience to avoid the numerical problem. Moreover, we can use the blockwise and damped Broyden–Fletcher–Goldfarb–Shanno (BFGS) update, which ensures positive definiteness of the $\mathbf{B}_t[k]$ to preserve the convergence properties of ALADIN proposed in [25].

3.2. Retrieving the Non-Zero Laplacian Eigenvalues

As stated before, the set of eigenvalues deriving from the Algorithm 1, denoted $\mathcal{S}_1$, composing of the set of non-zero distinct Laplacian eigenvalues $\mathcal{S}_2$.

Let $\hat{\hat{x}}_i$ be the final consensus value reconstructed by $\hat{\hat{x}}_i = \hat{x}_i(h) = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ and the iteration scheme of the finite-time average consensus is implemented as in (1). Following the idea of the Proposition 3 in [27], we step-by-step assume to leave one element of the $\mathcal{S}_1$, then, the remaining elements of this set are used to reconstruct $\hat{x}_i(h)$. If $\hat{\hat{x}}_i = \hat{x}_i(h)$ is satisfied, then the left element is not the expected Laplacian eigenvalue. Hence, we can eliminate it out of the set $\mathcal{S}_1$. Otherwise, the left element is one of non-zero distinct eigenvalues. We restore it in the set $\mathcal{S}_1$ and marked as an element in the set $\mathcal{S}_2$. Now, the procedure is continued with another element to the end.

The distributed non-zero Laplacian eigenvalues are described in Algorithm 2.

Algorithm 1 ALADIN-based Laplacian eigenvalues estimation

1. Initialization:

   - Number of nodes N, tolerance ϵ, initial input-output pairs $\{x_i(0), \bar{x}_i, i = 1, 2, \ldots, N\}$, where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ is retrieved from a standard average consensus algorithm.
   - Each node $i, i = 1, \ldots, N$ initializes:
     (a) random stepsizes $\alpha_{t,i}(0)$ for $t = 1, \ldots, h = N-1$.
     (b) random Lagrange multipliers $\beta_{t,i}, \eta_{t,i}$, for $t = 1, \ldots, h$ and $\lambda_i, \lambda_{QP,i}$.
     (c) Semi-positive definite matrices $\Sigma_t \in \Re^N$, learning rates b, penalty parameters ρ, c, μ.
   - Set $k = 0$;

2. Repeat:

   - Set $k := k + 1$,
   - Solving decouple NLP problem (7) for $t = 1, \ldots, h = N-1$:
     – Propagate Lagrange multipliers $\beta_{t,i}[k]$ for $t = h, \ldots, 2$ and $i = 1, \ldots, N$:
       (a) Set $\delta_{h,i}[k] = \beta_{t,i}[k]$.
       (b) $\delta_{t-1,i}[k] = \delta_{t,i}[k] + \alpha_{t,i}[k]\sum_{j\in\mathcal{N}_i}(\delta_{t,j}[k] - \delta_{t,i}[k])$.
     – Finite-time average Consensus steps:
     $$x_{t,i}[k] = x_{t-1,i}[k] + \alpha_{t,i}[k]\sum_{j\in\mathcal{N}_i}(x_{t-1,j}[k] - x_{t-1,i}[k]).$$
     – Propagate the error $e_{t,i}[k]$ by setting $e_{h,i}[k] = \mathbf{x}_{h,i}[k] - \bar{x}_i[k]$:
     $$e_{t-1,i}[k] = e_{t,i}[k] + \alpha_{t,i}\sum_{j\in\mathcal{N}_i}(e_{t,j}[k] - e_{t,i}[k]).$$
     – Update $\alpha_{t,i}$ for $t = 1, \ldots, h$:
     $$\begin{aligned}\alpha_{t,i}[k+1] = \alpha_{t,i}[k+1] &- b\sum_{j\in\mathcal{N}_i}(\alpha_{t,j}[k] - \alpha_{t,i}[k]) \\ &- b\lambda_i - b\rho[k]\sum_{j=1}^{N}\Sigma_t(i,j)(\alpha_{t,j}[k] - y_{t,j}[k]) \\ &+ b\sum_{j\in\mathcal{N}_i}(x_{t-1,j}[k] - x_{t-1,i}[k])\delta_{t,i}[k] \\ &+ b\sum_{j\in\mathcal{N}_i}(x_{t-1,j}[k] - x_{t-1,i}[k])e_{t,i}[k]\end{aligned}$$
     – Update NLP Lagrange multipliers $\beta_{t,i}$ for $t = 1, \ldots, h$ by (16).
   - Stopping criteria: if $\|\sum_{t=1}^{h}(\boldsymbol{\alpha}_t - \mathbf{y}_t)\| \le \epsilon$ and $\|\sum_{t=1}^{h}\boldsymbol{\alpha}_t^T\mathbf{L}\boldsymbol{\alpha}_t\| \le \epsilon$ are simultaneously satisfied, then stop the optimization procedure.
   - Compute Gradient, Jacobian matrices, and Hessian matrices as in (8)–(10), respectively.
   - Solving the coupling QP problem (11) via solving the linear Equation (19) to define $\Delta\alpha^*_{t,i}[k], \lambda^*_{QP,i}[k]$.
   - Update $\lambda, \hat{y}_{t,i}$:
     (a) $\lambda_i[k+1] = \lambda^*_{QP,i}[k]$
     (b) $\hat{y}_{t,i}[k] = \alpha_{t,i}[k+1] + \Delta\alpha^*_{t,i}[k]$
     (c) Projecting $\hat{y}_{t,i}[k]$ onto the constraint $\mathcal{C}$ by solving an ADMM-based optimization subproblem (20) to derive $y_{t,i}[k+1]$

Algorithm 2 Non-zero eigenvalues Estimation

1. Input: set of stepsizes, obtaining form Algorithm 1 $\mathcal{S}_1$, the input-output pairs $\{\mathbf{x}_i(0), \bar{\mathbf{x}}_i\}$, $i = 1, \ldots, N$, the threshold ϵ.
2. Set $\mathcal{S}_2 = \emptyset$, $\mathcal{S} = \mathcal{S}_1$.
3. Repeat: **While** $\mathcal{S} \neq \emptyset$, pick an element α_t out of $\mathcal{S}$, hence $\mathcal{S} \setminus \{\alpha_t\}$.
   - Construct average consensus iteration scheme from the remain stepsizes to determine $\hat{x}_i(h), i = 1, \ldots, N$ as follows:

$$x_{t,i} = x_{t-1,i} + \alpha_{t,i} \sum_{j \in \mathcal{N}_i} (x_{t-1,j} - x_{t-1,i}), t = 1, \ldots, h$$

   - If $\bar{x}_i - \hat{\bar{x}}_i \leq \epsilon$, then $\mathcal{S} = \mathcal{S} \setminus \alpha_t$ and return to (3)
   - If $\bar{x}_i - \hat{\bar{x}}_i > \epsilon$, then α_t is included in $\mathcal{S}_2$ and $\mathcal{S} = \mathcal{S} \cup \alpha_t$ and return to (3)
4. Output: If $\mathcal{S}$ is empty, then the non-zero distinct Laplacian eigenvalues can be derived by taking the inverse of the set $\mathcal{S}_2$'s elements.

3.3. Multiplicities Estimation

Now, turning to the last stage, which is the corresponding Laplacian eigenvalues multiplicities estimation. In [16], the authors have proposed a linear integer programming optimization problem to figure out the multiplicities:

Proposition 1 ([16]). *Consider a connected undirected graph of N vertices with degree sequence $\{d_i\}$ and Laplacian matrix $\boldsymbol{L}$, having $D = |\mathcal{S}_2|$ distinct non-zero Laplacian eigenvalues $\mathcal{S}_2$. Let $\boldsymbol{m} \in \mathbb{Z}^{+D\times 1}$ be the vector of the corresponding multiplicities and be obtained by solving the integer programming below:*

$$\begin{aligned} \min_{\boldsymbol{m} \in \mathbb{Z}^{+D\times 1}} \quad & \mathcal{S}_2^T \boldsymbol{m} \\ \textit{subject to} \quad & \mathcal{S}_2^T \boldsymbol{m} = \textstyle\sum_{i=1}^{D} d_i \\ & \boldsymbol{1}^T \boldsymbol{m} = N - 1 \\ & \boldsymbol{m} \in \mathbb{Z}^{+D\times 1} \end{aligned} \tag{24}$$

The proof was given in [16]. Since all the multiplicities $\mathbf{m}$ are positive integers, then Brand-and-Bound method has been deployed to derive $\mathbf{m}$. Therefore, the problem (24) can be rewritten equivalently in linear integer programming form as follows:

$$\begin{aligned} \min_{\mathbf{m} \in \mathbb{Z}^{+D\times 1}} \quad & \mathcal{S}_2^T \mathbf{m} - \textstyle\sum_{i=1}^{D} d_i \\ \text{subject to} \quad & \mathbf{1}^T \mathbf{m} = N - 1 \\ & \mathbf{m} \in \mathbb{Z}^{+D\times 1} \end{aligned} \tag{25}$$

The Algorithm for this problem has been described clearly in [16]. At this step, the whole Laplacian spectrum $sp(\mathbf{L})$ has been obtained.

In fact, the estimation problem can be converted into a convex form and can be solved effectively by using ADMM-based method proposed in [16]. However, the purpose of this study is extremely appropriate for non-convex optimization problem through deploying the promising ALADIN-based method.

4. Simulation Results

In this section, the efficiency of the proposed ALADIN-based method to estimate the Laplacian spectrum is evaluated by considering the two following case studies.

Firstly, it can be said that the Laplacian spectrum can decide the performance of the network since it reveals the connection of the network. For example, the robustness of the network can be estimated before starting the operations to avoid the interruption during these operations and, as a result, enhance the benefits technically and economically.

On the purpose of monitoring the connection of a large network $G^*(V^*, E^*)$, one may face with the numerical issue in step 3 of the ALADIN-based method to solve the linear system (19) due to the huge dimension of the obtained matrix that leads to the common ill-conditioning problem with the inverse matrix calculation.

In [16], the authors have suggested to partition the large network into M disjoint sub-networks U_ℓ, $\ell = 1, 2, \cdots, M$, [29]. Let us define $\mathcal{N}_i^* = \{j \in V^* : (i,j) \in E^*\}$ and its cardinality $|\mathcal{N}_i^*|$ as the set of neighbors of node i and its degree in G^*, respectively. Each sub-network is monitored by a super-node i which knows the number N_ℓ of agents in the sub-network and the associated average information state x_ℓ. Node $i \in U_\ell$ is a super-node if it has at least one neighbor in a different subset, i.e., $\exists \ell^* \neq \ell$ s.t. $\mathcal{N}_i \bigcap U_{\ell^*} \neq \emptyset$. Let us consider that two sub-networks are connected if there exist edges linking at least two agents of these sub-networks. If two sub-networks are connected then their super-nodes are linked as showed in Figure 2. Here, the network can be social network, power system network, molecular network, etc.

Let $G = (V, E)$ be the undirected graph representing a network with $N = |V|$ super-nodes, which are black nodes in Figure 2. G captures the interaction between sub-networks of G^*. Therefore, the large network is to be robust if the partitions are strongly linked to each other and if the critical threshold is high enough in [16]. Then, the Laplacian spectrum of network of super-nodes G can monitor the connection of the large network G^* via the robustness index.

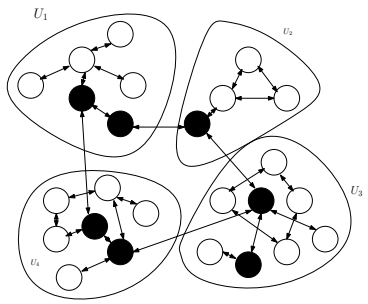

Figure 2. Network partitioned in 4 subsets. Super-nodes are depicted in black.

4.1. Case Study 1

Let us consider a large network partitioning into 4 disjoint sub-networks. Each sub-network has only one super-node. These super-nodes interact with each other by the graph $G = (V, E)$, depicted in Figure 3.

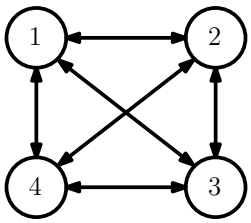

Figure 3. A network constituted by 4 nodes.

This network does have the Laplacian eigenvalues $sp(\mathbf{L}) = \{0, 4, 4, 4\}$. With the parameter of each super-node, denoted as $\mathbf{x}(0) = \{0.7417, 0.7699, 0.3216, 0.5466\}$, after deploying Algorithm 1, the set of $\boldsymbol{\alpha}_t, t = 1, \ldots, 3$ is obtained. The nodes trajectories are described as in Figure 4.

As can be seen, all $\boldsymbol{\alpha}_t$ execute the consensus problem at the beginning of the procedure, and then dig into satisfying the constraint.

Figure 5 illustrate the convergence of the cost function $\frac{1}{2}\sum_{t=1}^{N-1} \boldsymbol{\alpha}_t^T \mathbf{L} \boldsymbol{\alpha}_t$ in according to the constraint $\mathbf{x}(N-1) = \bar{x}\mathbf{1}$.

Furthermore, the Algorithm 2 is applied to eliminate the unexpected eigenvalues. As a result, we receive only one eigenvalue $\lambda = \frac{1}{0.25} = 4$. In order to accomplish the Laplacian spectrum estimation's procedure, we make use the Proposition 1 to pick out $m = 3$. Now, the entire Laplacian spectrum is achieved.

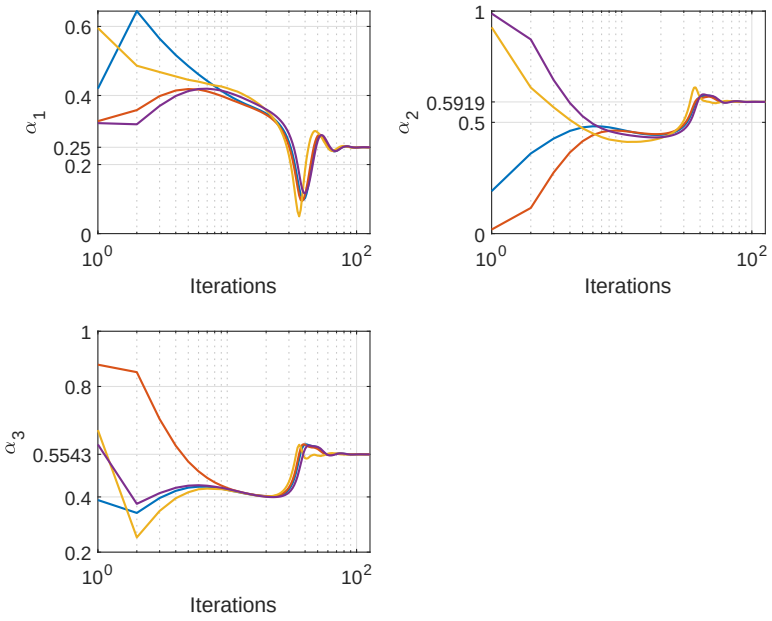

Figure 4. Trajectories convergence of $\boldsymbol{\alpha}_t$.

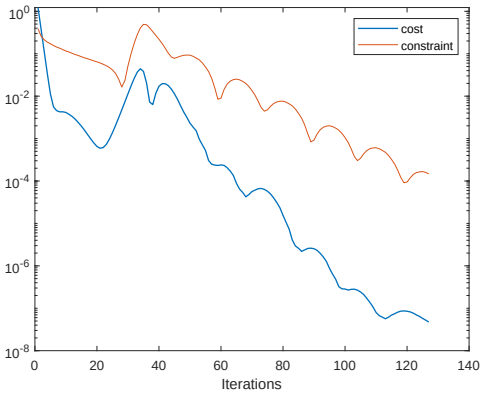

Figure 5. Convergence of the cost function according to its constraint.

At this step, the robustness index such as Kirchhoff index or the number of spanning trees can be calculated.

$$\mathcal{R}_{\mathbf{L}} = N \sum_{i=2}^{D+1} \frac{m_i}{\lambda_i} = 4\frac{3}{4} = 3.$$

Now, let us see how the proposed procedure works via the table below.

Table 1 is obtained after executing the proposed procedure.

As can be seen in Figure 4, at the iteration of around 90 the procedure can be stopped. In order to access the robustness of the whole large network, it is necessary to define the critical threshold, introduced in [16].

Next, we implement a comparison with the Lagrange method described in [1] by considering Case study 2.

Table 1. The achievement of the proposed procedure in the sense of 4-node topology.

$\boldsymbol{S}_1$	$\{0.25, 0.5919, 0.5543\}$
$\boldsymbol{S}_2$	$\{0.25\}$
$\mathbf{m}$	3
Iterations	130
$\mathcal{R}_{\mathbf{L}}$	3

4.2. Case Study 2

Let us consider a 6-node network described in Figure 6.

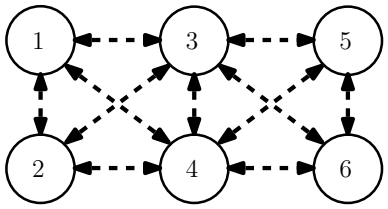

Figure 6. A new network constituted by 6 nodes.

It is known that for this topology, the Laplacian spectrum is $sp(\mathbf{L}) = \{0, 1, 2, 3, 3, 5\}$.

Let us define the same initial information state of each node at time $t = 0$: $\mathbf{x}(0) = \{0.5832, 0.74, 0.2348, 0.7350, 0.9706, 0.8669\}$ for both methods. By using a standard consensus algorithm [26], the consensus value $\bar{x} = 0.6884$ can be easily inferred.

It can be seen in Figure 7 that the Lagrange-based method in [1] takes a long time to achieve the consensus first and then track the constraint to obtain the stepsizes $\alpha_t, t = 1, ..., N-1$.

Now, with the same initial $\boldsymbol{\alpha}_t(0), t = 1, \ldots, 5$, for the iterative procedure of the proposed method, the $\boldsymbol{\alpha}_t$ after using the Algorithm 1 with the convergence trajectories of the $\boldsymbol{\alpha}_t$ are illustrated in Figure 8.

Figures 7 and 8 express the significant pros of our proposed method since the number of iterations is much less than that of Lagrange-based method. The consensus term is executed in advance from the start of the procedure and then try to reach the constraint term to figure out the expected values of the stepsizes $\boldsymbol{\alpha}_t = \{1, 0.5, 0.1027, 0.2, 0.3333\}$. Obviously, since the authors operate the proposed algorithm with the number of $\boldsymbol{\alpha}_t$ being $N - 1 = 5$, it is needed to run the next stage to eliminate the residual values. As can be seen clearly, the executive time for ALADIN-based method is significantly faster than the proposed method as showed in Figure 9.

Figure 9 shows that the ALADIN-based method approaches the destined values earlier than Lagrange-based method. Furthermore, in order to get the non-zero Laplacian eigenvalues of the given network, the Algorithm 2 is carried out to obtain vector of stepsizes $\boldsymbol{\alpha}_t = \{1, 0.5, 0.2, 0.3333\}$.

Finally, by constructing the Brand-and-Bound based method to solve the Problem 1, which has been proposed in [16], we achieve the vector of multiplicities $\mathbf{m} = \{1, 1, 1, 2\}$, hence deduce the Laplacian spectrum $sp(\mathbf{L}) = \{1, 2, 3, 3, 5\}$.

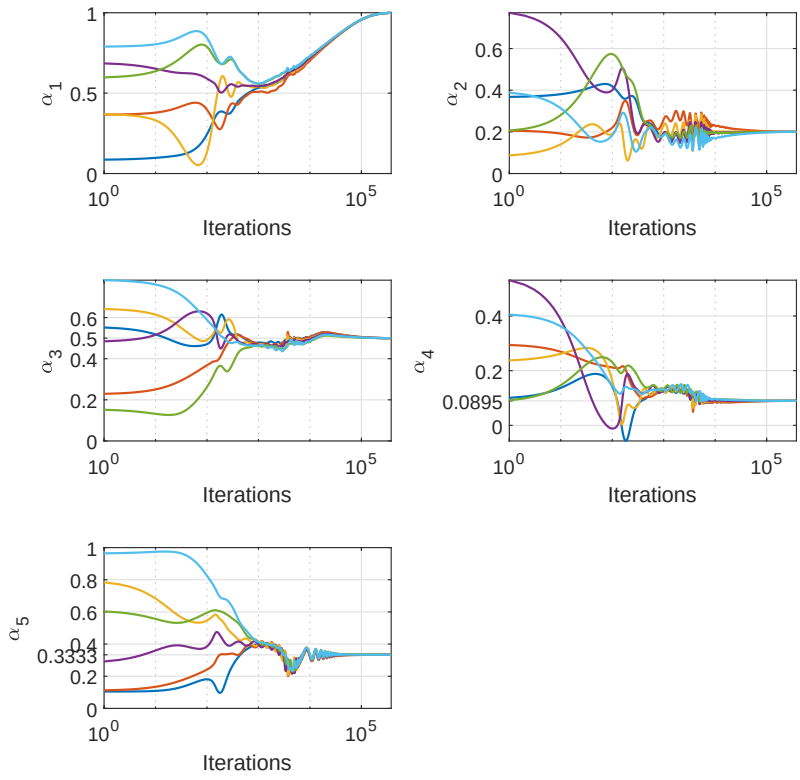

Figure 7. $\boldsymbol{\alpha}_t$ convergence trajectories implemented by Lagrange-based method.

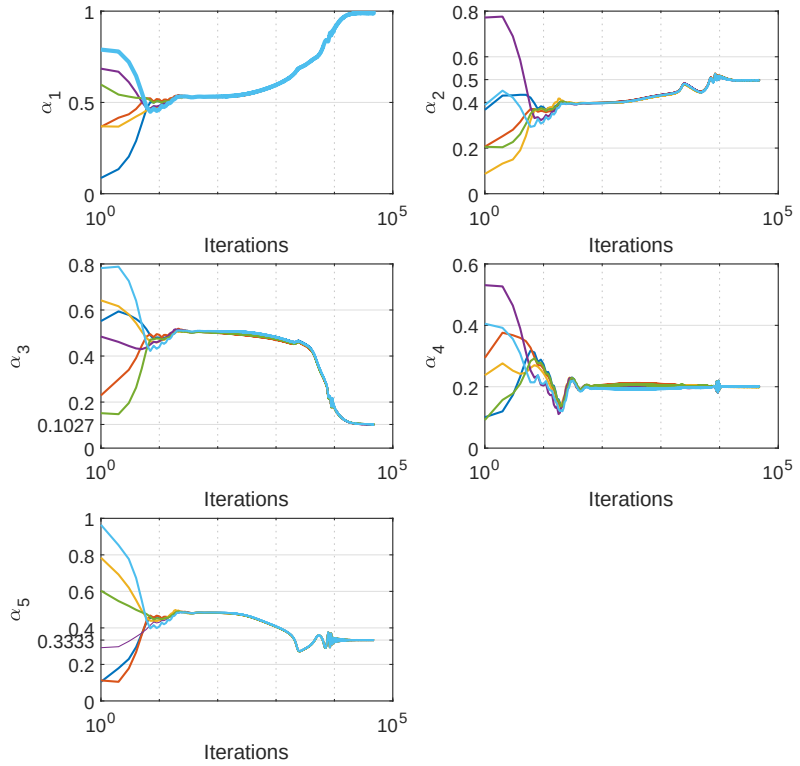

Figure 8. $\boldsymbol{\alpha}_t$ convergence trajectories implemented by the proposed method.

Now, let us see how the proposed procedure works via the table below.
Table 2 is obtained after executing the proposed procedure and the Lagrange-based method.

Table 2. The achievement of two methods in the sense of 6-node topology.

	ALADIN-Based Method	**Lagrange-Based Method**
$\boldsymbol{S}_1$	$\{1, 0.5, 0.1027, 0.2, 0.3333\}$	$\{0.9992, 0.2, 0.5001, 0.0895, 0.3333\}$
$\boldsymbol{S}_2$	$\{1, 0.5, 0.2, 0.3333\}$	$\{0.9992, 0.2, 0.5001, 0.3333\}$
$\mathbf{m}$	$1, 1, 1, 2$	
Iterations	33950	372800
$\mathcal{R}_{\mathbf{L}}$	14.2	

Notice that the proposed method gives results much better than the Lagrange-based method. Let us see that at the iteration of 33,950, the Lagrange-based method gives the set of $\boldsymbol{S}_1$ that has still not satisfied the constraint.

Recently, besides the Laplacian spectrum estimation basing on the optimization approaches, there are also some works that approximate the Laplacian spectrum via iterative dynamics process (taking random walk for an example). However, from our point of view, the dominant contribution of our proposed method is the possibility of implementation in a decentralized manner. Moreover, another method to faster the convergence rate of the Laplacian spectrum estimation procedure is the ADMM-based method, proposed in [16]. The important step in this work is to re-parameterize adequately the non-convex formulation into convex one. It is hard to compare the efficiency between ADMM-based method and the proposed method. Since, our study focuses on the non-convex formulation.

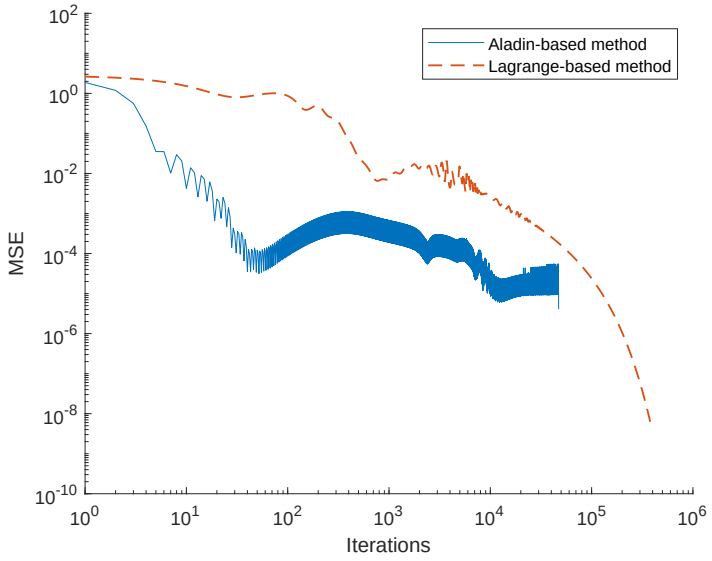

Figure 9. Convergence of the cost function according to its constraint.

5. Conclusions

In this paper, the authors have proposed a promising ALADIN-based method to find out the Laplacian spectrum of a given dynamic network in a distributed way. First and foremost, the study has assumed that the number of agents N in the network can be accumulated by random walk algorithm constructed in the configuration step. Briefly speaking, the proposed procedure is divided into 3 stages. The first stage is to determine the $N-1$ Laplacian eigenvalues. Then, retrieve the non-zero distinct Laplacian eigenvalues in stage 2 before estimating the corresponding multiplicities in stage 3. The ALADIN-based method is appropriate for carrying out the factorization of the average consensus matrix as factors of the Laplacian-based consensus matrices to minimize the disagreement

between neighbors on the values of $\alpha_{t,i}$. Then, the Laplacian eigenvalues can be defined by taking the inverse of these $\boldsymbol{\alpha}_t$. Herein, the authors are obviously interested in dealing with the non-convex optimization problems. From the simulation evaluation, it can be concluded that the proposed method converges much faster in comparison with the gradient descent method in [1] for the estimation of Laplacian spectrum.

Author Contributions: Conceptualization, methodology, funding acquisition, T.-M.-D.T.; writing—original draft preparation, T.-M.-D.T. and L.N.A.; resources L.N.A., writing—review and editing, T.-M.-D.T. and N.C.N.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by The University of Danang - University of Science and Technology, code number of Project: T2019-02-09.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Proof of Lemma 2

From now on, in order to avoid misunderstanding between the iterative step of the consensus algorithm $\mathbf{x}(h)$ and the iterative step of the optimization procedure k, let us denote $\mathbf{x}(h)$ by $\mathbf{x}_h$. From Section 2.1, we have:

$$\begin{aligned}
\mathbf{x}(h) - \bar{\mathbf{x}} &= \prod_{i=h}^{t+1} \mathbf{W}_i \mathbf{W}_t \mathbf{x}_{t-1} - \bar{\mathbf{x}} \\
&= \prod_{i=h}^{t+1} \mathbf{W}_i (\mathbf{I} - diag(\boldsymbol{\alpha}_t)\mathbf{L}) \mathbf{x}_{t-1} - \bar{\mathbf{x}} \\
&= \prod_{i=h}^{t+1} \mathbf{W}_i \mathbf{x}_{t-1} - \bar{\mathbf{x}} - \prod_{i=h}^{t+1} \mathbf{W}_i diag(\boldsymbol{\alpha}_t)\mathbf{L}\mathbf{x}_{t-1} \\
&= \prod_{i=h}^{t+1} \mathbf{W}_i \mathbf{x}_{t-1} - \bar{\mathbf{x}} - (\mathbf{x}_{t-1}^T \mathbf{L}^T \odot \prod_{i=h}^{t+1} \mathbf{W}_i)\boldsymbol{\alpha}_t
\end{aligned}$$

with $\odot$ being the Khaitri–Rao product. By employing the property of the Khatri–Rao product (Given matrix $\mathbf{A} \in R^{I \times F}$, and two vectors $\mathbf{b} \in R^{J \times 1}$, $\mathbf{d} \in R^{F \times 1}$, then $\mathbf{A}dieg(\mathbf{d})\mathbf{b} = (\mathbf{b}^T \odot \mathbf{A})\mathbf{d}$) in [1], the derivative of the Lagrange function is described as follows:

$$\frac{\partial \mathbf{H}_{1,t}}{\partial \boldsymbol{\alpha}_t} = \mathbf{L}\boldsymbol{\alpha}_t + \boldsymbol{\lambda} + \rho\boldsymbol{\Sigma}_t(\boldsymbol{\alpha}_t - \mathbf{y}_t) - diag(\mathbf{L}\mathbf{x}_{t-1}) \prod_{i=t+1}^{h} \mathbf{W}_i \boldsymbol{\beta}_t - c.diag(\mathbf{L}\mathbf{x}_{t-1}) \prod_{i=t+1}^{h} \mathbf{W}_i (\mathbf{x}(h) - \bar{\mathbf{x}})$$

Since $\boldsymbol{\delta}_h = \boldsymbol{\beta}_t$ and $\boldsymbol{\delta}_{h-1} = \mathbf{W}_h \boldsymbol{\delta}_h$ then $\prod_{i=t+1}^{h} \mathbf{W}_i \boldsymbol{\beta}_t = \boldsymbol{\delta}_t$.
Analogically, $\mathbf{e}_h = \mathbf{x}(h) - \bar{\mathbf{x}}$ and $\mathbf{e}_t = \mathbf{W}_{t+1}\mathbf{e}_{t+1}$.
On the other hand, $\mathbf{x}_{t-1} - \mathbf{x}_t = diag(\boldsymbol{\alpha}_t)\mathbf{L}\mathbf{x}_{t-1}$ then $\mathbf{L}\mathbf{x}_{t-1} = diag^{-1}(\boldsymbol{\alpha}_t)(\mathbf{x}_{t-1} - \mathbf{x}_t)$
Therefore, $\frac{\partial \mathbf{H}_{1,t}}{\partial \boldsymbol{\alpha}_t} = \mathbf{L}\boldsymbol{\alpha}_t + \boldsymbol{\lambda} + \rho\boldsymbol{\Sigma}_t(\boldsymbol{\alpha}_t - \mathbf{y}_t) - diag^{-1}(\boldsymbol{\alpha}_t)diag(\mathbf{x}_{t-1} - \mathbf{x}_t)\boldsymbol{\delta}_t - diag^{-1}(\boldsymbol{\alpha}_t)diag(\mathbf{x}_{t-1} - \mathbf{x}_t)\mathbf{e}_t$.

Appendix B. KKT Conditions of QP (11)

$$\frac{\delta \mathbf{H}_2}{\Delta \boldsymbol{\alpha}_t} = \mathbf{B}_t \Delta\alpha_t + g_t + \boldsymbol{\lambda}_{QP} + \mathbf{C}_t^T \eta_t = 0$$

$$\frac{\delta \mathbf{H}_2}{\mathbf{s}} = \boldsymbol{\lambda} - \boldsymbol{\lambda}_{QP} + \mu\mathbf{s} = 0$$

$$\frac{\delta \mathbf{H}_2}{\eta_t} = \mathbf{C}_t \Delta\boldsymbol{\alpha}_t = 0$$

$$\frac{\delta \mathbf{H}_2}{\boldsymbol{\lambda}_{QP}} = \sum_{t=1}^{h} \Delta\boldsymbol{\alpha}_t + \sum_{t=1}^{h} (\boldsymbol{\alpha}_t - \mathbf{y}_t) - \mathbf{s} = 0$$

References

1. Tran, T.; Kibangou, A.Y. Consensus-based Distributed Estimation of Laplacian Eigenvalues of Undirected Graphs. In Proceedings of the European Control Conference (ECC), Zurich, Switzerland, 17–19 July 2013; pp. 227–232.
2. Schwab, K. *The Fourth Industrial Revolution*; World Economic Forum: Cologny, Switzerland, 2016.
3. Godsil, C.; Royle, G. *Algebraic Graph Theory*; Springer: Berlin, Germany, 2001.
4. Merris, R. Laplacian matrices of a graph: A survey. *Linear Algebra Appl.* **1994**, *197*, 143–176. [CrossRef]
5. Friedler, M. Algebraic connectivity of graphs. *Czechoslov. Math. J.* **1973**, *23*, 298–305.
6. Olfati-Saber, R.; Fax, J.A.; Murray, R.M. Consensus and cooperation in networked multi-agent systems. *Proc. IEEE* **2007**, *95*, 215–233. [CrossRef]
7. Olfati-Saber, R.; Murray, R. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Autom. Control* **2004**, *49*, 1520–1533. [CrossRef]
8. Gulzar, M.; Rizvi, S.; Javed, M.Y.; Munir, U.; Asif, H. Multi-Agent Cooperative Control Consensus: A Comparative Review. *Electronics* **2018**, *7*, 22. [CrossRef]
9. Kempton, L.C.; Herrmann, G.; di Bernardo, M. Distributed optimisation and control of graph Laplacian eigenvalues for robust consensus via an adaptive multilayer strategy. *Int. J. Robust Nonlinear Control* **2017**, *27*, 1499–1525. [CrossRef]
10. Xiao, L.; Boyd, S. Fast linear iterations for distributed averaging. *Syst. Control Lett.* **2004**, *53*, 65–78. [CrossRef]
11. Kibangou, A. Finite-time average consensus based protocol for distributed estimation over awgn channels. In Proceedings of the IEEE Conference on Decision and Control (CDC), Orlando, FL, USA, 12–15 December 2011.
12. Kibangou, A. Graph Laplacian based matrix design for finite-time distributed average consensus. In Proceedings of the American Control Conference (ACC), Montreal, QC, Canada, 27–29 June 2012; pp. 1901–1906.
13. Abbas, W.; Egersredt, M. Robust graph Topologies for networked systems. In Proceedings of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NeCSYS), Santa Barbara, CA, USA, 13–14 September, 2012; pp. 85–90.
14. Wu, J.; Barahona, M.; Tan, Y.; Deng, H. Spectral Measure of Structural Robustness in Complex Networks. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2011**, *41*, 1244–1252. [CrossRef]
15. Klein, D.; Randić, M. Resistance distance. *J. Math. Chem.* **1993**, *12*, 81–95. [CrossRef]
16. Tran, T.; Kibangou, A.Y. Collaborative Network Monitoring by Means of Laplacian Spectrum Estimation and Average Consensus. *Int. J. Autom. Control Syst.* **2019**, *17*, 1826–1837. [CrossRef]
17. Zhao, C.; He, J.; Cheng, P.; Chen, J. Consensus-based energy management in smart grid with transmission losses and directed communication. *IEEE Trans. Smart Grid* **2017**, *8*, 2049–2061. [CrossRef]
18. Guo, L.; Zhao, C.; Low, S.H. Graph Laplacian Spectrum and Primary Frequency Regulation. In Proceedings of the 2018 IEEE Conference on Decision and Control (CDC), Miami Beach, FL, USA, 17–19 December 2018.
19. Franceschelli, M.; Martini, S.; Egerstedt, M.; Bicchi, A.; Giua, A. Observability and controllability verification in multi-agent systems through decentralized Laplacian spectrum estimation. In Proceedings of the IEEE Conference on Decision and Control (CDC), Atlanta, GA, USA, 15–17 December 2010; pp. 5775–5780.
20. Cooper, C.; Radzik, T.; Siantos, Y. Estimating network parameters using random walks. In Proceedings of the 2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN), Sao Carlos, Brazil, 21–23 November 2012; pp. 33–40.
21. Franceschelli, M.; Gasparri, A.; Giua, A.; Seatzu, C. Decentralized Estimation of Laplacian Eigenvalues in Multi-Agent Systems. *Automatica* **2013**, *49*, 1031–1036. [CrossRef]
22. Sahai, T.; Speranzon, A.; Banaszuk, A. Hearing the cluster of a graph: A distributed algorithm. *Automatica* **2012**, *48*, 15–24. [CrossRef]
23. Kibangou, A.Y.; Commault, C. Decentralized Laplacian Eigenvalues Estimation and Collaborative Network Topology Identification. In Proceedings of the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'12), Santa Barbara, CA, USA, 14–15 September 2012; pp. 7–12.
24. Tran, T.; Kibangou, A. Distributed estimation of Laplacian eigenvalues via constrained consensus optimization problems. *Syst. Control Lett.* **2015**, *80*, 56–62. [CrossRef]

25. Houska, B.; Frasch, J.; Diehl, M. An Augmented Lagrangian Based Algorithm for Distributed NonConvex Optimization. *SIAM J. Optim.* **2016**, *26*, 1101–1127. [CrossRef]
26. Xiao, L.; Boyd, S.; Kim, S. Distributed Average Consensus with Least-mean-square Deviation. *J. Parallel Distrib. Comput.* **2007**, *67*, 33–46. [CrossRef]
27. Tran, T.; Kibangou, A. Distributed Estimation of Graph Laplacian Eigenvalues by the Alternating Direction of Multipliers Method. In Proceedings of the 19th World Congress of the International Federation of Automatic Control, Cape Town, South Africa, 24–29 August 2014.
28. Chung, F.R.K. *Spectral Graph Theory*; American Mathematical Society: Providence, RI, USA, 1997.
29. Martin, N.; Frasca, P.; Canudas-De-Wit, C. Large-scale network reduction towards scale-free structure. *IEEE Trans. Netw. Sci. Eng.* **2018**, *14*, 1–12. [CrossRef]