mathematics

MDPI

Article

A Distributed Algorithm for the Assignment of the Laplacian Spectrum for Path Graphs

Gianfranco Parlangeli

Department of Engineering for Innovation, University of Salento, via per Monteroni, 73100 Lecce, Italy; gianfranco.parlangeli@unisalento.it; Tel.: +39-0832-297301

Abstract: In this paper, we bring forward a distributed algorithm for the assignment of a prescribed Laplacian spectrum for path graphs by means of asymmetric weight assignment. We first describe the meaningfulness and the relevance of this mathematical setting in modern technological applications, and some examples are reported, revealing its practical usefulness in a variety of applications. Then, the solution is derived both theoretically and through an algorithm. Special attention is devoted to a distributed implementation of the main algorithm, which is a valuable feature for several modern applications. Finally, the positivity is discussed, which is revealed to be a consequence of the strict interlacing property.

Keywords: Laplacian eigenvalues; path graphs; inverse eigenvalue problems

MSC: 93B55; 05C50; 94C15; 39A06; 37N35; 37M10

1. Introduction

In recent decades, an increasing number of research activities arose studying the behavior of interacting devices for applications in disparate technological areas, giving rise to the subject of networked systems [1–4]. Network topology became a fundamental new design paradigm, and mainly the spectral properties of some graph-related matrices, such as adjacency or Laplacian matrix, were researched.

The Laplacian matrix of a graph is a potent tool for the theoretical study and design of many contemporary technological applications where the graph encodes local interactions among agents, thus including [5,6] multi-agent systems, computer networks, sparse coding, and collaborative robotic networks. In such applications, the spectral properties of a graph-related matrix, and especially the Laplacian matrix, often play a key role in explaining some internal structure or condition, such as clustering, partitioning, consensus or synchronization [5,7,8]. Grounded Laplacian refers to the main submatrix that results from eliminating the i-th row and column of a Laplacian matrix. The performance of various applications is also directly correlated with the spectral features of a grounded Laplacian, which have undergone many studies in recent years as well [9,10]. In this paper, we shed a light on the class of Laplacian matrices associated with path graphs, which have the structure of (asymmetric) Jacobi matrices, and they constitute a class of graphs of great importance which has been widely studied over the years [11–13].

On the one hand, from a mathematical standpoint, this topic comes within the category of inverse eigenvalue problems, that is, the existence analysis and computation of structured matrices with assigned eigenstructure [14], which has a long history among mathematicians and it is still widely debated [15–17]. With reference to such literature, the contribution of this paper is the derivation of an algorithm, which can be distributed with respect to the nodes of the path, and this is a fundamental peculiarity for several modern applications, as detailed in Section 2.

On the other hand, however, the topic of the present paper stems from modern applications, and this makes the challenge different from the classical mathematical settings,

Citation: Parlangeli, G. A Distributed Algorithm for the Assignment of the Laplacian Spectrum for Path Graphs. *Mathematics* **2023**, *11*, 2359. https://doi.org/10.3390/math11102359

Academic Editor: Kinkar Chandra Das

Received: 28 March 2023
Revised: 16 May 2023
Accepted: 17 May 2023
Published: 18 May 2023

as described in [14]. It also has strong connections with control systems theory, and more specifically, it is close to the pole allocation through feedback [18–20]. As a final remark, it is worth noting that the knowledge of the Laplacian spectrum allows to monitor the system evolution and detect any anomaly caused by edge malfunctioning or malicious behaviors [21,22].

An early short version of this work was presented in [23]. The paper is organized as follows. In Section 2, after a brief introduction of the notation, we provide some motivating examples that are strongly related to the mathematical problem. In Section 3, the main results of the paper are derived; first, a global algorithm solving the stated problem is deduced using and inductive logic over n, then, it is rearranged to implement it in a distributed fashion, where each node can compute its own weights based only on data gathered by its neighborhood. Here, the analysis is thorough, including complete proofs for all statements. Finally, a discussion on the positivity of the gains provided by the algorithm closes the paper. Furthermore, this paper contains a much improved comprehensive and thorough treatment. A final section describes an illustrative example in the distributed setting for a network of six nodes.

Notation

We now briefly introduce the notation adopted in this paper and we report some basic facts of Graph Theory that are useful to understand the following sections. Given $A \in \mathbb{R}^{d_1 \times d_2}$, $[A]_{ij}$ denotes its (i,j)-th entry and $[A]_i$ its i-th column, while $(A)_{ij} \in \mathbb{R}^{(n-1)\times(n-1)}$ is the submatrix obtained by removing the i-th row and j-th column of A. The *Laplace expansion rule* for the computation of the determinant of a square matrix $A \in \mathbb{R}^{n\times n}$ is: $\det A = \sum_{j=1}^{n}(-1)^{i+j}[A]_{ij}\det(A)_{ij}$.

A nonnegative (positive) matrix $A \in \mathbb{R}^{n\times n}$ is a matrix with the property $[A]_{ij} \geq 0$ ($[A]_{ij} > 0$). A matrix $A \in \mathbb{R}^{n\times n}$ is combinatorially symmetric [24] or structurally symmetric when it holds that $[A]_{ij} \neq 0$ if and only if $[A]_{ji} \neq 0$. A matrix $A \in \mathbb{R}^{n\times m}$ is called generalized row stochastic matrix if there exists a $c \in \mathbb{R}$ so that $\sum_{j=1}^{n}[A]_{ij} = c$ for any $i = 1,\ldots,n$, and *zero row sum matrix* if such a c is $c = 0$. In this paper, we deal with *zero-row sum combinatorially symmetric matrices* [25]. In the following, we denote by Λ the *spectrum* of a matrix $A \in \mathbb{R}^{n\times n}$, namely the set of its eigenvalues, and by ρ_A the *spectral radius* of A, i.e., the maximum modulus of its eigenvalues. For a reference book on polynomials, the reader is referred to the book [26].

A graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ is a mathematical object made of a *vertex set* $\mathcal{V} = \{1,2,\ldots,n\}$ and an *edge set* $\mathcal{E} \subset \mathcal{V}\times\mathcal{V}$. A *path* connecting vertex j_1 with j_{k+1} is a subset of edges $\{(j_\ell,_{\ell+1})\}_{|_{\ell\in[1,k]}}$. Given a graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$, we define the *adjacency matrix* $A \in \mathbb{R}^{n\times n}$ as $[A]_{ij} = 1$ if $(i,j) \in \mathcal{E}$ and $[A]_{ij} = 0$ if $(i,j) \notin \mathcal{E}$, and the *weighted adjacency matrix* as $[A_w]_{ij} = \alpha_{ij}$ with $\alpha_{ij} > 0$ if $(i,j) \in \mathcal{E}$ and $[A_w]_{ij} = 0$ if $(i,j) \notin \mathcal{E}$. Finally, the *weighted Laplacian matrix* is defined as $[L_w]_{ij} = -[A_w]_{ij}$ if $i \neq j$ and $[L_w]_{ii} = \sum_{j=1,i\neq j}^{n}[A_w]_{ij}$. By construction, $L_w\mathbf{1} = \mathbf{0}$ for any $L_w \in \mathcal{L}_w$, i.e., the Laplacian is a *zero-row sum* matrix. Finally, the $\mathbb{R}^{(n-1)\times(n-1)}$ matrix obtained by removing the ℓ-th row and column of L_w is the so-called *grounded Laplacian matrix* (or *Dirichlet Laplacian* matrix) (see for information [9,27]), which we denote in the following by $L_w^{(\ell)}$.

Conversely, for a graph $\mathcal{G}$, we use the symbol $\mathcal{L}_w(\mathcal{G})$ to denote the set of all possible weighted Laplacian matrices L_w whose graph is $\mathcal{G}$; in other words, $\mathcal{L}_w(\mathcal{G})$ denotes the set of all $n \times n$ *zero-row sum $\mathcal{G}$-structured square matrices*.

2. Motivating Examples

In this section, we provide an abstract multi-agent system setting where the problem afforded in this paper is meaningful, and then, we provide some examples of several modern technological fields where the algorithms provided in this paper can be effectively adopted.

2.1. Collaborative Multi-Agent Systems and Consensus Networks

In the following, we describe the abstract multi-agent setting [5,28], which is the mathematical framework of the applications described soon after. Given a set of N variables $x_i(t)$, $i \in \{1, \dots, N\}$ updating as $\dot{x}_i(t) = u_i(t)$, where $u_i(t)$ is an update variable chosen by each node i. A graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ describes the communication among nodes, namely two *neighbor nodes* i, j which are connected through an edge $(i, j) \in \mathcal{E}$ are able to exchange their values (that is, node i sends $x_i(t)$ to node j and vice versa), so node i can choose $u_i(t) = \sum_{j \in \mathcal{N}_i} k_{ij}(x_j(t) - x_i(t))$, where k_{ij} are set by node i. The resulting dynamics of the team can be compactly written $\dot{\mathbf{x}}(t) = -L_\kappa \mathbf{x}(t)$ using a vector $\mathbf{x} \in \mathbb{R}^{\mathbf{N}}$ built as $[\mathbf{x}]_i(t) = x_i(t)$ for a L_κ, which belongs to the set $\mathcal{L}_w$, and the evolution of the network is governed by the spectrum of L_κ [29]. In this framework, the design of asymmetric gains was recently studied [30–32] to increase the convergence rate, which is a key feature; this was recently explored in [33–35].

Assume now that one node, e.g., node 1, takes the role of a leader and it adds a driving signal $v_1(t)$ to its update variable, namely:

$$u_1(t) = \sum_{j \in \mathcal{N}_1} k_{1j}(x_j(t) - x_1(t)) + v_1(t) \tag{1}$$

to take control of the team evolution [36]. This setting is known as *leader follower network*, [37–39] and its dynamics is described by:

$$\dot{\mathbf{x}}(t) = -L_\kappa \mathbf{x}(t) + \mathbf{e}_1 v_1(t) \tag{2}$$

$$y(t) = \mathbf{e}_1^T \mathbf{x}(t) \tag{3}$$

with $L_\kappa \in \mathcal{L}_w$ or, equivalently, using the transfer function:

$$G_1(s) = \mathbf{e}_1^T (sI + L_\kappa)^{-1} \mathbf{e}_1 = \frac{n(s)}{d(s)}$$

where $d(s) = \det(sI + L_\kappa)$ and $n(s) = \mathbf{e}_1^T Adj(sI + L_\kappa)\mathbf{e}_1 = [Adj(sI + L_\kappa)]_{11} = \det(sI + L_k^{(1)})$, and hence, the zeros of $n(s)$ and $d(s)$ are the poles and zeros of system (2), (3), so that their allocation allows to set all the fundamental input-output properties of the team (such as the zero-pole distance and, in turn, to influence several dynamical properties of the team driven by a prescribed node, such as controllability and observability of the whole network from node 1). It is worth remarking that coprimeness between $n(s)$ and $d(s)$ is necessary for the full control of the team by the leader [40].

2.2. Some Examples of Applications Scenarios

The above setting is applied to several technological modern applications, for example: (a) Robotic networks. In this application area, the above setting is widely used to perform global actions (such as rendez-vous, deployment, formation control) without recurring to a supervisory device [41]. (b) Wireless sensor networks. Several fundamental issues of wireless sensor networks are solved using the above setting, for example the synchronization problem [42,43] and the average value of a measured environmental quantity [44]. (c) Vehicle platooning. The above setting is adopted to model the dynamics of vehicles platoon, where Laplacian and grounded Laplacian of path graphs are well suited to describe the common case of bidirectional non-cyclic interactions between the agents [45]. It is worth remarking that weight asymmetry has already been investigated [46] to improve the team performances. (d) Electrical power systems. The multi-agent setting is widely adopted when the numerous challenges of the evolution of the electric grid toward a smart grid are taken into account [47–49].

2.3. Problem Statement

Based on the motivating examples introduced so far, we introduce the Problem Statement afforded in the following.

Problem 1. *Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and two polynomials $a(s)$, $b(s)$ having zeros as $\lambda_i, \mu_i \in \mathbb{R}$ satisfying $0 < \mu_1 < \lambda_2 < \ldots < \mu_{n-1} < \lambda_n$, compute gains α_{ij}, such that $\alpha_{ij} = 0$ if $(i,j) \notin \mathcal{E}$ and $\alpha_{ij} > 0$ if $(i,j) \in \mathcal{E}$ satisfying $\det[sI - L_w] = a(s)$ and $\det[sI - L_w^{(1)}] = b(s)$.*

Remark 1. *The above Problem Statement is strictly related to the examples in Section 2.2, and does not include the possibility of overlap between Λ and $\Lambda^{(1)}$, which is an unfavorable condition in most applications (as, for example, it implies the loss of controllability and observability by the leader [50]).*

3. Problem Solution

The proposed solution follows an inductive approach over $n \in \mathbb{N}$. We start with the solution of $n = 2$ and $n = 3$, then we extend the solution to a generic n through an iterative algorithm.

3.1. Preliminary Results on n = 2 and n = 3

Consider $n = 2$, which is described by the following matrices:

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad A_w = \begin{bmatrix} 0 & \alpha_{12} \\ \alpha_{21} & 0 \end{bmatrix} \qquad L_w = \begin{bmatrix} \alpha_{12} & -\alpha_{12} \\ -\alpha_{21} & \alpha_{21} \end{bmatrix}, \qquad L_w^{(1)} = \begin{bmatrix} \alpha_{21} \end{bmatrix}. \tag{4}$$

For any $0 < \mu_1 < \lambda_2$, the solution of the problem can be parametrized as:

$$L_w = \begin{bmatrix} \lambda_2 - \mu_1 & \mu_1 - \lambda_2 \\ -\mu_1 & \mu_1 \end{bmatrix} \tag{5}$$

We now consider case $n = 3$ when ℓ is a leaf, that is:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad A_w = \begin{bmatrix} 0 & \alpha_{12} & 0 \\ \alpha_{21} & 0 & \alpha_{23} \\ 0 & \alpha_{32} & 0 \end{bmatrix} \text{ and } L_w = \begin{bmatrix} \alpha_{12} & -\alpha_{12} & 0 \\ -\alpha_{21} & \alpha_{21} + \alpha_{23} & -\alpha_{23} \\ 0 & -\alpha_{32} & \alpha_{32} \end{bmatrix} \tag{6}$$

with associated grounded Laplacian $L_w^{(1)} = \begin{bmatrix} \alpha_{21} + \alpha_{23} & -\alpha_{23} \\ -\alpha_{32} & \alpha_{32} \end{bmatrix}$.

The computation of the characteristic polynomial shows that:

$$\det[sI - L_w] = (s - \alpha_{12})\phi_2(s) - s\alpha_{21}\psi_2(s), \tag{7}$$

$$\det[sI - L_w^{(1)}] = \phi_2(s) - \alpha_{21}\psi_2(s) \tag{8}$$

where

$$\phi_2(s) = \det \begin{bmatrix} s - \alpha_{23} & \alpha_{23} \\ \alpha_{32} & s - \alpha_{32} \end{bmatrix} \qquad \text{and} \qquad \psi_2(s) = \det \begin{bmatrix} s - \alpha_{32} \end{bmatrix}.$$

Imposing Equations (7) and (8) for given $a(s)$ and $b(s)$ allows to compute the weights α_{12}, α_{21}, and the two polynomials $\phi_2(s)$, $\psi_2(s)$. In turn, by making use of the solution (5), it is easy to compute the remaining coefficients α_{23}, α_{32} from the polynomials $\phi_2(s)$, $\psi_2(s)$ obtained in the previous step. In the following, we prove that the above procedure can be extended to a general $n \in \mathbb{N}$.

3.2. A Recursive General Solution When ℓ Is a Leaf

We are now ready to derive a solution for a generic $n \in \mathbb{N}$. It follows an inductive logic over $\mathbb{N}$ and the solution is recursive.

Theorem 1. *Let $\mathcal{G}$ be a path graph and node 1 a leaf node, and $L_w \in \mathcal{L}_w$, $L_w^{(1)} \in \mathcal{L}_w^{(1)}$ be, respectively, its weighted Laplacian and grounded Laplacian. Take two polynomials $a(s) = s^n + a_{n-1}s^{n-1} + \ldots + a_1 s$ and $b(s) = s^{n-1} + b_{n-2}s^{n-2} + \ldots + b_1 s + b_0$ such that their zeros satisfy the Problem Statement (Section 2.3).*

There always exist $\alpha_{ij} \in \mathbb{R}$ such that $\det[sI - L_w] = a(s)$ and $\det\left[sI - L_w^{(1)}\right] = b(s)$, and they can be computed using Algorithm 1.

Algorithm 1: Computation of the edge weights as the solution of the Problem Statement (Section 2.3).

Data: $a(s)$, $b(s)$
Result: α_{ij}, $i, j = 1, \ldots, n$
initialization: $\phi^{(n)}(s) = a(s)$, $\psi^{(n)}(s) = b(s)$, $\alpha_{ij} = 0$, for all $i, j = 1, \ldots, n$;
for $i = 0, \ldots, n-2$ **do**
  $\alpha_{n-i-1,n-i} = \psi_{n-i-1}^{(n-i)} - \phi_{n-i-1}^{(n-i)}$;
  for $j = 1, \ldots, n-2$ **do**
    $\phi_j^{(n-i-1)} = \dfrac{\psi_{j-1}^{(n-i)} - \phi_j^{(n-i)}}{\alpha_{n-i-1,n-i}}$;
  end
  $\alpha_{n-i,n-i-1} = \phi_{n-i-2}^{(n-i-1)} - \psi_{n-i-1}^{(n-i)}$;
  for $j = 0, \ldots, n-3$ **do**
    $\psi_j^{(n-i-1)} = \dfrac{\phi_j^{(n-i-1)} - \psi_j^{(n-i)}}{\alpha_{n-i,n-i-1}}$;
  end
end

Proof. For a path graph, $A = \begin{bmatrix} 0 & 1 & 0 & \ldots \\ 1 & 0 & & \\ & & \ddots & \\ \ldots & 0 & 1 & 0 \end{bmatrix}$, $A_w = \begin{bmatrix} 0 & \alpha_{(n-1,)n} & 0 & \ldots \\ \alpha_{n,(n-1)} & 0 & & \\ & & \ddots & \alpha_{1,2} \\ \ldots & 0 & \alpha_{2,1} & 0 \end{bmatrix}$ and

$$L_w = \begin{bmatrix} \alpha_{(n-1),n} & -\alpha_{(n-1),n} & & \ldots \\ -\alpha_{n,(n-1)} & a_{(n-1),(n-1)} & & \\ & & \ddots & -\alpha_{1,2} \\ \ldots & 0 & -\alpha_{2,1} & \alpha_{2,1} \end{bmatrix}, L_w^{(1)} = \begin{bmatrix} a_{(n-1),(n-1)} & -\alpha_{(n-2),(n-1)} & \ldots \\ -\alpha_{(n-1),(n-2)} & \ddots & \\ & & -\alpha_{1,2} \\ \ldots & 0 & \alpha_{2,1} \end{bmatrix} \tag{9}$$

where $a_{(n-1),(n-1)} = \alpha_{n,(n-1)} + \alpha_{(n-2),(n-1)}$. Define:

$$\phi_n(s) = \det[sI - L_w] \text{ and } \psi_n(s) = \det\left[sI - L_w^{(1)}\right]; \tag{10}$$

using the Laplace rule for the determinant along the first line, one can further elaborate:

$$\det[sI - L_w] = (s - \alpha_{(n-1),n}) \det \begin{bmatrix} s - a_{(n-1),(n-1)} & \alpha_{(n-2),(n-1)} & \cdots \\ \alpha_{(n-1),(n-2)} & \ddots & \\ & & \alpha_{1,2} \\ \cdots & 0 & s - \alpha_{2,1} \end{bmatrix} -$$
$$- \alpha_{(n-1),n}\alpha_{n,(n-1)} \det \begin{bmatrix} s - a_{(n-2),(n-2)} & \alpha_{(n-3),(n-2)} & \cdots \\ \alpha_{(n-2),(n-3)} & \ddots & \\ & & \alpha_{1,2} \\ \cdots & 0 & s - \alpha_{2,1} \end{bmatrix} =$$
$$= (s - \alpha_{(n-1),n}) \left[\phi_{(n-1)}(s) - \alpha_{n,(n-1)}\psi_{(n-1)}(s)\right] - \alpha_{(n-1),n}\alpha_{n,(n-1)}\psi_{(n-1)}(s)$$

where $\phi_{(n-1)}(s)$ (respectively, $\psi_{(n-1)}(s)$) is the characteristic polynomial of the Laplacian (respectively, grounded Laplacian) of the path graph made of $(n-1)$ nodes once that node n is removed.

The above computation shows that the following iterative relations hold:

$$\phi_n(s) = (s - \alpha_{(n-1),n})\phi_{(n-1)}(s) - s\alpha_{n,(n-1)}\psi_{(n-1)}(s) \tag{11}$$
$$\psi_n(s) = \phi_{(n-1)}(s) - \alpha_{n,(n-1)}\psi_{(n-1)} \tag{12}$$

Let now $a(s) = s^n + a_{n-1}s^{n-1} + \ldots + a_1 s$ and $b(s) = s^{n-1} + b_{n-2}s^{n-2} + \ldots + b_1 s + b_0$ be two polynomials having $\{0, \lambda_2, \ldots, \lambda_n\}$ and $\{\mu_1, \mu_2, \ldots, \mu_{n-1}\}$ as zeros, respectively. Forcing $\phi_n(s) = a(s)$ and $\psi_n(s) = b(s)$ into Equations (11) and (12), one has:

$$\begin{cases} a(s) = (s - \alpha_{(n-1),n})\phi_{(n-1)}(s) - s\alpha_{n,(n-1)}\psi_{(n-1)}(s) \\ b(s) = \phi_{(n-1)}(s) - \alpha_{n,(n-1)}\psi_{(n-1)}(s) \end{cases} \tag{13}$$

and, exploiting their expanded form, namely $\phi_{(n-1)}(s) = s^{(n-1)} + \bar{\phi}_{n-2}s^{(n-2)} + \ldots + \bar{p}_1 s$ and $\psi_{(n-1)}(s) = s^{(n-2)} + \bar{\psi}_{n-3}s^{(n-3)} + \ldots + \bar{\psi}_1 s + \bar{\psi}_0$, it is possible to elaborate further (13) as:

$$\begin{aligned} a(s) &= s^n + (\bar{\phi}_{n-2} - \alpha_{(n-1),n} - \alpha_{n,(n-1)})s^{(n-1)} + (\bar{\phi}_{n-3} - \alpha_{(n-1),n}\bar{\phi}_{n-2} - \alpha_{n,(n-1)}\bar{\psi}_{n-3})s^{(n-2)} + \\ &\quad + \cdots + (\bar{\phi}_1 - \alpha_{(n-1),n}\bar{\phi}_2 - \alpha_{n,(n-1)}\bar{\psi}_1)s^2 + (-\alpha_{(n-1),n}\bar{\phi}_1 - \alpha_{n,(n-1)}\bar{\psi}_0)s, \\ b(s) &= s^{(n-1)} + (\bar{\phi}_{n-2} - \alpha_{n,(n-1)})s^{(n-2)} + (\bar{\phi}_{n-3} - \alpha_{n,(n-1)}\bar{\psi}_{n-3})s^{(n-2)} + \\ &\quad + \cdots + (\bar{\phi}_1 - \alpha_{n,(n-1)}\bar{\psi}_1)s - \alpha_{n,(n-1)}\bar{\psi}_0 \end{aligned} \tag{14}$$

and to see that it is possible to have the desired polynomials if the weights $\alpha_{(n-1),n}$, $\alpha_{n,(n-1)}$ and the coefficients $\bar{\phi}_i$ $\bar{\psi}_i$ satisfy:

$$\begin{aligned} \bar{\phi}_{n-2} - \alpha_{(n-1),n} - \alpha_{n,(n-1)} &= a_{n-1} \\ \bar{\phi}_{n-3} - \alpha_{(n-1),n}\bar{\phi}_{n-2} - \alpha_{n,(n-1)}\bar{\psi}_{n-3} &= a_{n-2} \\ \vdots &= \vdots \end{aligned}$$

and

$$\begin{aligned} \bar{\phi}_{n-2} - \alpha_{n,(n-1)} &= b_{n-2} \\ \bar{\phi}_{n-3} - \alpha_{n,(n-1)}\bar{\psi}_{n-3} &= b_{n-3} \\ \vdots &= \vdots \end{aligned}$$

Consider now the first two equations. Inserting the second into the first one gives $b_{n-2} - \alpha_{(n-1),n} = a_{n-1}$, so that we choose to set:

$$\alpha_{(n-1),n} = b_{n-2} - a_{n-1}. \tag{15}$$

Consider now the two equations at the second line. Inserting the second relation into the first one has $b_{n-3} - \alpha_{(n-1),n}\bar{\psi}_{n-3} = a_{n-2}$, and considering the previous choice of $\alpha_{(n-1),n}$ we obtain $\bar{\phi}_{n-2} = \dfrac{b_{n-3} - a_{n-2}}{b_{n-2} - a_{n-1}}$ and, proceeding iteratively backwards:

$$\bar{\phi}_j = \frac{b_{j-1} - a_j}{b_{n-2} - a_{n-1}} \qquad \text{for} \quad j = 1, \ldots, n-2. \tag{16}$$

Substituting these values into the second column, one obtains:

$$\alpha_{n,(n-1)} = \frac{b_{n-3} - a_{n-2}}{b_{n-2} - a_{n-1}} - b_{n-2} \tag{17}$$

and, in turn, with the above choices, it is possible to compute:

$$\bar{\psi}_j = \frac{\bar{\phi}_j - b_j}{\alpha_{n,(n-1)}} \qquad \text{for} \quad j = 0, \ldots, n-3 \tag{18}$$

as functions of the coefficients of $a(s)$ and $b(s)$.

The above procedure allows us to compute the edge weights $\alpha_{(n-1),n}$ and $\alpha_{n,(n-1)}$ together with the two polynomials $\phi_{(n-1)}(s)$ (respectively, $\psi_{(n-1)}(s)$) of the path graph of $(n-1)$ nodes obtained by removing node n.

Iterating recursively the previous procedure allows computing all coefficients $\alpha_{i,j}$, as follows.

First note that Equations (11) and (12) can be written in a general form for a Path Graph with $\mathcal{V} = \{1, \ldots, j\}$ choosing j as grounded node, namely:

$$\begin{cases} \phi_j(s) = (s - \alpha_{(j-1),j})\phi_{(j-1)}(s) - s\alpha_{j,(j-1)}\psi_{(j-1)}(s) \\ \psi_j(s) = \phi_{(j-1)}(s) - \alpha_{j,(j-1)}\psi_{(j-1)} \end{cases} \tag{19}$$

for $3 \leq j \leq n$ and $\psi_2(s) = s - \alpha_{2,1}$, $\phi_2(s) = s^2 - (\alpha_{1,2} + \alpha_{2,1})s$, which extend Equations (11) and (12).

Put the polynomials $\phi_{(n-1)}(s)$, $\psi_{(n-1)}(s)$ computed in the previous step as reference polynomials into the two Equation (19) for $j = n-1$. This allows to compute $\alpha_{(n-2),(n-1)}$ $\alpha_{(n-1),(n-2)}$ together with $\phi_{(n-2)}(s)$ and $\psi_{(n-2)}(s)$. Iterating this logic using (19) backwards until $j = 1$ allows effectively to obtain all α_{ij}, and it can be effectively encapsulated into an algorithm, as Algorithm 1. □

It is worth noting that Algorithm 1 must be iterated a finite number of steps, equal to the eccentricity of the leader node (namely, the length of a longest shortest path starting at the leader node). This makes the proposed algorithm well suited as a preliminary routine to be run before the main application.

4. A Distributed Implementation of Algorithm 1

One key remark for modern engineering and technological applications is that Algorithm 1 is suited to be implemented in a distributed fashion. In such a case, each node can retrieve the weights of its algorithm by processing data received by its neighbor only.

To achieve this goal, the leader should:

- Compute the polynomials $a(s)$ and $b(s)$ starting from the desired zeros and set them as reference polynomials.

- Compute:
$$\alpha_{(n-1),n} = b_{n-2} - a_{n-1}$$
to obtain the value of its own weight and the polynomial $\phi^{(n-1)}(s) = s^{(n-1)} + \bar{\phi}_{n-2}^{(n-1)} s^{n-2} + \ldots + \bar{\phi}_1^{(n-1)} s$ through the computation of its coefficients:
$$\bar{\phi}_j^{(n-1)} = \frac{b_{j-1} - a_j}{b_{n-2} - a_{n-1}} \qquad \text{for} \quad j = 1, \ldots, n-2.$$

- Transmit $\alpha_{(n-1),n}$, $\phi^{(n-1)}(s)$ and $b(s)$ to node $n-1$.

When node n ends the above elaboration, based on the received data, node $n-1$ can trigger its elaboration as follows:

- Retrieve $\alpha_{n,(n-1)}$ and $\psi^{(n-1)}(s) = s^{(n-2)} + \bar{\psi}_{n-3}^{(n-1)} s^{(n-3)} + \ldots + \bar{\psi}_1^{(n-1)} s + \bar{\psi}_0^{(n-1)}$ by performing the following elaboration:
$$\begin{aligned} \alpha_{n,(n-1)} &= \bar{\phi}_{n-2}^{(n-1)} - b_{n-2} \\ \bar{\psi}_j^{(n-1)} &= \frac{\bar{\phi}_j - b_j}{\alpha_{n,(n-1)}} \qquad \text{for} \quad j = 0, \ldots, n-3. \end{aligned} \tag{20}$$

- Then:
$$\begin{aligned} \alpha_{(n-2),(n-1)} &= \bar{\psi}_{n-3}^{(n-1)} - \bar{\phi}_{n-2}^{(n-1)} \\ \bar{\phi}_j^{(n-2)} &= \frac{\psi_{j-1}^{(n-1)} - \phi_j^{(n-1)}}{\bar{\psi}_{n-3}^{(n-1)} - \bar{\phi}_{n-2}^{(n-1)}} \qquad \text{for} \quad j = 1, \ldots, n-3. \end{aligned} \tag{21}$$

- Transmit $\alpha_{(n-2),(n-1)}$, $\phi^{(n-2)}(s) = s^{(n-2)} + \bar{\phi}_{n-3}^{(n-2)} s^{(n-3)} + \ldots + \bar{\phi}_1^{(n-2)} s$ and $\psi^{(n-1)}(s)$ to node $n-2$.

Based on the above elaboration, node $n-1$ is able compute its own weights $\alpha_{n,(n-1)}$ and $\alpha_{(n-2),(n-1)}$ through an elaboration on the received data only. The above steps can be generalized using (19), and hence, it is possible to organize the algorithm so that it can be executed in a distributed fashion, as follows. Consider a node $\ell = 2, \ldots, n-1$; upon receipt of $\alpha_{\ell,\ell+1}$, $\phi^{(\ell)}(s)$ and $\psi^{(\ell+1)}(s)$ from node $(\ell+1)$, node ℓ should:

- Retrieve $\alpha_{(\ell+1),\ell}$ and $\psi^{(\ell)}(s) = s^{(\ell-1)} + \bar{\psi}_{\ell-2}^{(\ell)} s^{(\ell-2)} + \ldots + \bar{\psi}_1^{(\ell)} s + \bar{\psi}_0^{(\ell)}$ by performing the following elaboration:
$$\begin{aligned} \alpha_{(\ell+1),\ell} &= \bar{\phi}_{\ell-1}^{(\ell)} - \bar{\psi}_{\ell-1}^{(\ell+1)} \\ \bar{\psi}_j^{(\ell)} &= \frac{\bar{\phi}_j^{(\ell)} - \bar{\psi}_j^{(\ell+1)}}{\alpha_{(\ell+1),\ell}} \qquad \text{for} \quad j = 0, \ldots, \ell-2 \end{aligned} \tag{22}$$

- Then:
$$\begin{aligned} \alpha_{(\ell-1),\ell} &= \bar{\psi}_{\ell-2}^{(\ell)} - \bar{\phi}_{\ell-1}^{(\ell)} \\ \bar{\phi}_j^{(\ell-1)} &= \frac{\bar{\psi}_{j-1}^{(\ell)} - \bar{\phi}_j^{(\ell)}}{\bar{\psi}_{\ell-2}^{(\ell)} - \bar{\phi}_{\ell-1}^{(\ell)}} \qquad \text{for} \quad j = 1, \ldots, n-3. \end{aligned} \tag{23}$$

- Transmit $\alpha_{(\ell-1),\ell}$, $\phi^{(\ell-1)}(s) = s^{\ell-1} + \bar{\phi}_{\ell-2}^{(\ell-1)} s^{\ell-2} + \ldots + \bar{\phi}_1^{(\ell-1)} s$ and $\psi^{\ell}(s)$ to node $\ell-1$.

The above procedure allows each node ℓ to compute its own gains $\alpha_{(\ell+1),\ell}$, $\alpha_{(\ell-1),\ell}$ through an elaboration of local information only (namely, data received from its neighbor).

The technical requirement to achieve this is that each node j should be able to send two vectors encoding the coefficients of the two polynomials $\phi^{(j-1)}(s)$, $\psi^{(j)}(s)$.

The architecture of the distributed implementation of Algorithm 1 is depicted in Figure 1. Finally, it is worth remarking that the algorithm has finite time execution equal to n time steps so that it can be run as a preliminary routine before other operations in applications.

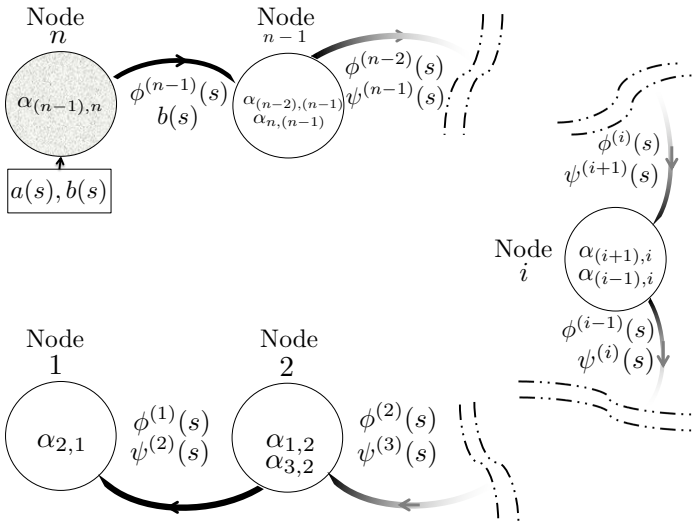

Figure 1. Sketch of the distributed architecture of Algorithm 1. Each node receives its reference polynomials, computes its own gains, and transmit the remainder polynomials to its neighbor.

5. A Final Remark on the Solution of Algorithm 1

In this section, we discuss the sign of the weights obtained as solutions of Algorithm 1, and it is revealed that the condition $\alpha_{i,j} \geq 0$ is a consequence of the relative location of λ_i, μ_i in the complex plane. In other words, if the interlacing property is set for λ_i, μ_i beforehand, then the solution of Algorithm 1 always has the property $\alpha_{i,j} \geq 0$.

Theorem 2. *Let* $\phi(s) = s^n + \phi_{n-1}s^{n-1} + \phi_{n-2}s^{n-2} + \ldots \phi_1 s$ *and* $\psi(s) = s^{n-1} + \psi_{n-2}s^{n-2} + \ldots \psi_1 s + \psi_0$ *be two polynomials satisfying the Problem Statement (Section 2.3). Then, Algorithm 1 always provides positive weights* $\alpha_{i,j} \in \mathbb{R}_+$ *satisfying* $\det[sI - L_w] = \phi(s)$ *and* $\det[sI - L_w^{(1)}] = \psi(s)$.

Proof. Algorithm 1 provides $\alpha_{i,j}$ by solving recursively Equation (13), so we prove the statement by showing that, if $a(s)$ and $b(s)$ satisfy the interlacing property, then the solutions of Equation (13) are positive weights $\alpha_{(n-1,)n}$ $\alpha_{n,(n-1)}$ and a pair of polynomials $\phi_{(n-1)}(s)$, $\psi_{(n-1)}(s)$, which in turn satisfy the interlacing property between themselves. This allows to trigger a recursive logic, which proves that all $\alpha_{i,j}$ are positive.

From Equation (13), it is possible to obtain $a(s) - sb(s) = -\alpha_{(n-1),n}\phi_{(n-1)}(s)$ so that $\alpha_{n-1,n} = a_{n-1} - b_{n-2}$ and, considering that $a_{n-1} = -\sum_{i=1}^{n-1}\lambda_i$ and analogously for b_{n-2}, we obtain $\sum_{i=1}^{n}\lambda_i - \sum_{i=1}^{n-1}\mu_i = \sum_{i=1}^{n-1}\lambda_{i+1} - \mu_i$, which is positive under the strict interlacing condition between $\{\lambda_i\}$ and μ_i provided by the Problem Statement. Consider now $\hat{\phi}(s) = \dfrac{a(s) - sb(s)}{-\alpha_{(n-1),n}s}$, which is a monic polynomial of degree $n-1$. Computing $\hat{\phi}(s)$ at several points $s = \lambda_i$ and $s = \mu_i$, one has: $\hat{\phi}(\lambda_n) = \frac{b(\lambda_n)}{\alpha_{(n-1),n}} = \frac{\prod_{i=1}^{n-1}(\lambda_n - \mu_i)}{\alpha_{(n-1),n}} > 0$ because $(\lambda_n - \mu_i) > 0$ and $\alpha_{(n-1),n} > 0$ under (13), and $\hat{\phi}(\mu_{n-1}) = \frac{a(\mu_{n-1})}{-\alpha_{(n-1),n}} = \frac{\prod_{i=2}^{n}(\mu_{n-1} - \lambda_i)}{-\alpha_{(n-1),n}} > 0$, $\hat{\phi}(\lambda_{n-1}) = \frac{b(\lambda_{n-1})}{\alpha_{(n-1),n}} = \frac{\prod_{i=1}^{n-1}(\lambda_{n-1} - \mu_i)}{\alpha_{(n-1),n}} < 0$, and $\hat{\phi}(\mu_{n-1}) = \frac{a(\mu_{n-1})}{-\alpha_{(n-1),n}} = \frac{\prod_{i=2}^{n}(\mu_{n-1} - \lambda_i)}{-\alpha_{(n-1),n}} < 0$, so that, if ν_{n-1} denotes the largest zero of $\hat{\phi}(s)$, then $\nu_{n-1} \in (\lambda_{n-1}, \mu_{n-1})$. If ν_i contains any zero of $\hat{\phi}(s)$,

the computation of $\hat{\phi}(\lambda_i)$ and $\hat{\phi}(\mu_i)$ backwards (iterating the above reasonings from $i-2$ to 0) allows to obtain $\nu_i \in (\lambda_i, \mu_i)$.

Consider now the second of Equation (13), which can be written as $b(s) - \phi_{n-1}(s) = -\alpha_{n,n-1}\psi_{n-1}(s)$ and the zeros of $b(s)$, $\phi_{n-1}(s)$ satisfy $0 < \nu_1 < \mu_1 < \cdots < \nu_{n-1} < \mu_{n-1}$. For the same reasons as before, $\alpha_{n,n-1} = \sum_{i=1}^{n-1}(\mu_i - \nu_i) > 0$. Building the polynomial $\hat{\psi}(s) = \dfrac{b(s) - \phi_{n-1}(s)}{-\alpha_{n,(n-1)}}$ and defining v_i as the zeros of $\hat{\psi}(s)$, it is possible to prove analogously that $0 < v_1 < \nu_1 < \cdots < v_{n-2} < \nu_{n-1}$. □

6. An Illustrative Example

In this section, we provide an illustrative example on the distributed implementation of the Algorithm on a network of six nodes.

Consider that a leaf of a path graph of six nodes starts the algorithm setting $a(s) = s(s-2)(s-4)(s-6)(s-8)(s-10) = s^6 - 30s^5 + 340s^4 - 1800s^3 + 4384s^2 - 3840s$ and $b(s) = (s-1)(s-3)(s-5)(s-7)(s-9) = s^5 - 25s^4 + 230s^3 - 950s^2 + 1689s - 945$.

According to Section 4, it computes $\alpha_{56} = b_5 - a_4 = 5$ and $\phi^{(5)}(s) = s^5 - 22s^4 + 170s^3 - 539s^2 + 579s$. Finally, it sends α_{56}, $\phi^{(5)}(s)$ and $b(s)$.

Once that node 5 receives its data, it computes: $\alpha_{65} = 3$, $\psi^{(5)}(s) = s^4 - 20s^3 + 137s^2 - 370s + 315$, and then, $\alpha_{45} = 2$, $\phi^{(4)}(s) = s^4 - 16.5s^4 + 84.5s^2 - 132s$. It sends α_{45}, $\phi^{(4)}(s)$ and $\psi^{(5)}(s)$ to node 4.

The Algorithm prosecutes at node 4 with the values $\alpha_{54} = 3.5$, $\psi^{(4)}(s) = s^3 - 15s^3 + 68s^2 - 90$, and then, $\alpha_{34} = 1.5$, $\phi^{(3)}(s) = s^3 - 11s^2 + 28s$. It sends data to node 3, which computes $\alpha_{43} = 4$, $\psi^{(3)}(s) = s^2 - 10s + 22.5$, and then, $\alpha_{23} = 1$, $\phi^{(2)}(s) = s^2 - 5.5s$, and sends data to node 2, which in turn computes $\alpha_{32} = 4.5$, $\psi^{(2)}(s) = s - 5.5$, and then, $\alpha_{12} = 5$.

The resulting Laplacian matrix is as follows:

$$\mathcal{L} = \begin{pmatrix} 5 & -5 & 0 & 0 & 0 & 0 \\ -0.5 & 5 & -4.5 & 0 & 0 & 0 \\ 0 & -1 & 5 & 4 & 0 & 0 \\ 0 & 0 & -1.5 & 5 & -3.5 & 0 \\ 0 & 0 & 0 & -2 & 5 & -3 \\ 0 & 0 & 0 & 0 & -5 & 5 \end{pmatrix}.$$

7. Conclusions

This paper proposes an algorithm for the assignment of the Laplacian eigenvalues by a suitable choice of edge gains. An iterative algorithm is derived first, after which a distributed implementation is sought, and its usefulness is also shown using an illustrative example. Several research directions are currently under investigation, which includes more complex kinematics models, such as [51], as well as the analysis of several different graph topologies [1].

Funding: This work was partially supported by the European Union's Horizon 2020 research and innovation program under the project EUMarineRobots: Marine robotics research infrastructure network, grant agreement N.731103 (call H2020-INFRAIA-2017-1-two-stage).

Data Availability Statement: The datasets generated for this study are available on request to the corresponding author.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Bullo, F. *Lectures on Network Systems*; Kindle Direct Publishing: Seattle, DC, USA, 2020; Volume 1.
2. Mullender, S. *Distributed Systems*; ACM: New York, NY, USA, 1990.

3. Qu, Z. *Cooperative Control of Dynamical Systems: Applications to Autonomous Vehicles*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 3.
4. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Comput. Netw.* **2008**, *52*, 2292–2330. [CrossRef]
5. Bullo, F. *Lectures on Network Systems*, 1.6 ed.; Kindle Direct Publishing: Seattle, DC, USA, 2022.
6. Gao, S.; Tsang, I.W.H.; Chia, L.T. Laplacian sparse coding, hypergraph laplacian sparse coding, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *35*, 92–104. [CrossRef] [PubMed]
7. Ng, A.; Jordan, M.; Weiss, Y. On spectral clustering: Analysis and an algorithm. In *NIPS'01: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, Vancouver, BC, Canada, 3–8 December 2001*; MIT Press: Cambridge, MA, USA, 2001; Volume 14, pp. 849–856.
8. Spielman, D.A. Spectral graph theory and its applications. In Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07), Providence, RI, USA, 21–23 October 2007; pp. 29–38.
9. Pirani, M.; Sundaram, S. On the smallest eigenvalue of grounded Laplacian matrices. *IEEE Trans. Autom. Control* **2015**, *61*, 509–514. [CrossRef]
10. Xia, W.; Cao, M. Analysis and applications of spectral properties of grounded Laplacian matrices for directed networks. *Automatica* **2017**, *80*, 10–16. [CrossRef]
11. Wu, X. A divide and conquer algorithm on the double dimensional inverse eigenvalue problem for Jacobi matrices. *Appl. Math. Comput.* **2012**, *219*, 3840–3846. [CrossRef]
12. Wei, Y.; Dai, H. An inverse eigenvalue problem for Jacobi matrix. *Appl. Math. Comput.* **2015**, *251*, 633–642. [CrossRef]
13. Zhang, J.; Wei, G. Reconstruction of Jacobi matrices with mixed spectral data. *Linear Algebra Its Appl.* **2020**, *591*, 44–60. [CrossRef]
14. Chu, M.T.; Golub, G.H. *Inverse Eigenvalue Problems: Theory, Algorithms, and Applications*; Oxford University Press: Oxford, UK, 2005; Volume 13.
15. Smillie, J. Competitive and cooperative tridiagonal systems of differential equations. *SIAM J. Math. Anal.* **1984**, *15*, 530–534. [CrossRef]
16. Ghanbari, K. A survey on inverse and generalized inverse eigenvalue problems for Jacobi matrices. *Appl. Math. Comput.* **2008**, *195*, 355–363. [CrossRef]
17. Cai, J.; Chen, J. Iterative solutions of generalized inverse eigenvalue problem for partially bisymmetric matrices. *Linear Multilinear Algebra* **2017**, *65*, 1643–1654. [CrossRef]
18. Maarouf, H. The resolution of the equation XA + XBX = HX and the pole assignment problem: A general approach. *Automatica* **2017**, *79*, 162–166. [CrossRef]
19. Padula, F.; Ferrante, A.; Ntogramatzidis, L. Eigenstructure assignment in linear geometric control. *Automatica* **2021**, *124*, 109363. [CrossRef]
20. Katewa, V.; Pasqualetti, F. Minimum-gain Pole Placement with Sparse Static Feedback. *IEEE Trans. Autom. Control.* **2020**, *66*, 3445–3459. [CrossRef]
21. Valcher, M.E.; Parlangeli, G. On the effects of communication failures in a multi-agent consensus network. In Proceedings of the 2019 23rd International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 9–11 October 2019; pp. 709–720.
22. Parlangeli, G. A Supervisory Algorithm Against Intermittent and Temporary Faults in Consensus-Based Networks. *IEEE Access* **2020**, *8*, 98775–98786. [CrossRef]
23. Parlangeli, G. Laplacian eigenvalue allocation by asymmetric weight assignment for path graphs. In Proceedings of the 2022 26th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 19–21 October 2022; pp. 503–508.
24. Maybee, J.S. Combinatorially symmetric matrices. *Linear Algebra Its Appl.* **1974**, *8*, 529–537. [CrossRef]
25. Boukas, A.; Feinsilver, P.; Fellouris, A. On the Lie structure of zero row sum and related matrices. *Random Oper. Stoch. Equations* **2015**, *23*, 209–218. [CrossRef]
26. Barbeau, E.J. *Polynomials*; Springer Science & Business Media: Berlin, Germany, 2003.
27. Barooah, P.; Hespanha, J.P. Graph effective resistance and distributed control: Spectral properties and applications. In Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, CA, USA, 13–15 December 2006; pp. 3479–3485.
28. Olfati-Saber, R.; Fax, A.J.; Murray, R. Consensus and Cooperation in Networked Multi-Agent Systems. *Proc. IEEE* **2007**, *95*, 215–233. [CrossRef]
29. Liu, Z.; Guo, L. Synchronization of multi-agent systems without connectivity assumptions. *Automatica* **2009**, *45*, 2744–2753. [CrossRef]
30. Hao, H.; Barooah, P. Improving convergence rate of distributed consensus through asymmetric weights. In Proceedings of the 2012 American Control Conference (ACC), Montreal, QC, Canada, 27–29 June 2012; pp. 787–792.
31. Sardellitti, S.; Giona, M.; Barbarossa, S. Fast distributed average consensus algorithms based on advection-diffusion processes. *IEEE Trans. Signal Process.* **2009**, *58*, 826–842. [CrossRef]
32. Chen, Y.; Tron, R.; Terzis, A.; Vidal, R. Corrective consensus with asymmetric wireless links. In Proceedings of the 2011 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 12–15 December 2011; pp. 6660–6665.
33. Song, Z.; Taylor, D. Asymmetric Coupling Optimizes Interconnected Consensus Systems. *arXiv* **2021**, arXiv:2106.13127.
34. Parlangeli, G.; Valcher, M.E. Leader-controlled protocols to accelerate convergence in consensus networks. *IEEE Trans. Autom. Control* **2018**, *63*, 3191–3205. [CrossRef]

35. Parlangeli, G.; Valcher, M.E. Accelerating consensus in high-order leader-follower networks. *IEEE Control Syst. Lett.* **2018**, *2*, 381–386. [CrossRef]
36. Herman, I.; Martinec, D.; Sebek, M. Zeros of transfer functions in networked control with higher-order dynamics. *IFAC Proc. Vol.* **2014**, *47*, 9177–9182. [CrossRef]
37. Torres, J.A.; Roy, S. Graph-theoretic analysis of network input–output processes: Zero structure and its implications on remote feedback control. *Automatica* **2015**, *61*, 73–79. [CrossRef]
38. Roy, S.; Torres, J.A.; Xue, M. Sensor and actuator placement for zero-shaping in dynamical networks. In Proceedings of the 2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, NV, USA, 12–14 December 2016, pp. 1745–1750.
39. Xue, M.; Roy, S. Input-output properties of linearly-coupled dynamical systems: Interplay between local dynamics and network interactions. In Proceedings of the 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia, 12–15 December 2017, pp. 487–492.
40. Rahmani, A.; Ji, M.; Mesbahi, M.; Egerstedt, M. Controllability of multi-agent systems from a graph-theoretic perspective. *SIAM J. Control Optim.* **2009**, *48*, 162–186. [CrossRef]
41. Bullo, F.; Cortés, J.; Martínez, S. *Distributed Control of Robotic Networks*; Applied Mathematics Series; Princeton University Press: Princeton, NJ, USA, 2009.
42. Swain, A.R.; Hansdah, R. A model for the classification and survey of clock synchronization protocols in WSNs. *Ad Hoc Netw.* **2015**, *27*, 219–241. [CrossRef]
43. Schenato, L.; Fiorentin, F. Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks. *Automatica* **2011**, *47*, 1878–1886. [CrossRef]
44. Oliveira, L.M.; Rodrigues, J.J. Wireless Sensor Networks: A Survey on Environmental Monitoring. *J. Commun.* **2011**, *6*, 143–151. [CrossRef]
45. Stüdli, S.; Seron, M.M.; Middleton, R.H. From vehicular platoons to general networked systems: String stability and related concepts. *Annu. Rev. Control* **2017**, *44*, 157–172. [CrossRef]
46. Herman, I.; Knorn, S.; Ahlén, A. Disturbance scaling in bidirectional vehicle platoons with different asymmetry in position and velocity coupling. *Automatica* **2017**, *82*, 13–20. [CrossRef]
47. Farhangi, H. The path of the smart grid. *IEEE Power Energy Mag.* **2009**, *8*, 18–28. [CrossRef]
48. Fang, X.; Misra, S.; Xue, G.; Yang, D. Smart grid—The new and improved power grid: A survey. *IEEE Commun. Surv. Tutorials* **2011**, *14*, 944–980. [CrossRef]
49. Molzahn, D.K.; Dörfler, F.; Sandberg, H.; Low, S.H.; Chakrabarti, S.; Baldick, R.; Lavaei, J. A survey of distributed optimization and control algorithms for electric power systems. *IEEE Trans. Smart Grid* **2017**, *8*, 2941–2962. [CrossRef]
50. Mesbahi, M.; Egerstedt, M. *Graph Theoretic Methods in Multiagent Networks*; Princeton University Press: Princeton, NJ, USA, 2010.
51. Parlangeli, G.; Indiveri, G. Single range observability for cooperative underactuated underwater vehicles. *Annu. Rev. Control* **2015**, *40*, 129–141. [CrossRef]