Distributed Network Design for Laplacian Eigenvalue Placement

Victor M. Preciado, *Member, IEEE,* and Michael M. Zavlanos, *Member, IEEE*

***Abstract*—We propose a distributed iterative algorithm in which a group of n autonomous agents self-organize the structure of their communication network in order to control the network's Laplacian eigenvalue spectrum. We assume that every agent has only access to a local ("myopic") view of the network around it and that there is no centralized coordinator. With every iteration of our algorithm, the agents share *local* information about their myopic views of the network in order to distributedly find the most beneficial *global* edge addition/deletion, defined as the one that minimizes a pseudometric defined in the space of Laplacian spectra. The proposed pseudometric is defined in terms of the Laplacian spectral moments and allows for an efficient distributed implementation. The proposed approach is greedy in nature and stable by construction, i.e., it locally minimizes the distance of the network's eigenvalue spectrum to a desired spectrum. We illustrate the performance of our approach with several numerical simulations.**

***Index Terms*—Complex network design, algebraic graph theory, eigenvalue placement, distributed control.**

I. Introduction

A wide variety of complex networks composed of autonomous agents are able to display a remarkable level of self-organization despite the absence of a centralized coordinator [1]. For example, the intricate structure of many biological, social and economic networks, emerges as the result of local interactions between agents aiming to optimize their local utilities [2]. In most real cases, these agents have only access to myopic information about the structure of the network around them. Despite the limited information accessible to each agent, most of these "self-engineered" networks are able to efficiently satisfy their functional requirements.

The behavior of many networked dynamical processes, such as information spreading, synchronization, or decentralized coordination, is directly related to the network eigenvalue spectra [3]. In particular, the spectrum of the Laplacian matrix of a network plays a key role in the analysis of synchronization in networks of nonlinear oscillators [4], as well as in the behavior of many distributed algorithms [5], and decentralized control problems [6], [7]. Motivated by the relationship between a network's eigenvalue spectrum and the behavior of dynamical processes evolving in it, we propose a distributed iterative algorithm in which a group of autonomous agents self-organize the structure of their communication network in order to control the network's eigenvalue spectrum. The evolution of the graph is ruled by a decentralized decision process in which agents share limited information about their myopic vision of the network to decide which network adjustment is most beneficial globally.

Optimization of network eigenvalues has been studied by several authors in both centralized [8], [9], [10] and decentralized settings [11]. In these papers, the objective is usually to find the weights associated to the edges of a given network in order to optimize eigenvalues of particular relevance, such as the Laplacian spectral gap or spectral radius, i.e., the second smallest and largest eigenvalues of the Laplacian matrix, respectively. In many other graph-theoretical applications, not only isolated eigenvalues are relevant, but the whole spectrum plays a role. For example, the steady-state mean-square deviation in noisy consensus algorithms depends on the whole Laplacian spectrum [12]. Similarly, the effective resistance of electrical networks [13], and the mean first-passage time in random walks [14] also depend on the whole Laplacian spectrum. Therefore, the development of algorithms aiming to design the whole Laplacian eigenvalue spectrum, instead of isolated eigenvalues, is of practical interest. In contrast to existing techniques, we propose a framework where we control the whole Laplacian eigenvalue spectrum of a network using the so-called spectral moments [15] by iteratively modifying the structure of the network, while ensuring preservation of network connectivity that is necessary for distributed coordination [16], [17]. We show that the benefits of controlling the spectral moments, instead of individual eigenvalues, lies in lower computational cost and efficient distributed implementation. Our proposed approach is greedy in nature, stable by construction, i.e., it locally optimizes the network's eigenvalue spectrum, and is shown to perform well in numerical simulations.

The rest of this paper is organized as follows. In Section II, we review terminology and formulate the problem under consideration. In Section III, we introduce a decentralized algorithm to compute the spectral moments of the Laplacian matrix from myopic views of the network's structure. We also introduce a novel perturbation technique to efficiently compute the effect of adding or removing edges on the spectral moments. Based on these results, in Section IV, we develop a distributed algorithm in which a group of autonomous agents modify their network of interconnections to control the network's spectral moments to desired values while ensuring that network connectivity is preserved. Finally, in Section V, we illustrate our approach with several computer simulations.

This work is supported in part by the NSF awards CNS #1302222, CNS #1302284, and IIS #1447470.

Victor M. Preciado is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA (e-mail: preciado@seas.upenn.edu). Michael M. Zavlanos is with the Department of Mechanical Engineering and Materials Science, Duke University, Durham, NC 27708, USA (e-mail: michael.zavlanos@duke.edu).

II. Preliminaries & Problem Definition

A. Eigenvalues of Graphs and Spectral Moments

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where $\mathcal{V} = \{1, \ldots, n\}$ denotes a set of n nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes a set of e undirected edges. If $(i,j) \in \mathcal{E}$, we call nodes i and j *adjacent* (or first-neighbors), which we denote by $i \sim j$. We define the set of first-neighbors of a node i as $\mathcal{N}_i = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}$. The *degree* d_i of a vertex i is the number of nodes adjacent to it, i.e., $d_i = |\mathcal{N}_i|$.[1] An undirected graph is called *simple* if its edges are unweighted and it has no self-loops[2]. A graph is *weighted* if there is a real number associated with every edge. More formally, a weighted graph $\mathcal{H}$ can be defined as the triad $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges in $\mathcal{H}$, and $\mathcal{W} = \{w_{ij} \in \mathbb{R}, \text{ for all } (i,j) \in \mathcal{E}\}$ is the set of (possibly negative) weights.

Graphs can be algebraically represented via matrices. The *adjacency matrix* of a simple undirected graph $\mathcal{G}$, denoted by $A_{\mathcal{G}} = [a_{ij}]$, is an $n \times n$ symmetric matrix defined entry-wise as $a_{ij} = 1$ if nodes i and j are adjacent, and $a_{ij} = 0$ otherwise. Given a weighted, undirected graph $\mathcal{H}$, the weighted, symmetric adjacency matrix is defined by $W_{\mathcal{H}} = [w_{ij}]$, where w_{ij} is the weight associated to edge $(i,j) \in \mathcal{E}$ and $w_{ij} = 0$ if i is not adjacent to j. We define the *degree matrix* of a simple graph $\mathcal{G}$ as the diagonal matrix $D_{\mathcal{G}} = diag(d_i)$. We define the *Laplacian matrix* $L_{\mathcal{G}}$ (also known as combinatorial Laplacian, or Kirchhoff matrix) of a simple graph as $L_{\mathcal{G}} = D_{\mathcal{G}} - A_{\mathcal{G}}$. For simple undirected graphs, $L_{\mathcal{G}}$ is a symmetric, positive semidefinite matrix, which we denote by $L_{\mathcal{G}} \succeq 0$, [18]. Thus, $L_{\mathcal{G}}$ has a full set of n real and orthogonal eigenvectors with real nonnegative eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Furthermore, the trivial eigenvalue $\lambda_1 = 0$ of $L_{\mathcal{G}}$ always admits a corresponding eigenvector $v_1 = (1, 1, \ldots, 1)^T$. The algebraic multiplicity of the trivial eigenvalue is equal to the number of connected components in $\mathcal{G}$. The smallest and largest nontrivial eigenvalues of $L_{\mathcal{G}}$, λ_2 and λ_n, are called the spectral gap and spectral radius of $L_{\mathcal{G}}$, respectively.

Given an undirected (possibly weighted) graph $\mathcal{G}$, we denote its Laplacian spectrum by $\mathcal{S}(\mathcal{G}) = \{\lambda_1, \ldots, \lambda_n\}$, and define the k-th Laplacian spectral moment of $\mathcal{G}$ as, [18]:

$$m_k(\mathcal{G}) = \frac{1}{n} \sum_{i=1}^{n} \lambda_i^k. \tag{1}$$

The following theorem states that an eigenvalue spectrum is uniquely characterized by a finite sequence of moments:

Theorem 2.1: Consider two undirected (possibly weighted) graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with Laplacian eigenvalue spectra $\mathcal{S}(\mathcal{G}_1) = \{\lambda_1^{(1)} \leq \ldots \leq \lambda_n^{(1)}\}$ and $\mathcal{S}_2(\mathcal{G}_1) = \{\lambda_1^{(2)} \leq \ldots \leq \lambda_n^{(2)}\}$. Then, $\lambda_i^{(1)} = \lambda_i^{(2)}$ for all $1 \leq i \leq n$ if and only if $m_k(\mathcal{G}_1) = m_k(\mathcal{G}_2)$ for $1 \leq k \leq n-1$.

Proof: In the Appendix. ■

In the rest of this paper we will focus on the spectrum of the graph Laplacian matrix $L_{\mathcal{G}}$ and its spectral moments, which we denote by $m_k(\mathcal{G})$. In this case, Theorem 2.1, implies that the Laplacian spectral moment of a graph on n nodes is uniquely characterized by the sequence of $n-1$ spectral moments $\{m_k(\mathcal{G})\}_{k=1}^{n-1}$. It is worth remarking that two nonisomorphic[3] graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can present the same eigenvalue spectrum [19], in which case we say that $\mathcal{G}_1$ and $\mathcal{G}_2$ are isospectral. In other words, the eigenvalue spectrum of a graph is not enough to characterize its structure. On the other hand, as we shall show in Section III, there are many interesting connections between the structural features of a graph $\mathcal{G}$ and the spectral moments of its Laplacian matrix, $m_k(\mathcal{G})$.

B. Local Structural Properties of Graphs

In this section we define a collection of structural properties that are important in our derivations. A *walk* of length k from node i_1 to node i_{k+1} is an ordered sequence of nodes $(i_1, i_2, \ldots, i_{k+1})$ such that $i_j \sim i_{j+1}$ for $j = 1, 2, \ldots, k$. One says that the walk *touches* each of the nodes that comprises it. If $i_1 = i_{k+1}$, then the walk is closed. A closed walk with no repeated nodes (with the exception of the first and last nodes) is called a *cycle*. Given a walk $p = (i_1, i_2, \ldots, i_{k+1})$ in a weighted graph $\mathcal{H}$ with weighted adjacency matrix $W_{\mathcal{H}} = [w_{ij}]$, we define the weight of the walk as, $\omega(p) = w_{i_1 i_2} w_{i_2 i_3} \ldots w_{i_k i_{k+1}}$.

We now define the concept of a local neighborhood around a node. Let $\delta(i,j)$ denote the *distance* between two nodes i and j, i.e., the minimum length of a walk from i to j. By convention, we assume that $\delta(i,i) = 0$. We define the r-th order neighborhood $\mathcal{G}_{i,r} = (\mathcal{N}_{i,r}, \mathcal{E}_{i,r})$ around a node i as the subgraph $\mathcal{G}_{i,r} \subseteq \mathcal{G}$ with node-set $\mathcal{N}_{i,r} = \{j \in \mathcal{V} : \delta(i,j) \leq r\}$, and edge-set $\mathcal{E}_{i,r} = \{(v,w) \in \mathcal{E} \colon v, w \in \mathcal{N}_{i,r}\}$. Given a set of k nodes $\mathcal{K} \subseteq \mathcal{V}$, we define $\mathcal{G}_{\mathcal{K}}$ as the subgraph of $\mathcal{G}$ with node-set $\mathcal{V}(\mathcal{G}_{\mathcal{K}}) = \mathcal{K}$ and edge-set $\mathcal{E}(\mathcal{G}_{\mathcal{K}}) = \{(i,j) \in \mathcal{E} \colon i, j \in \mathcal{K}\}$. We define $L_{\mathcal{G}}(\mathcal{K})$ as the $k \times k$ *submatrix* of $L_{\mathcal{G}}$ formed by selecting the rows and columns of $L_{\mathcal{G}}$ indexed by $\mathcal{K}$. Compactly, we define the Laplacian submatrix as $L_{i,r} = L_{\mathcal{G}}(\mathcal{N}_{i,r})$.

We say that a structural measurement is local with a certain radius r if it can be computed from the set of local neighborhoods $\{\mathcal{G}_{i,r}, \ i = 1, \ldots, n\}$. For example, the degree sequence of $\mathcal{G}$ is a local structural measurement (with radius 1), since we can compute the degree of each node i from the neighborhood $\mathcal{N}_{i,1}$. In contrast, the eigenvalue spectrum of the Laplacian matrix is not a local property, since we cannot compute the eigenvalues unless we know the complete graph structure (although it is possible to extract useful relevant information from local properties [20], [21]). One of the main contributions of this paper is to propose a novel methodology to extract global information regarding the Laplacian eigenvalue spectrum from the set of local neighborhoods.

C. Spectral Metrics and Problem Definition

As discussed in Section I, our goal is to propose a distributed algorithm to control the eigenvalue spectrum of a multi-agent network, via its spectral moments, by iteratively adding/removing edges in the network; see Section II-B. For this, we define the following spectral distance between

[1] We define by $|\mathcal{X}|$ the cardinality of the discrete set $\mathcal{X}$.

[2] A self-loop is an edge of the type (i,i).

[3] Two simple graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with adjacency matrices $A_{\mathcal{G}_1}$ and $A_{\mathcal{G}_2}$ are *isomorphic* if there exists a permutation matrix P_n such that $A_{\mathcal{G}_1} = P_n A_{\mathcal{G}_2} P_n^T$.

two graphs $\mathcal{G}_a$ and $\mathcal{G}_b$, with spectra $\mathcal{S}_a = \{\lambda_i^{(a)}\}_{i=1}^n$ and $\mathcal{S}_b = \{\lambda_i^{(b)}\}_{i=1}^n$, as[4]

$$d_{n-1}(\mathcal{S}_a, \mathcal{S}_b) = \sum\nolimits_{k=1}^{n-1} \left(m_k(\mathcal{G}_a)^{1/k} - m_k(\mathcal{G}_b)^{1/k} \right)^2. \quad (2)$$

According to Theorem 2.1, two graphs are isospectral if their first $n-1$ spectral moments coincide; thus, d_{n-1} in (2) is in fact a distance function in the space of graph spectra. We further define the *spectral pseudometric*[5]:

$$d_K(\mathcal{S}_a, \mathcal{S}_b) = \sum\nolimits_{k=1}^{K} \left(m_k(\mathcal{G}_a)^{1/k} - m_k(\mathcal{G}_b)^{1/k} \right)^2, \quad (3)$$

for $K < n-1$. The benefit of using the spectral pseudodistance versus other spectral distances is due to the fact that, as we shall show in Section III, we can efficiently compute the first K spectral moments of the Laplacian matrix from the set of local Laplacian submatrices with radius $\lfloor K/2 \rfloor$, i.e., $\{L_{i,\lfloor K/2 \rfloor}, i \in \mathcal{V}\}$. In other words, assuming that each agent has access to the Laplacian submatrix associated with its neighborhood with radius r, we shall show how to distributedly compute the first $2r+1$ Laplacian moments of the complete graph $L_{\mathcal{G}}$. With the notation defined above, we can rigorously state the problem addressed in this paper as follows:

Problem 1: Given a desired spectrum $\mathcal{S}^* = \{\lambda_i^*\}_{i=1}^n$, find a simple undirected graph $\mathcal{G}^*$ such that its Laplacian eigenvalue spectrum, denoted by $\mathcal{S}(\mathcal{G}^*)$, minimizes $d_{n-1}(\mathcal{S}(\mathcal{G}^*), \mathcal{S}^*)$.

Finding a simple graph with a given, feasible,[6] eigenvalue spectrum is, in general, a hard combinatorial problem, even in a centralized setting. In this paper, we propose a distributed algorithm to find a graph with a spectrum "close to" $\mathcal{S}^*$ in the d_K pseudometric sense (instead of in the d_{n-1} spectral metric). The key idea is to let a group of agents located at the nodes of a network to coordinate in a distributed way and iteratively add/remove edges in the network in order to drive the network's global eigenvalue spectrum towards the desired spectrum $\mathcal{S}^*$.

Our solution to Problem 1 relies on considering only K spectral moments (i.e., substitute d_{n-1} for d_K in Problem 1) and defining the notion of *edit distance* $d_E(\mathcal{G}_a, \mathcal{G}_b)$ between two graphs $\mathcal{G}_a$ and $\mathcal{G}_b$, which is the minimum number of edge additions plus edge deletions to transform $\mathcal{G}_a$ into a graph that is isomorphic to $\mathcal{G}_b$. Then, the proposed distributed algorithm constructs a sequence of graphs $\{\mathcal{G}(t)\}_{t\geq 0}$, starting from any graph $\mathcal{G}_0$, such that

$$\begin{aligned} \mathcal{G}(t+1) \;=\; & \operatorname{argmin}_{\mathcal{G}} \quad d_K(\mathcal{S}(\mathcal{G}), \mathcal{S}^*) \\ & \text{s.t} \quad d_E(\mathcal{G}(t), \mathcal{G}) = 1, \;\; \lambda_2(\mathcal{G}) > 0. \quad (4) \end{aligned}$$

By construction of this sequence $\{\mathcal{G}(t)\}_{t\geq 0}$, the corresponding sequence of spectra $\{\mathcal{S}(\mathcal{G}(t))\}_{t\geq 0}$ approaches $\mathcal{S}^*$ as t grows. The constraint $d_E(\mathcal{G}(t), \mathcal{G}(t+1)) = 1$ enforces only single edge additions or deletions at each iteration, while the requirement $\lambda_2(\mathcal{G}(t)) > 0$ enforces graph connectivity at all times, which is necessary for distributed coordination between the agents, as discussed in Section IV. Note that the solution of problem (4) requires global knowledge of the network structure. In this paper, instead, we propose a computationally efficient, distributed algorithm in which agents in the network solve (4) using only their local, myopic views of the network structure. In particular, we propose a method that allows the agents to compute, in a distributed fashion, the effect of an edge addition/deletion on the first $2r+1$ Laplacian moments. We then integrate this process with a distributed algorithm to find the edge addition/deletion that minimizes the resulting value of the spectral pseudodistance to $\mathcal{S}^*$, while ensuring network connectivity. Before we discuss the implementation details of our proposed distributed method in Section IV, we first provide the theoretical foundation for our approach in Section III.

Remark 2.1 (Convergence): Several remarks are in order. First, note that it is not always possible to find a simple graph that exactly matches a given eigenvalue spectrum. Second, the spectral pseudometric $d_K(\mathcal{S}(\mathcal{G}), \mathcal{S}^*)$ may present multiple minima for a given $\mathcal{S}^*$. These minima could correspond, for example, to several isospectral graphs matching the desired spectrum $\mathcal{S}^*$ [19]. Therefore, iteration (4) may converge to different isospectral graphs depending on the initial condition $\mathcal{G}_0$. On the other hand, it was numerically shown in [22] that the ratio between the number of isospectral graphs and the number of possible graphs on n nodes goes to zero as n grows to infinity. Therefore, it is reasonable to expect that the structure of a large-scale network is almost surely characterized by its spectrum, although this claim remains a conjecture. Third, iteration (4) finds the most beneficial edge addition/deletion in each time step, hence, this greedy approach may get trapped in a local minimum. In practice, however, we observe that the spectra of these local minima are remarkably close to those of the desired spectrum.

III. Moment-Based Analysis of the Laplacian Matrix

In this section, we use tools from algebraic graph theory to compute the spectral moments of the Laplacian matrix of $\mathcal{G}$ when only the set of local Laplacian submatrices $\{L_{i,r},\, i \in \mathcal{V}\}$ is available. As a result of our analysis, we propose a decentralized algorithm to compute a truncated sequence of Laplacian spectral moments via a single distributed averaging. Furthermore, we also present an efficient approach to compute the effect of adding or deleting an edge in the Laplacian spectral moments of the graph. Particularly useful in our derivations will be the following result from algebraic graph theory:

Lemma 3.1 ([18]): Let $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ be a weighted graph with weighted adjacency matrix $W_{\mathcal{H}} = [w_{ij}]$. Then

$$[W_{\mathcal{H}}^k]_{ii} = \sum\nolimits_{p \in P_{i,k}(\mathcal{H})} \omega(p),$$

where $P_{i,k}(\mathcal{H})$ is the set of closed walks of length k starting and finishing at node i in the weighted graph $\mathcal{H}$.

[4]Note that d_{n-1} is a distance in the space of eigenvalue spectra, but not in the space of graphs, since we can find nonisomorphic graphs that are isospectral.

[5]A pseudometric is a generalization of distance in which two distinct points (in our case, two distinct spectra) can have zero distance.

[6]We say that an eigenvalue spectrum is feasible if there is a simple graph whose Laplacian matrix presents that spectrum.

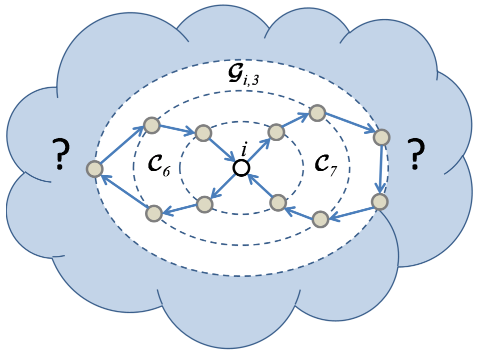


Fig. 1. Cycles $\mathcal{C}_6$ and $\mathcal{C}_7$, of lengths 6 and 7 , in a neighborhood of radius 3 around node i .

A. Algebraic Analysis of Structured Matrices

Consider the symmetric Laplacian matrix $L_{\mathcal{G}}$ of a simple graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. We denote by $\mathcal{G}_{i,r} = (\mathcal{N}_{i,r}, \mathcal{E}_{i,r})$ the neighborhood of radius r around node i and define the *local Laplacian submatrix* $L_{i,r}$, as the submatrix $L_{\mathcal{G}}(\mathcal{N}_{i,r})$, formed by selecting the rows and columns of $L_{\mathcal{G}}$ indexed by the set of nodes $\mathcal{N}_{i,r}$. By convention, we associate the first row and column of the submatrix $L_{i,r}$ with node $i \in \mathcal{V}$, which can be done via a simple permutation of rows and columns.[7] For a simple graph $\mathcal{G}$ with Laplacian matrix $L_{\mathcal{G}}$, we define $\mathcal{L}(\mathcal{G})$ as the weighted graph whose adjacency matrix is equal to $L_{\mathcal{G}}$. In other words, $\mathcal{L}(\mathcal{G})$ has edges with weight -1 for $(i,j) \in \mathcal{E}(\mathcal{G})$, 0 for $(i,j) \notin \mathcal{E}(\mathcal{G})$, and d_i for all self-loops (i,i), $i \in \mathcal{V}(\mathcal{G})$. We also define $\mathcal{H}_{i,r}$ as the weighted subgraph of $\mathcal{L}(\mathcal{G})$ with node set $\mathcal{N}_{i,r}$, containing all the edges of $\mathcal{L}(\mathcal{G})$ connecting pairs of nodes in $\mathcal{N}_{i,r}$ (including self-loops). Notice that, according to this definition, the weighted adjacency matrix of $\mathcal{H}_{i,r}$ is equal to $L_{i,r}$.

In this paper, we assume that each agent in the network knows the structure of its local neighborhood $\mathcal{G}_{i,r}$, for a fixed r. Therefore, agent i has access to the local Laplacian submatrix $L_{i,r}$. The following results allow us aggregate information from the set of local Laplacian submatrices, $\{L_{i,r}\}_{i\in\mathcal{V}}$, to compute a sequence of spectral moments of the (global) Laplacian matrix $L_{\mathcal{G}}$.

Theorem 3.2: Consider a simple graph $\mathcal{G}$ with Laplacian matrix $L_{\mathcal{G}}$. Then, for a given radius r, the Laplacian spectral moments can be written as

$$m_k(\mathcal{G}) = \frac{1}{n}\sum\nolimits_{i=1}^{n} \left[L_{i,r}^k\right]_{11}, \tag{5}$$

for $k \leq 2r+1$.

Proof: Since the trace of a matrix is the sum of its eigenvalues, we can expand the k-th spectral moment of the Laplacian matrix as follows:

$$m_k(\mathcal{G}) = \frac{1}{n}\mathrm{Tr}\left(L_{\mathcal{G}}^k\right) = \frac{1}{n}\sum\nolimits_{i=1}^{n}\left[L_{\mathcal{G}}^k\right]_{ii}$$

Therefore, since $L_{\mathcal{G}}$ is the weighted adjacency matrix of the Laplacian graph $\mathcal{L}(\mathcal{G})$, we have (from Lemma 3.1)

$$m_k(\mathcal{G}) = \frac{1}{n}\sum\nolimits_{i=1}^{n}\sum\nolimits_{p\in P_{i,k}(\mathcal{L}(\mathcal{G}))}\omega(p), \tag{6}$$

[7]Notice that permuting the rows and columns of the Laplacian matrix does not change the topology of the underlying graph.

Algorithm 1 Decentralized moment computation

```
Require: Local Laplacian submatrices $L_{i,r}$ for all nodes $i \in \mathcal{V}$;
1: For every node $i \in \mathcal{V}$ initialize a vector $\mu_i(0) = [\mu_{i,1}(0), \mu_{i,2}(0), \mu_{i,3}(0), \ldots, \mu_{i,2r+1}(0)]^T$, where $\mu_{i,k}(0) = [L_{i,r}^k]_{1,1}(0)$, as well as an iteration index $l = 0$;
2: while {not converged} do
3:     For all $i \in \mathcal{V}$, update the variables $\mu_i$ by the average consensus rule:
           $\mu_i(l+1) = \frac{1}{|\mathcal{N}_{i,1}|+1}\sum_{j\in\mathcal{N}_{i,1}}\mu_j(l)$
4:     Update the iteration index $l := l+1$;
5: end while
6: Set $[m_1(L_{\mathcal{G}}), m_2(L_{\mathcal{G}}), \ldots, m_{2r+1}(L_{\mathcal{G}})]^T := \mu_i^*$, where $\mu_i^* \to \frac{1}{n}\sum_{i=1}^{n}\mu_i(0)$ is the limit of the average consensus update for agent $i$ that converges to the desired average values.
```

where the weights $\omega(p)$ are summed over the set of closed walks of length k starting at node i in the weighted graph $\mathcal{L}(\mathcal{G})$.

For a fixed value of k, closed walks of length k in $\mathcal{L}(\mathcal{G})$ starting at node i can only touch nodes within a certain distance $r(k)$ of i, where $r(k)$ is a function of k (see Fig. 1). In particular, for k even (resp. odd), a closed walk of length k starting at node i can only touch nodes at most $k/2$ (resp. $\lfloor k/2 \rfloor$) hops away from i. Therefore, closed walks of length k starting at i are always contained within the neighborhood of radius $\lfloor k/2 \rfloor$. In other words, the neighborhood $\mathcal{G}_{i,r}$ of radius r contains all closed walks of length up to $2r+1$ starting at node i. Therefore, for $k \leq 2r+1$, we have that

$$\sum\nolimits_{p\in P_{i,k}(\mathcal{L})}\omega(p) = \sum\nolimits_{p\in P_{1,k}(\mathcal{H}_{i,r})}\omega(p),$$

where $\mathcal{H}_{i,r}$ is the weighted graph whose adjacency matrix is equal to the local Laplacian submatrix $L_{i,r}$ (notice that, by convention, we associate the first row and column of $L_{i,r}$ with node i). Therefore, according to Lemma 3.1, we have

$$\sum\nolimits_{p\in P_{1,k}(\mathcal{H}(L_{i,r}))}\omega(p) = \left[L_{i,r}^k\right]_{1,1}. \tag{7}$$

Then, substituting (7) into (6), we obtain the statement of our Theorem. ■

Since every node i has access to its local neighborhood $\mathcal{G}_{i,r}$, Theorem 3.2 implies that it is possible to compute the first $2r+1$ spectral moments of $L_{\mathcal{G}}$ from the local submatrices $L_{i,r}$, by developing a simple distributed averaging of the quantities $\{[L_{i,r}^k]_{1,1}\}_{i\in\mathcal{V}}$ [5]. This process is summarized in Algorithm 1. Algorithm 1 efficiently aggregates local pieces of local structural information (described by the local Laplacian submatrices) to produce a truncated sequence of spectral moments of the (global) Laplacian matrix. Note that, computing the spectral moments via (5) is much more efficient than computing these moments via an explicit eigenvalue decomposition for many real-world networks. In most real applications, the Laplacian matrix representing the network structure is a sparse graph for which the number of nodes in the neighborhood $\mathcal{N}_{i,r}$ is very small compared to n, for moderate values of r. On the other hand, in the case of small-world networks (and other networks of large expansion [23]), the neighborhoods grow exponentially fast with the radius. In this case, we should limit our analysis to small radii in order to avoid the computational burden of handling neighborhoods of very large sizes. As we will numerically illustrate in Section V, for networks of large expansion, it is enough to consider neighborhoods of radius 1 or 2 to extract rich spectral information about the whole graph (even though they do not uniquely characterize the spectrum).

B. Moment-Based Perturbation Analysis

In this section, we use spectral graph theory to compute the effect of adding or deleting an edge on the spectral moments of the Laplacian matrix. Traditionally, the effect of a matrix perturbation on the eigenvalue spectrum is analyzed using eigenvalue perturbation techniques [24]. In particular, the effect of adding a "small" perturbation matrix δW to an $n \times n$ symmetric matrix W with eigenvalue spectrum $\{\sigma_k\}_{k=1}^n$ can be approximated, in the first-order, by [24]

$$\widetilde{\sigma}_k - \sigma_k \approx u_k^T \ \delta W \ u_k,$$

where u_k is the eigenvector of W associated with the eigenvalue σ_k, and $\{\widetilde{\sigma}_k\}_{k=1}^n$ is the eigenvalue spectrum of the perturbed matrix $W + \delta W$. In the case of the Laplacian matrix, the perturbation matrix δW corresponding to the addition of an edge (i,j) can be written as, $\delta W = (e_i - e_j)(e_i - e_j)^T$, where e_i is the unit vector in the direction of the i-th coordinate. We denote by $\mathcal{G} + (i,j)$ the graph resulting from adding edge (i,j) to $\mathcal{G}$, and $\{\widetilde{\lambda}_k\}_{k=1}^n$ is the Laplacian spectrum of $\mathcal{G} + (i,j)$. Therefore, adding edge (i,j) perturbs the eigenvalues of the Laplacian matrix as follows:

$$\widetilde{\lambda}_k - \lambda_k \approx v_k^T (e_i - e_j)(e_i - e_j)^T v_k = (v_{k,i} - v_{k,j})^2,$$

where v_k is the eigenvector of $L_{\mathcal{G}}$ associated to λ_k, and $v_{k,j}$ is its j-th component. Hence, the resulting spectral radius can be approximated as

$$\widetilde{\lambda}_n \approx \lambda_n + (v_{n,i} - v_{n,j})^2 .$$

Therefore, computing the effect of an edge addition on the spectral radius using traditional perturbation techniques requires computation of the dominant eigenvalue and eigenvector of $L_{\mathcal{G}}$, which is computationally expensive for very large graphs. As an alternative to the traditional analysis, we propose a novel approach, based on algebraic graph theory, to compute the effect of structural perturbation on the spectral moments of the Laplacian matrix $L_{\mathcal{G}}$ *without explicitly computing the eigenvalues or eigenvectors of* $L_{\mathcal{G}}$. Furthermore, our approach can be efficiently implemented in a fully decentralized manner.

In our derivations, we use the following result from algebraic graph theory:

Lemma 3.3: Let $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ be a weighted graph with weighted adjacency matrix $W_{\mathcal{H}} = [w_{ij}]$. Then

$$m_k(W_{\mathcal{H}}) = \frac{1}{n} \sum\nolimits_{p \in P_k(\mathcal{H})} \omega(p), \tag{8}$$

where $P_k(\mathcal{H})$ is the set of closed walks of length k in the weighted graph $\mathcal{H}$.

Proof: This lemma is a consequence of Lemma 3.1. Specifically, we have that

$$\begin{aligned} m_k(W(\mathcal{H})) &= \frac{1}{n}\mathrm{Tr}(W(\mathcal{H})^k) = \frac{1}{n}\sum\nolimits_{i \in \mathcal{V}} [W(\mathcal{H})^k]_{i,i} \\ &= \frac{1}{n}\sum_{i \in \mathcal{V}} \sum_{p \in P_{i,k}(\mathcal{H})} \omega(p) = \frac{1}{n} \sum_{p \in P_k(\mathcal{H})} \omega(p), \end{aligned}$$

where $P_k(\mathcal{H}) = \cup_{i \in \mathcal{V}} P_{i,k}(\mathcal{H})$ is the set of all closed walks of length k in $\mathcal{H}$ (for any starting node $i \in \mathcal{V}$). ■

C. Perturbation on the Spectral Moments

Consider a simple graph $\mathcal{G}$ with Laplacian matrix $L_{\mathcal{G}}$. We denote by $\mathcal{G} + (i,j)$ (resp. $\mathcal{G} - (i,j)$) the graph resulting from adding (resp. removing) edge (i,j) to (resp. from) $\mathcal{G}$. Consider the sets of nodes $\mathcal{N}_{i,r}$ and $\mathcal{N}_{j,r}$ being within a radius r from node i and node j, respectively. Let us define the following submatrices indexed by the set of nodes in $\mathcal{N}_{i,r} \cup \mathcal{N}_{j,r}$:

$$\begin{aligned} U_{r,(i,j)} &= L_{\mathcal{G}}(\mathcal{N}_{i,r} \cup \mathcal{N}_{j,r}), \\ U^+_{r,(i,j)} &= L_{\mathcal{G}+(i,j)}(\mathcal{N}_{i,r} \cup \mathcal{N}_{j,r}), \\ U^-_{r,(i,j)} &= L_{\mathcal{G}-(i,j)}(\mathcal{N}_{i,r} \cup \mathcal{N}_{j,r}). \end{aligned}$$

The following lemma allows us to efficiently compute the increment (resp. decrement) in the Laplacian spectral moments of $\mathcal{G}$ due to the addition (resp. removal) of edge (i,j):

Theorem 3.4: Given a simple graph $\mathcal{G}$ with Laplacian matrix $L_{\mathcal{G}}$, the increment (decrement) in the k-th Laplacian spectral moment of a graph $\mathcal{G}$ due to the addition or deletion of an edge (i,j) can be written as

$$m_k(\mathcal{G} \pm (i,j)) - m_k(\mathcal{G}) = \frac{1}{n}\Big(\mathrm{Tr}(U^{\pm}_{r,(i,j)})^k - \mathrm{Tr}(U_{r,(i,j)})^k\Big) \tag{9}$$

for $k \le 2r + 1$.

Proof: Consider the weighted Laplacian graphs of $L_{\mathcal{G}}$, $L_{\mathcal{G}+(i,j)}$, and $L_{\mathcal{G}-(i,j)}$, which we denote by $\mathcal{H} = \mathcal{L}(\mathcal{G})$, $\mathcal{H}^+ = \mathcal{L}(\mathcal{G} + (i,j))$ and $\mathcal{H}^- = \mathcal{L}(\mathcal{G} - (i,j))$, respectively. (By definition, the adjacency matrices of the Laplacian graphs are the Laplacian matrices of the graphs.) Then, according to Lemma 3.3, we have that the k-th spectral moments $m_k(\mathcal{G})$, $m_k(\mathcal{G} + (i,j))$ and $m_k(\mathcal{G} - (i,j))$ can be written as weighted sums over the sets of all closed walks of length k in $\mathcal{H}$, $\mathcal{H}^+$, and $\mathcal{H}^-$, as follows,

$$\begin{aligned} m_k(\mathcal{G}) &= \frac{1}{n}\sum\nolimits_{p \in P_k(\mathcal{H})} \omega(p), \\ m_k(\mathcal{G} \pm (i,j)) &= \frac{1}{n}\sum\nolimits_{p \in P_k(\mathcal{H}^{\pm})} \omega(p). \end{aligned}$$

We define $P^{(i,j)}_{k,r}(\mathcal{H})$, $P^{(i,j)}_{k,r}(\mathcal{H}^+)$, and $P^{(i,j)}_{k,r}(\mathcal{H}^-)$ as the sets of closed walks of length k in, respectively, $\mathcal{H}$, $\mathcal{H}^+$, and $\mathcal{H}^-$ visiting only nodes in the set $\mathcal{N}_{i,r} \cup \mathcal{N}_{j,r}$. Then, we can split the summation in (8) for the Laplacian matrices, as follows:

$$m_k(\mathcal{G}) = \frac{1}{n}\sum_{p\in P_{k,r}^{(i,j)}(\mathcal{H})}\omega(p) + \frac{1}{n}\sum_{p\in P_k\backslash P_{k,r}^{(i,j)}(\mathcal{H})}\omega(p), \qquad (10)$$

$$m_k(\mathcal{G}\pm(i,j)) = \frac{1}{n}\sum_{p\in P_{k,r}^{(i,j)}(\mathcal{H}^{\pm})}\omega(p) + \frac{1}{n}\sum_{p\in P_k\backslash P_{k,r}^{(i,j)}(\mathcal{H}^{\pm})}\omega(p). \qquad (11)$$

Notice that, as we illustrated in Fig. 1, none of the closed walk of length $k \le 2r+1$ touching node i (resp. node j) can leave the neighborhood $\mathcal{N}_{i,r}$ (resp. $\mathcal{N}_{j,r}$). Therefore, all closed walks of length $k \le 2r+1$ touching either node i or j (or both) are contained[8] in $\mathcal{N}_{i,r}\cup\mathcal{N}_{j,r}$. As a consequence, none of the closed walks in $P_k\backslash P_k^{(i,j)}(\mathcal{H})$ or $P_k\backslash P_k^{(i,j)}(\mathcal{H}^{\pm})$ touches node i or j. Since addition/removal of edge (i,j) does not influence those walks not touching i or j, we have that

$$\sum_{p\in P_k\backslash P_{k,r}^{(i,j)}(\mathcal{H})}\omega(p) = \frac{1}{n}\sum_{p\in P_k\backslash P_{k,r}^{(i,j)}(\mathcal{H}^{\pm})}\omega(p).$$

Thus, from (10) and (11) we have

$$m_k\left(\mathcal{G}^{\pm}(i,j)\right) - m_k(\mathcal{G}) = \frac{1}{n}\sum_{p\in P_{k,r}^{(i,j)}(\mathcal{H}^{\pm})}\omega(p) - \frac{1}{n}\sum_{p\in P_{k,r}^{(i,j)}(\mathcal{H})}\omega(p). \qquad (12)$$

Since $P_{k,r}^{(i,j)}(\mathcal{H})$ is the set of all closed walks of length k in $\mathcal{H}$ visiting nodes in the set $\mathcal{N}_{i,r}\cup\mathcal{N}_{j,r}$, we can apply Lemma 3.3 to obtain

$$\frac{1}{n}\sum_{p\in P_{k,r}^{(i,j)}(\mathcal{H})}\omega(p) = m_k\left(\mathcal{G}\left(\mathcal{N}_{i,r}\cup\mathcal{N}_{j,r}\right)\right) = \frac{1}{n}\mathrm{Tr}\left(U_{r,(i,j)}\right)^k. \qquad (13)$$

Similarly, for $P_{k,r}^{(i,j)}\left(\mathcal{H}^{\pm}\right)$, we obtain

$$\frac{1}{n}\sum\nolimits_{p\in P_{k,r}^{(i,j)}(\mathcal{H}^{\pm})}\omega(p) = \frac{1}{n}\mathrm{Tr}(U_{r,(i,j)}^{\pm})^k. \qquad (14)$$

Finally, substituting (13) and (14) in (12) provides us with the statement of our theorem. ■

Remark 3.1 (Computational cost): According to Theorem 3.4, we can compute the increment or decrement in the Laplacian spectral moments (up to order $2r+1$) by computing $\mathrm{Tr}(U_{r,(i,j)})^k$ and $\mathrm{Tr}(U_{r,(i,j)}^{\pm})^k$.

IV. Decentralized Control of Spectral Moments

In this section, we integrate the results developed in Section III with a novel technique for distributed connectivity verification of edge additions or deletions in order to obtain a distributed solution to Problem 1 in the form of (4), as discussed in Section II-C. This relies on the assumption that an agent at node i is able to communicate at time slot t with all the agents in its first-order neighborhood $\mathcal{N}_{i,1}(t)$ only.[9] Moreover, we also assume that every agent has only a myopic view of the network structure. This means that at time slot t agent $i\in\mathcal{V}$ only knows the topology of the neighborhood $\mathcal{G}_{i,r}(t)$, within a particular radius r. This limits the set of possible actions that every agent i can take in every step of the iteration (4), to be *local* edge additions of non-edges[10] $(i,j)\notin\mathcal{E}(t)$ in $\mathcal{G}_{i,r}(t)$ or local edge deletions of edges $(i,j)\in\mathcal{E}(t)$ in $\mathcal{G}_{i,1}(t)$.

In what follows, it will be useful to predetermine the *master node* for each edge $(i,j)\in\mathcal{E}(t)$, which can be arbitrarily chosen from the set of nodes $\{i,j\}$. The notion of master node is useful to coordinate actions in our decentralized algorithm. The agent located at the master node of (i,j) is the only one with the authority to decide if edge (i,j) is deleted. We denote by $\mathcal{D}_i(t)$ the set of edges having node i as its master. In this paper, we choose this set to be $\mathcal{D}_i(t) = \{(i,j)\in\mathcal{E}(t) \mid i>j\}$.[11] Similarly, it is useful to predefine a master node for each non-edge $(i,k)\notin\mathcal{E}(t)$. The agent located at the master node of the non-edge is the only one with the authority to decide if edge (i,k) is added to the network. We denote by $\mathcal{A}_i(t)$ the set of non-edges having node i as its master. In our case, we define this set as $\mathcal{A}_i(t) = \{(i,k)\notin\mathcal{E}(t) \mid k\in\mathcal{N}_{i,r}(t) \text{ and } i>k\}$, where we limit node k to be in $\mathcal{N}_{i,r}(t)$, since we are only considering local edge additions.

A. Connectivity-Preserving Edge Deletions

In a centralized framework, network connectivity can be inferred from the number of trivial eigenvalues of the Laplacian matrix. However, when only local network information is available, only sufficient conditions for connectivity can be verified. One such condition is the following: if $\mathcal{G}$ is connected, $(i,j)\in\mathcal{E}$, and $|\mathcal{N}_{j,1}\cap\mathcal{N}_{i,r}|>2$, then the graph $\mathcal{G}-(i,j)$ is connected. If the conditions of this claim are satisfied, it is easy to prove that, besides the direct link (i,j), there is at least one more path connecting nodes i and j in $\mathcal{G}$. Therefore, the link (i,j) can be safely deleted to preserve network connectivity. Since this condition is only sufficient but not necessary for connectivity preservation, we need a mechanism to check connectivity for those edges in the set

$$\mathcal{C}(t) = \{(i,j)\in\mathcal{E}(t) \ : \ |\mathcal{N}_{j,1}(t)\cap\mathcal{N}_{i,r}(t)| = 1\}.$$

of critically connected edges, for which the sufficient condition does not hold.

The proposed mechanism relies on the concept of a maximum consensus. In particular, consider a graph $\mathcal{G}(t) = (\mathcal{V},\mathcal{E}(t))$ at time $t\ge 0$ and let $(i,j)\in\mathcal{C}(t)$ be a critically connected edge that needs to be verified as to whether its deletion can violate connectivity. Associate, further, a randomly initialized scalar variable $x_k(s)\in\mathbb{R}$ with every node $k\in\mathcal{V}$ and run the following maximum consensus update

$$x_k(s+1) = \max_{l\in\mathcal{N}_{k,1}-\{i,j\}}\{x_l(s)\} \qquad (15)$$

on the graph $\mathcal{G}(t)-(i,j)$ obtained by virtually disabling the link (i,j) via blocking communication through it. Then, the network $\mathcal{G}(t)-(i,j)$ is almost surely connected if and only if the variables $x_k(s)$ for all $k\in\mathcal{V}$ converge to the common

[8] We say that a walk is *contained* in a set of nodes N if it only touches nodes in N.

[9] Notice that, since $\mathcal{G}(t)$ is time-dependent, so are the neighborhoods $\mathcal{G}_{i,r}(t) = (\mathcal{N}_{i,r}(t), \mathcal{E}_{i,r}(t))$.

[10] A pair of nodes (i,k) is a non-edge of $\mathcal{G}$ if $(i,k)\notin\mathcal{E}(\mathcal{G})$.

[11] Since the indices of all nodes in the network are distinct natural numbers, this definition results in a unique assignment.

value $\max_k x_k(0)$. Note that convergence in this case takes place in finite time that is upper bounded by the diameter of the network [25].

This idea can be extended to simultaneous verification of multiple link deletions in $\mathcal{C}(t)$. For this, every node k in the network needs to participate in the verification of all links in $\mathcal{C}(t)$ that need to be tested as to whether their deletion can violate connectivity. As before, this can be achieved by means of a maximum consensus algorithm, similar to (15), where now every node updates a vector $\mathbf{x}_k(s) \in \mathbb{R}^{|\mathcal{C}(t)|}$, each entry of which corresponds to a distinct edge in $\mathcal{C}(t)$ that need to be verified with respect to connectivity. To develop the corresponding maximum consensus algorithm, we need to define the virtual networks over which the entries of $\mathbf{x}_k(s)$ are updated. Note that these virtual networks are not the same for all entries in $\mathbf{x}_k(s)$, as each such entry corresponds to a different edge in $\mathcal{C}(t)$ that needs to be virtually disabled during execution of the associated maximum consensus update.

Since every edge is assigned a unique master agent, we can partition the set $\mathcal{C}(t)$ in to $|\mathcal{V}|$ disjoint subsets $\mathcal{C}(t) \cap \mathcal{D}_i(t)$ for all $i \in \mathcal{V}$, so that the set $\mathcal{C}(t) \cap \mathcal{D}_i(t)$ contains all critically connected edges that have as a master agent i. This allows us to decompose the vector $\mathbf{x}_k(s)$ for every node $k \in \mathcal{V}$ as $\mathbf{x}_k(s) = [\mathbf{x}_{k,1}^T(s) \dots \mathbf{x}_{k,n}^T(s)]^T$, where $\mathbf{x}_{k,i}(s) \in \mathbb{R}^{|\mathcal{C}(t) \cap \mathcal{D}_i(t)|}$ is a vector whose entries $[\mathbf{x}_{k,i}(s)]_{(i,j)} \in \mathbb{R}$ correspond to edges $(i,j) \in \mathcal{C}(t) \cap \mathcal{D}_i(t)$ for which agent i is the master. Using this notation, we can simultaneously verify connectivity for all edges in $\mathcal{C}(t)$ by a high-dimensional consensus. For this, every agent k initializes randomly all vectors $\mathbf{x}_{k,i}(s) \in \mathbb{R}^{|\mathcal{C}(t) \cap \mathcal{D}_i(t)|}$ for all masters $i \in \mathcal{V}$ and updates their values as follows:
Case I: If k is not a neighbor of the master agent i, i.e., if $k \notin \mathcal{N}_i$, then agent k updates the vectors $\mathbf{x}_{k,i}(s)$ as

$$\mathbf{x}_{k,i}(s+1) := \max_{l \in \mathcal{N}_k(t)} \{\mathbf{x}_{k,i}(s), \mathbf{x}_{l,i}(s)\}, \tag{16}$$

where the maximum is applied elementwise on the vectors.
Case II: If k is a neighbor of the master agent i, i.e., if $k \in \mathcal{N}_i$, then

(a) if $(k,i) \in \mathcal{C}(t) \cap \mathcal{D}_i(t)$, then agent k virtually removes link (k,i) and updates the entry $[\mathbf{x}_{k,i}(s)]_{(k,i)}$ as

$$[\mathbf{x}_{k,i}(s+1)]_{(k,i)} := \max_{l \in \mathcal{N}_k(t) \setminus \{i\}} \left\{[\mathbf{x}_{k,i}(s)]_{(k,i)}, [\mathbf{x}_{l,i}(s)]_{(k,i)}\right\}, \tag{17}$$

(b) for all other links $(j,i) \in \mathcal{C}(t) \cap \mathcal{D}_i(t)$ with $j \neq k$ agent k updates the entries $[\mathbf{x}_{ki}(s)]_{(j,i)}$ same as in (16), i.e.,

$$[\mathbf{x}_{k,i}(s+1)]_{(j,i)} := \max_{l \in \mathcal{N}_k(t)} \left\{[\mathbf{x}_{k,i}(s)]_{(j,i)}, [\mathbf{x}_{l,i}(s)]_{(j,i)}\right\}. \tag{18}$$

Case III: For all variables in $\mathbf{x}_{k,k}(s)$ for which k is the master, agent k virtually removes every link $(k,j) \in \mathcal{C}(t) \cap \mathcal{D}_k(t)$ and updates the corresponding entriy $[\mathbf{x}_{k,k}(s)]_{(k,j)}$ as

$$[\mathbf{x}_{k,k}(s+1)]_{(k,j)} := \max_{l \in \mathcal{N}_k(t) \setminus \{j\}} \left\{[\mathbf{x}_{k,k}(s)]_{(k,j)}, [\mathbf{x}_{l,k}(s)]_{(k,j)}\right\}. \tag{19}$$

The high-dimensional consensus defined by (16)–(19) converges in a finite time $\tau > 0$ [25]. When this happens, every master node k requests the entries $[\mathbf{x}_{i,k}(\tau)]_{(k,i)}$ from all its neighbors $i \in \mathcal{N}_k(t)$ for which $(k,i) \in \mathcal{C}(t) \cap \mathcal{D}_k(t)$ and compares them with $[\mathbf{x}_{k,k}(\tau)]_{(k,i)}$. Since, violation of connectivity due to deletion of (k,i) would result in nodes k and i being in different connected components, if $[\mathbf{x}_{k,k}(s)]_{(k,i)} = [\mathbf{x}_{i,k}(s)]_{(k,i)}$ then the network $\mathcal{G}(t) - (k,i)$ would still remain connected. Hence, we can define the set

$$\mathcal{R}_k(t) = \left\{(k,i) \in \mathcal{C}(t) \cap \mathcal{D}_k(t) : [\mathbf{x}_{k,k}(\tau)]_{(k,i)} = [\mathbf{x}_{i,k}(\tau)]_{(k,i)}\right\}, \tag{20}$$

containing the edges in $\mathcal{C}(t) \cap \mathcal{D}_k(t)$ whose removal does not disconnect the network.

Algorithm 2 Connectivity verification
```
Require: $\mathbf{x}_{i,j}(0) \in \mathbb{R}^{|\mathcal{C}(t) \cap \mathcal{D}_j(t)|}$ for all $i,j \in \mathcal{V}$ ;
1: for $s = 1 : \tau$ do
2:     Update $\mathbf{x}_{i,j}(s+1)$ by (16)–(19);
3: end for
4: Compute $\mathcal{R}_i(t)$ by (20);
```

B. Greedy Local Action

To solve Problem 1 via the iterative algorithm proposed in (4), we need to add or delete an edge (i,j) that minimizes the spectral pseudometric $d_K(\mathcal{S}(\mathcal{G}_{\pm(i,j)}(t)), \mathcal{S}^*)$ at every time step t. For this, let $\Delta_i(t) = d_K(\mathcal{S}(\mathcal{G}(t)), \mathcal{S}^*)$ denote a local copy of the spectral distance of the graph $\mathcal{G}(t)$ that is available to agent i, so that initially $\Delta_i(0) = \Delta(0)$ for all agents $i \in \mathcal{V}$. The quantity $\Delta(0)$ can be computed in a distributed way by means of distributed averaging, according to Theorem 3.2. Then, the key idea is that every master agent i computes the spectral distance $\Delta_{\pm(i,j)}(s) = d_K\left(\mathcal{S}\left(\mathcal{G}_{\pm(i,j)}(s)\right), \mathcal{S}^*\right)$ resulting from adding a link $(i,j) \in \mathcal{A}_i(t)$ or removing a link $(i,j) \in \mathcal{R}_i(t)$. Computation of this distance relies on Theorem 3.4 and requires that agent i has knowledge of the structure of its neighborhoods $\mathcal{G}_{i,r}$ only, for $r = \lfloor K/2 \rfloor$. For all possible local edge additions or deletions, master agent i determines the most beneficial one

$$(i, j_i^*(t)) = \underset{(i,j) \in \mathcal{A}_i(t) \cup \mathcal{R}_i(t)}{\operatorname{argmin}} \left\{\Delta_{\pm(i,j)}(t) - \Delta_i(t)\right\}.$$

Note that the minimization above may result in multiple edges having the same optimal value. Such ties can be broken via, e.g., a coin toss. Then, the largest decrease in the error associated with the most beneficial edge $(i, j_i^*(t))$ becomes:

$$\Delta_i(t) = \begin{cases} \Delta_{\pm(i,j_i^*)}(t), & \text{if } \min\left\{\Delta_{\pm(i,j)}(t) - \Delta_i(t),\right. \\ & \left.\text{s.t. } (i,j) \in \mathcal{A}_i(t) \cup \mathcal{R}_i(t)\right\} \leq 0 \\ D, & \text{otherwise} \end{cases}$$

for a large constant $D > 0$. In other words, $\Delta_i(t)$ is nontrivially defined only if there exists a link adjacent to node i that if added or deleted decreases the error function $\Delta(t)$. Otherwise, a large value $D > 0$ is assigned to $\Delta_i(t)$ to indicate that this action is not beneficial to the final objective. Finally, for each node i, we initialize the state vector

$$\mathbf{b}_i(0) = \left[i \; j_i^*(t) \; \Delta_i(t) \; \mathbf{m}(i, j_i^*(t))\right]^T,$$

containing the best local action $(i, j_i^*(t))$, the associated spectral pseudodistance $\Delta_i(t)$, and the vector of resulting moments

$$\mathbf{m}(i, j_i^*(t)) = \left[m_k\left(\mathcal{S}\left(\mathcal{G}_{\pm(i,j_i^*)}(t)\right)\right)\right]_{k=1}^K.$$

Algorithm 3 Globally most beneficial action

```
Require: $\mathbf{b}_i(0) = [i\ j_i^*(t)\ \Delta_i(t)\ \mathbf{m}(i, j_i^*(t))]^T$;
1: for $s = 1 : \tau$ do
2:     $\mathbf{b}_i(s+1) := \mathbf{b}_j(s)$, with $j = \max\{\text{argmin}_{k \in \mathcal{N}_i(t)}\{[\mathbf{b}_i(s)]_3, [\mathbf{b}_k(s)]_3\}$;
3: end for
4: if $[\mathbf{b}_i(\tau)]_3 < D$ then
5:     Update $\mathcal{N}_i(t+1)$, $\mathbf{m}_i(t+1)$ and $\Delta_i(t+1)$ according to (21)–(24);
6: else if $[\mathbf{b}_i(\tau)]_3 = D$ then
7:     No beneficial action. Algorithm has converged;
8: end if
```

In the following section, we discuss how to compare all local actions $\mathbf{b}_i(t)$ for all nodes $i \in \mathcal{V}$ to find the best global action that minimizes the spectral pseudometric.

C. Greedy Global Action

In order to obtain the overall most beneficial action, all local actions need to be propagated in the network and compared against each other. For this, every agent i communicates with its neighbors and updates its desired action $\mathbf{b}_i(s)$ with the action $\mathbf{b}_j(s)$ corresponding to the node j that contains the smallest distance to the target moments $[\mathbf{b}_j(s)]_3 = \Delta_i(t)$, i.e.,

$$\begin{aligned} \mathbf{b}_i(s+1) &= \mathbf{b}_j(s), \text{ where} \\ j &= \text{argmin}_{k \in \mathcal{N}_i(t)}\{[\mathbf{b}_i(s)]_3, [\mathbf{b}_k(s)]_3\}. \end{aligned}$$

In case of ties in the distances to the targets $[\mathbf{b}_j(s)]_3$, then the node with the largest index is selected (line 2, Alg. 3). Note that line 2 of Alg. 3 is essentially a minimum consensus update on the entries $[\mathbf{b}_i(s)]_3$ and will converge to a common outcome for all nodes in finite time $\tau > 0$, when they have all been compared to each other. When the consensus has converged, if there exists a node whose desired action decreases the distance to the target moments, i.e., if $[\mathbf{b}_i(s)]_3 < D$ (line 4, Alg. 3), then Alg. 3 terminates with a greedy action and node i updates its set of neighbors $\mathcal{N}_i(t+1)$ and vector of moments $\mathbf{m}_i(t+1)$ (line 5, Alg. 3). If the optimal action is a link addition, i.e., if $[\mathbf{b}_i(\tau)]_2 \notin \mathcal{N}_i(t)$, then

$$\mathcal{N}_i(t+1) := \mathcal{N}_i(t) \cup \{[\mathbf{b}_i(\tau)]_2\}. \tag{21}$$

On the other hand, if the optimal action is a link deletion, i.e., if $[\mathbf{b}_i(\tau)]_2 \in \mathcal{N}_i(t)$, then

$$\mathcal{N}_i(t+1) := \mathcal{N}_i(t) \backslash \{[\mathbf{b}_i(\tau)]_2\}. \tag{22}$$

In all cases, the moments and error function are updated by

$$\mathbf{m}_i(t+1) := [[\mathbf{b}_i(\tau)]_4 \ldots [\mathbf{b}_i(\tau)]_{4+K}]^T \tag{23}$$

and

$$\Delta_i(t+1) := [\mathbf{b}_i(\tau)]_3, \tag{24}$$

respectively. Finally, if all local desired actions increase the distance to the target moments, i.e., if $[\mathbf{b}_i(\tau)]_3 = D$ (line 6, Alg. 3), then no action is taken and the algorithm terminates with a network topology with almost the desired spectral properties. This is because no action exists that can further decrease the distance to the target moments.

To summarize, our algorithm performs the following steps:

1) *Initialization*: Construct an arbitrary connected graph with n nodes and no self-loops $\mathcal{G}_0$ (e.g., a connected Erdos-Renyi realization). Compute the spectral moments of $\mathcal{G}_0$ using Algorithm 1. Set $\mathcal{G}(0) = \mathcal{G}_0$ and $t = 1$.
2) *Iterations*: At iteration t, we run the following subroutines for each one of the $\binom{n}{2}$ pairs of nodes (i,j), $i < j$, in $\mathcal{G}(t)$:
    a) If $(i,j) \in \mathcal{E}(t)$, check that the removal of (i,j) does not disconnected the resulting graph using the subroutine in Algorithm 2.
    b) Use (9) to compute the perturbation on the first K Laplacian spectral moments of adding/removing edge (i,j).
    c) Find the most beneficial edge addition/removal (among those that do not disconnect the graph) using the subroutine in Algorithm 3.
    d) Set $t = t+1$ and generate $\mathcal{G}(t)$ from $\mathcal{G}(t-1)$ using the most beneficial action.
    e) Repeat this iteration until we reach a prescribed spectral pseudometric or reach a prescribed upper bound in the number of iterations.

It is worth remarking that this algorithm can be easily adapted to a centralized setting. In this case, the objective of the central designer is to find a graph matching a collection of given spectral moments. Notice that, in this setting, the graph does not need to be connected, since connectivity is only assumed to allow for the decentralized coordination of agents in the network. Therefore, in the centralized case, the implementation is identical to the one described above, but removing Step 2.a. As a result, the designed graph could be disconnected.

V. Numerical Simulations

In the following numerical examples, we illustrate the performance and limitations of our iterative graph process. The objective of our simulations is to find a graph whose Laplacian spectral moments match those of a desired spectrum. To measure the distance between the desired and the designed spectrum, $\mathcal{S}_a = \{\lambda_i^{(a)}\}_{i=1}^n$ and $\mathcal{S}_b = \{\lambda_i^{(b)}\}_{i=1}^n$, respectively, we define the *quadratic spectral distance* as

$$\delta_S(\mathcal{S}_a, \mathcal{S}_b) = \frac{1}{n}\sum_{i=1}^n \left(\lambda_i^{(a)} - \lambda_i^{(b)}\right)^2, \tag{25}$$

where the eigenvalues are indexed in increasing order. In the following examples, we show how our algorithm is able to efficiently generate graphs matching the spectral properties of three popular synthetic network models: the small-world [26], the scale-free [27], and random geometric networks [28].

A. Small-World Networks

The small-world model was proposed by Watts and Strogatz [26] to generate networks with high clustering[12] coefficients

[12]The clustering coefficient of a network is a measure of the number of triangles present in the network.

and small average distance. We can generate a small-world network by following these steps: (1) take a ring graph with n nodes, (2) connect each node in the ring to all its neighborhoods within a distance k, and (3) add random edges with a probability p. In this example, we generate a small-world network with $n = 40$, $k = 2$, and $p = 3/n$. The first three spectral moments of a random realization of this network are $(m_k)_{k=1}^{3} = (6.55, 51.9, 457)$. Then, we run our algorithm to generate a graph whose first three spectral moments are close to those of the small-world network. After running our algorithm for 78 iterations, we obtain a graph topology with a spectral pseudodistance very close to zero (in particular, $1.7e-3$) and an eigenvalue spectrum remarkably similar to that of the small-world network, as shown in Fig. 2.

Apart from this particular network realization, we illustrate the performance of our algorithm using several realizations for different values of r and n. In these simulations, we generate 5 random realizations of the small-world model when n is in the range $[25 : 25 : 100]$. For each realization, we run the spectral design algorithm when we consider 3, 4, and 5 spectral moments (i.e., $K \in \{3, 4, 5\}$). In Table I, we include a summary of our results. For example, in those columns corresponding to the small-world model, we observe how as we increase n from 25 to 100, the average diameter, denoted by $\overline{\phi}_a$, takes the values 2.6, 3, 3, and 3, while in the designed network (using $K = 5$) the average diameter, denoted by $\overline{\phi}_b^{(5)}$, takes the values 2.4, 3, 3.2, and 3.2.

We now shift our attention to the average quadratic spectral distance between the random realization of the model and the network designed using the spectral pseudometric. We denote by $\mathcal{S}_a$ the desired eigenvalue spectrum and by $\mathcal{S}_b^{(K)}$ the spectrum designed using K spectral moments. In Table I, we include the average quadratic spectral distance, denoted by $\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(K)})$, as we vary the size of the network in the range n=[25:25:100] and the number of moments used in the design in the range $K \in \{3, 4, 5\}$. From this table, we observe how as we increase the size of the network for a constant value of K, the spectral distance does not vary much. This indicates that the algorithm's performance is stable with respect to the network size. We also observe that, as we increase the number of moments K for a fixed network size, the performance does not tend to increase. This can be explained from the fact that the small-world model generates networks with a very small diameter, which makes neighborhoods of radius one very informative.

B. Power-Law Networks

Another popular model in the network science literature is the scale-free network. This model was proposed by Barabási and Albert in [27] to explain the presence of heavy-tailed degree distributions in many real-world networks. In this example, we generate a random power-law network with $n = 50$ nodes and $m = 4$, where m is a parameter that characterizes the average degree of the resulting network (see [27] for more details about this model). A random realization of this network presents the following sequence of moments: $(m_k)_{k=1}^{5} = (7.72, 111, 2.81e3, 9.70e4, 3.82e6)$. Then, after running our algorithm for 98 iterations, we obtain a graph topology with a spectral pseudodistance very close to zero (in particular, $5.5e-2$). The eigenvalue spectrum of the resulting topology is remarkably similar to that of the small-world network, as shown in Fig. 3.

As we did for the case of small-world networks, we illustrate the performance of our algorithm for different values of r and n in the range [25:25:100]. In the columns in Table I corresponding to the preferential attachment model, we observe how, as we increase n, the average diameter takes the values 3, 3.4, 4, and 4, while in the network designed using 5 moments the average diameter takes the values 2.6, 3.8, 3.6, and 4. In Table I, we also include the average spectral distance $\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(K)})$ as we vary the size of the network in the range n=[25:25:100] and the number of moments in the range $K \in \{3, 4, 5\}$. As we observed in the small-world model, as we increase the size of the network for a constant value of K, the quadratic spectral distance does not vary much. Also, as we increase the number of moments K for a fixed network size, the quadratic spectral distance does not tend to increase. This, again, can be explained from the the relatively small diameter of the network model, even for large n.

C. Random Geometric Graphs

In this subsection, we analyze a random graph model for which the diameter depends very strongly on the size of the network: the so-called random geometric graph [28]. A random realization of this model can be generated in two steps: (1) locate n random points independently from the uniform distribution on the square $[0, 1]^2$, and (2) connect a pair of points if the Euclidean distance between them is less than a given connectivity radius $\rho > 0$. We evaluate the performance of our algorithm by generating 5 random realizations for different values of $r \in \{3, 4, 5\}$, n in the range $[25 : 25 : 100]$, and $\rho = \sqrt{15/\pi n}$ (i.e., the average degree of each node is 15, [28]). In Table I, we observe how as we increase n, the average diameter of the desired and designed networks grows almost affinely. We also observe how the average quadratic spectral distance $\delta_S(\mathcal{S}_a, \mathcal{S}_b^{(K)})$ remains in the same range of values as we vary the size of the network and the number of moments. Although the spectral error tends to slowly decrease as we use moments of higher order, our results present a consistent performance for different network sizes. Also, in practice, the algorithm presents a good performance when only three spectral moments are used, which can be computed using first-order neighborhoods.

VI. Conclusions

In this paper, we have described a fully decentralized algorithm that iteratively modifies the structure of a network of agents with the objective to control the eigenvalues of the Laplacian matrix. Although every agent was assumed to have access to local information regarding the graph structure, we showed that the group is able to collectively aggregate their local information to take a global optimal decision. This decision corresponds to the most beneficial (greedy) link addition or deletion, defined as the one that minimizes

	Small-World				Preferential Attachment				Random Geometric Graph			
n nodes	25	50	75	100	25	50	75	100	25	50	75	100
$\overline{\phi}_a$	2.6	3	3	3	3	3.4	4	4	4.2	5.4	6.6	7.8
$\overline{\phi}_b^{(5)}$	2.4	3	3.2	3.2	2.6	3.8	3.6	4	3.8	4.8	5.6	6.6
$\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(3)})$	0.098	0.113	0.080	0.056	0.292	0.259	0.320	0.345	0.149	0.171	0.140	0.134
$\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(4)})$	0.081	0.098	0.064	0.046	0.367	0.249	0.243	0.256	0.139	0.147	0.105	0.082
$\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(5)})$	0.194	0.166	0.097	0.043	0.452	0.127	0.243	0.215	0.174	0.159	0.116	0.094

TABLE I
EACH ROW REPRESENTS: n THE NUMBER OF NODES, $\overline{\phi}_a$ THE AVERAGE DIAMETER OF THE RANDOM MODEL (OVER FIVE REALIZATIONS), $\overline{\phi}_b^{(5)}$ THE AVERAGE DIAMETER OF THE GRAPH DESIGNED USING 5 SPECTRAL MOMENTS, AND $\overline{\delta}_S(\mathcal{S}_a, \mathcal{S}_b^{(K)})$ THE AVERAGE QUADRATIC SPECTRAL DISTANCE BETWEEN THE SPECTRUM OF THE RANDOM MODEL AND THE SPECTRUM OF THE NETWORK DESIGNED USING K SPECTRAL MOMENTS.

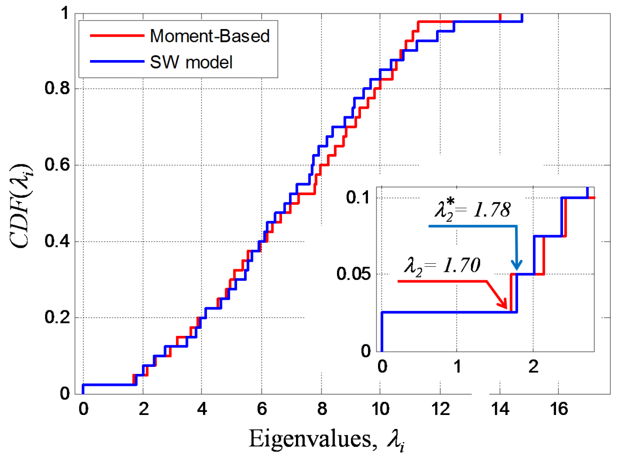


Fig. 2. Empirical cumulative distribution for the eigenvalue spectrum of the small-world graph in Example V-A (blue) and the topology resulting from our algorithm (red).

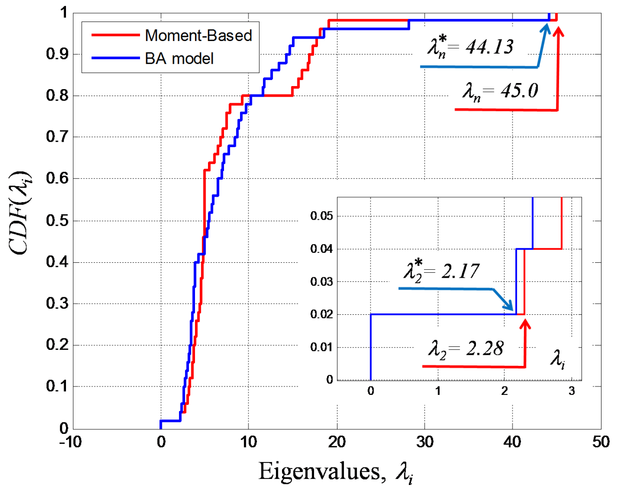


Fig. 3. Empirical cumulative distribution for the eigenvalue spectrum of the power-law graph in Example V-B (blue) and the topology resulting from our algorithm (red).

a distance function that captures the eigenvalue distribution of the network through its Laplacian spectral moments. The aggregation of the local information was achieved via gossip algorithms, which were also used to ensure network connectivity throughout the evolution of the network. Being greedy in nature, our approach is stable by construction. It was also show to perform remarkably well in practice, returning networks that with Laplacian eigenvalues very close to the desired ones.

APPENDIX

Proof of Theorem 2.1: The theorem states that the spectrum $S(A) = \{\lambda_i\}_{i=1}^n$ of any $n \times n$ symmetric matrix A is uniquely characterized by its first $n-1$ spectral moments. First, we use Cayley-Hamilton theorem to prove that the first $n-1$ spectral moments of the spectrum S characterize the whole infinite sequence of moments $(m_k(S))_{k=0}^{\infty}$, as follows. Let $\phi(\lambda) = \det(\lambda I_n - A) = \lambda^n + \alpha_{n-1}\lambda^{n-1} + ... + \alpha_0$, be the characteristic equation of A. Then, from Cayley-Hamilton, we have $\phi(A) = 0$. Multiplying $\phi(A)$ by $\frac{1}{n}A^t$, and applying the trace operator, we have that,

$$\begin{aligned}\frac{1}{n}\text{Tr}\left[A^t\phi(A)\right] &= \frac{1}{n}\text{Tr}\left(A^{t+n}\right) + ... + \alpha_0\frac{1}{n}\text{Tr}\left(A^t\right)\\ &= m_{t+n}(A) + ... + \alpha_0 m_t(A) = 0,\end{aligned}$$

for all $t \in \mathbb{N}$. Therefore, given the sequence of moments $(m_k(A))_{k=0}^{n-1}$, we can use the recursion

$$m_{t+n}(A) = -\alpha_{n-1}m_{t+n-1}(A) - ... - \alpha_1 m_{t+1}(A) - \alpha_0 m_t(A),$$

to uniquely characterize the infinite sequence of moments $(m_k(A))_{k=0}^{\infty}$.

Second, we prove that the infinite sequence of moments $(m_k(A))_{k=0}^{\infty}$ uniquely characterizes the eigenvalue spectrum. Let us define the *spectral measure* of the matrix A with real eigenvalues $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$, as

$$\mu_A(x) = \sum\nolimits_{i=1}^n \delta(x - \lambda_i),$$

where $\delta(\bullet)$ is the Dirac delta function. In what follows, we prove that the spectral measure of A is uniquely characterized by its infinite sequence of spectral moments using Carleman's condition [29]. Since there is a trivial bijection between the eigenvalue spectrum of A and its spectral measure, uniqueness of the spectral measure imply uniqueness of the eigenvalue spectrum.

Carleman's condition states that a measure μ on $\mathbb{R}$ is uniquely characterized by its infinite sequence of moments $(M_k(\mu))_{k=1}^{\infty}$ if (*i*) $M_k(\mu) < \infty$ for all $k \in \mathbb{N}$, and (*ii*)

$$\sum\nolimits_{s=1}^{\infty} (M_{2s}(\mu))^{-1/2s} = \infty.$$

In our case, the moments of the spectral measure μ_A are

$$M_k(\mu_A) = \int_{-\infty}^{+\infty} x^k d\mu_A(x) = \sum_{i=1}^n \lambda_i^k = n\ m_k(A).$$

These moments satisfy: (*i*) $M_k(\mu_A) \leq n\lambda_n^k < \infty$, for any finite matrix A, and (*ii*)

$$\sum_{s=1}^{\infty} (M_{2s}(\mu_A))^{-1/2s} = \sum_{s=1}^{\infty} \left(\sum_{i=1}^{n} \lambda_i^{2s}\right)^{-1/2s} \geq \sum_{s=1}^{\infty} \left(\lambda_n^{2s}\right)^{-1/2s} = \sum_{s=1}^{\infty} \lambda_n^{-1} = \infty,$$

for any $A \neq 0$. As a consequence, the spectral measure of any finite matrix $A \neq 0$ with real eigenvalues is uniquely characterized by $(M_k(\mu_A))_{k=0}^{\infty}$. Since, $M_k(\mu_A) = n\, m_k(A)$, we have that the sequence of moments $(m_k(A))_{k=0}^{n-1}$ uniquely characterizes $(M_k(\mu_A))_{k=0}^{\infty}$. Therefore, the sequence of moments $(m_k(A))_{k=0}^{n-1}$ uniquely characterize the spectral measure μ_A and the real eigenvalue spectrum $S = \{\lambda_i\}_{i=1}^{n}$. ■

References

[1] N. Wiener, *The Mathematics of Self-Organising Systems. Recent Developments in Information and Decision Processes*, Macmillan, 1962.

[2] M.O. Jackson, *Social and Economic Networks*, Princeton University Press, 2008.

[3] V.M. Preciado, *Spectral Analysis for Stochastic Models of Large-Scale Complex Dynamical Networks*, Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., MIT, Cambridge, MA, 2008.

[4] L.M. Pecora and T.L. Carroll, "Master Stability Functions for Synchronized Coupled Systems," *Physics Review Letters*, vol. 80, pp. 2109-2112, 1998.

[5] N.A. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers, 1997.

[6] A. Fax and R. M. Murray, "Information Flow and Cooperative Control of Vehicle Formations," *IEEE Transactions on Automatic Control*, vol. 49, pp. 1465-1476, 2004.

[7] R. Olfati-Saber and R. M. Murray, "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, vol. 49, pp. 1520-1533, 2004.

[8] R. Grone, R. Merris, and V.S. Sunder, "The Laplacian Spectrum of a Graph," *SIAM Journal Matrix Analysis and Applications*, vol. 11, pp. 218-238, 1990.

[9] A. Ghosh and S. Boyd, "Growing Well-Connected Graphs," *Proc. of the 45th IEEE Conference on Decision and Control*, pp. 6605-6611, 2006.

[10] Y. Kim and M. Mesbahi, "On Maximizing the Second-Smallest Eigenvalue of a State Dependent Graph Laplacian," *IEEE Transactions on Automatic Control*, vol. 51, pp. 116-120, 2006.

[11] M.C. DeGennaro and A. Jadbabaie, "Decentralized Control of Connectivity for Multi-Agent Systems," *Proc. of the 45th IEEE Conference on Decision and Control*, San Diego, CA, Dec. 2006, pp. 3628-3633.

[12] L. Xiao, S. Boyd, and S.J. Kim, "Distributed Average Consensus with Least-Mean-Square Deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33-46, 2007.

[13] P.G. Doyle and J.L. Snell, *Random Walks and Electrical Networks*. The Mathematical Association of America, 1984.

[14] A.K. Chandra, P. Raghavan, W.L. Ruzzo, R. Smolensky, and P. Tiwari, "The Electrical Resistance of a Graph Captures its Commute and Cover Times," *Computational Complexity*, vol. 6, no. 4, pp. 312-340, 1996.

[15] V.M. Preciado and A. Jadbabaie, "Moment-Based Spectral Analysis of Large-Scale Networks Using Local Structural Information," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 373-382, 2013.

[16] M. M. Zavlanos, M. B. Egerstedt and G. J. Pappas, "Graph Theoretic Connectivity Control of Mobile Robot Networks.," *Proceedings of the IEEE: Special Issue on Swarming in Natural and Engineered Systems*, vol. 99, no. 9, pp. 1525-1540, 2011.

[17] M. M. Zavlanos and G. J. Pappas, "Distributed Connectivity Control of Mobile Networks," *IEEE Transactions on Robotics*, vol. 24, pp. 1416-1428, 2008.

[18] N. Biggs, *Algebraic Graph Theory*, Cambridge University Press, 2^{nd} Edition, 1993.

[19] D.M. Cvetković, M. Doob, and H. Sachs, *Spectra of Graphs*, 3^{rd} Edition, Wiley-VCH, 1998.

[20] V.M. Preciado, A. Jadbabaie, and G.C. Verghese, "Structural Analysis of Laplacian Spectral Properties of Large-Scale Networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2338-2343, September 2013.

[21] V.M. Preciado and A. Jadbabaie, "From Local Measurements to Network Spectral Properties: Beyond Degree Distributions," *IEEE Conference on Decision and Control*, pp. 2686-2691, 2010.

[22] W.H. Haemers and E. Spence, "Enumeration of Cospectral Graphs," *European Journal of Combinatorics*, vol. 25, no. 2, pp. 199-211, 2004.

[23] N. Alon, "Eigenvalues and Expanders," *Combinatorica*, vol. 6, no. 2, pp. 83-96, 1986.

[24] J. H. Wilkinson, *The Algebraic Eigenvalue Problem,* Oxford University Press, 1965.

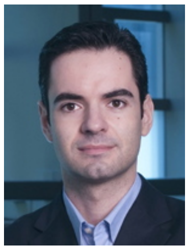
[25] J. Cortes, "Distributed Algorithms for Reaching Consensus on General Functions," *Automatica*, vol. 44, pp. 726-737, 2008.

[26] D.J. Watts and S. Strogatz, "Collective Dynamics of Small World Networks," *Nature*, vol. 393, pp. 440-42, 1998.

[27] A. L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 285, pp. 509-512, 1999.

[28] M. Penrose, *Random Geometric Graphs*. Oxford University Press, 2003.

[29] N.I. Akhiezer, *The Classical Moment Problem and Some Related Questions in Analysis*, Oliver & Boyd, 1965.



Victor M. Preciado received the Ph.D. degree in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology, Cambridge in 2008.

He is currently the Raj and Neera Singh Assistant Professor of Electrical and Systems Engineering at the University of Pennsylvania. He is a member of the Networked and Social Systems Engineering (NETS) program and the Warren Center for Network and Data Sciences. His research interests include network science, dynamic systems, control theory, and convex optimization with applications in socio-technical networks, technological infrastructure, and biological systems.

Michael Zavlanos (S05M09) received the Diploma in mechanical engineering from the National Technical University of Athens (NTUA), Athens, Greece, in 2002, and the M.S.E. and Ph.D. degrees in electrical and systems engineering from the University of Pennsylvania, Philadelphia, in 2005 and 2008, respectively.

From 2008 to 2009, he was a Postdoctoral Researcher in the Department of Electrical and Systems Engineering at the University of Pennsylvania, Philadelphia. He then joined the Stevens Institute of Technology, Hoboken, NJ, as an Assistant Professor of Mechanical Engineering, where he remained until 2012. Currently, he is an assistant professor of mechanical engineering and materials science at Duke University, Durham, NC. He also holds a secondary appointment in the department of electrical and computer engineering. His research interests include a wide range of topics in the emerging discipline of networked control systems, with applications in robotic, sensor, communication, and biomolecular networks. He is particularly interested in hybrid solution techniques, on the interface between control theory, distributed optimization, estimation, and networking.

Dr. Zavlanos is a recipient of the 2014 Office of Naval Research Young Investigator Program (YIP) Award, the 2011 National Science Foundation Faculty Early Career Development (CAREER) Award, and a finalist for the Best Student Paper Award at CDC 2006.