Self-organization of weighted networks for optimal synchronizability

Louis Kempton, Guido Herrmann and Mario di Bernardo

***Abstract*—We show that a network can self-organize its existing topology, i.e. by adapting edge weights, in a completely decentralized manner in order to maximize its synchronizability whilst satisfying local constraints: we look specifically at non-negativity of edge weights, and maximum weighted degree of nodes. A novel multilayer approach is presented which uses a decentralized strategy through which each node can estimate one of two spectral functions of the graph Laplacian, the algebraic connectivity λ_2 or the eigenratio $r = \lambda_n/\lambda_2$. These local estimates are then used to evolve the edge weights so as to maximize λ_2, or minimize r and, hence, achieve globally optimal values for the edge weights for the synchronization of a network of coupled systems.**

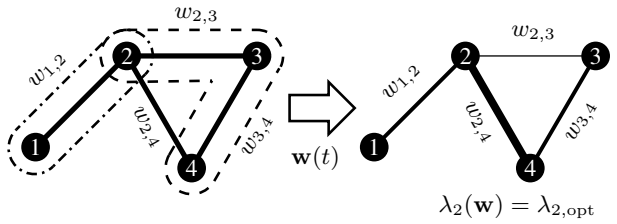

Figure 1. A schematic diagram of the problem where we wish to adapt edge weights $\mathbf{w} = [w_{i,j}]_{m\times 1}$ in time to maximize λ_2 of a given graph with n nodes and m edges, in a distributed manner. For clarity, the horizon of knowledge for edge $\{1, 2\}$ (dot-dash) and node 3 (dashed) are shown. We call nodes 2 and 4 the one-hop neighbourhood of 3, i.e. $\mathcal{N}_3 = \{2, 4\}$. Different values of the edge weights are represented in terms of thickness of the network links.

I. Introduction

Many network systems that we see in nature adapt their structure in time to better perform a specific function. Learning through changing the strength of synapses [1], or through rewiring of a collection of neurons [2] can be seen as a paragon of this observation. For such networks, a desired global behaviour emerges from the local interactions of agents who can adapt their dynamics and interaction strengths in response to their local environment to steer the macroscopic network behaviour and functionality. Devising microscopic strategies able to mimic this ability to adapt and evolve in order to steer the macroscopic behaviour of physical networks in a controlled way can offer a number of benefits in engineering applications. Examples range from the synchronization of power networks [3] to the coordination of robotic swarms [4]. Indeed, adaptive networks can be robust, coping well with missing or broken parts; they are able to self-organize, removing the requirement for a dedicated designer, and adapting to changes in the operating environment in real time. Moreover these systems tend to cope well with an increasing number of agents, adjusting and readapting their dynamics to maintain a desired function [5].

In this paper we focus on the problem of synthesizing a decentralized adaptive strategy able to make an undirected network self-organize into an optimal structure. As a representative problem we choose that of making the network achieve optimal synchronizability, that is, a configuration allowing its nodes to synchronize with maximal range of coupling gains (see [6]–[9] for further details). As discussed in previous work (e.g. [9]–[11]), maximal synchronizability can be achieved by maximizing the algebraic connectivity λ_2, the second smallest eigenvalue of the graph Laplacian, or by minimizing the eigenratio $r = \lambda_n/\lambda_2$, with λ_n being the largest Laplacian eigenvalue. We add some constraints to model the realistic case of each agent having a limited communication bandwidth. Thus, for edges being physically feasible they should have non-negative weight, though it should be noted that allowing negative weights can improve synchronizability [12]. For limited communication, the weights cannot be infinitely large. To efficiently achieve this using local constraints, we upper bound the weighted degree of each node by a constant value, and require that edge weights to be non-negative. A real world example of these constraints are the limitations of the shared link bandwidths common in wired and wireless sensor and actuator networks (e.g., in distributed robot swarms), where distributed optimization of the network can be of great benefit; see, for example, [13].

The problem of rewiring a network or adapting its structure in order to achieve maximal synchronizability has been well studied in previous literature, with many techniques proposed, both for weighted and unweighted, directed and undirected networks. However, previous methods either require global knowledge of the network to find the optimal structure, cannot guarantee optimality when local rules are used solely, or does not provide strong algorithm robustness, in particular when the estimated and optimized eigenvalues are non-unique during optimization. In the first case, it has been shown that finding the structure which maximizes algebraic connectivity can be formulated as a semi-definite program (SDP) [14] and extended to the unweighted case using mixed integer semi-

This research is funded by the Engineering and Physical Sciences Research Council through the Bristol Centre for Complexity Sciences

L. Kempton is with the Bristol Centre for Complexity Sciences, University of Bristol, UK l.kempton@bristol.ac.uk

G. Herrmann is with the Department of Mechanical Engineering, University of Bristol, UK g.herrmann@bris.ac.uk

M. di Bernardo is with the Department of Engineering Mathematics, University of Bristol, UK. Also with the Department of Electrical Engineering and Information Technology, University of Naples Federico II, Italy m.dibernardo@bris.ac.uk

definite programming (MISDP) [15], [16]. To find a solution, it is assumed that the solver has complete knowledge of the network. Likewise, methods employing simulated annealing with edge-rewiring, to find optimal, or near-optimal, graphs [10], [17], [18] also require an external supervisor of the network with global knowledge to evolve the network structure. In the second case, only local information is used to assign edge weights. An example is found in [19] where node degree is used to modify edge weights, enhancing λ_n/λ_2, but convergence towards an optimal structure is not guaranteed. An application of such decentralized algorithms was recently presented in [20] where the problem is discussed of maintaining a minimal algebraic connectivity in a network of mobile robots. This is achieved by increasing the determinant of a reduced Laplacian. Thus the problem of minimizing the algebraic connectivity is addressed indirectly by looking at the Laplacian determinant. Some recent work by [21] and [22] has targeted similar problems to those presented in our paper; control of the algebraic connectivity in a random ad-hoc network, and optimization of λ_2 and λ_n for network averaging. However, in both of these works communication constraints on the nodes are not addressed, and in particular the estimation algorithms for λ_2 and λ_n presented in [22] fail when the controlled eigenvalues become non-unique. This strongly contrasts to our solutions as demonstrated later, and this issue of non-distinctness in the extremal eigenvalues becomes increasingly prevalent as networks become larger.

In this paper we present a new optimal strategy that adapts the edge weights so as to directly maximize λ_2 or minimize λ_n/λ_2 for a given undirected graph. Unlike previous methods which find the optimal network structure, we impose the further constraint of only using information local to each node to adapt the network, see Figure 1 for a schematic of the problem and the horizons of each agent's knowledge. Namely, nodes may only communicate with their one-hop neighbours. Thus, the contributions of this paper are

1) Decentralized estimation of largest eigenvalue λ_n of the Laplacian in modification to the algebraic connectivity estimator of [23].
2) Decentralized estimation of the eigenratio λ_n/λ_2 (ratio of the largest eigenvalue of the Laplacian to the algebraic connectivity) and of its partial derivatives with respect to edge weights.
3) Presentation of an online-capable (distributed) gradient descent and barrier function based optimization approach, which permits the constrained minimization of the ratio of the largest eigenvalue to the algebraic connectivity and for the constrained maximization of the algebraic connectivity. The approach is based on the convexity and quasi-convexity of the optimized variables.
4) Distributed three stage approach for optimization and also control of the estimated eigenvalues of the Laplacian. The algorithm uses the above mentioned gradient descent techniques and the decentralized eigenvalue estimators. Moreover, preliminary ideas to prove stability using a singular perturbation technique are discussed.

This paper is structured accordingly: Section II defines the mathematical problem to be solved. Section III explains the overall approach, where Section III-A presents the novel weight adaptation algorithm, Section III-B reiterates the estimator of λ_2 and of its sensitivity [23], Section III-C introduces the new estimator of λ_n and λ_n/λ_2 and the relevant sensitivities to summarize also and Section III-E discusses the novel combined, distributed optimization strategy with respect the constrained the maximization of λ_2 and the minimizer of λ_n/λ_2. Section IV shows in several numerical examples that it is possible to control and optimize the network weights using either λ_2 or λ_n/λ_2.

II. Problem Formulation

We shall consider a generic homogeneous network of n nonlinear dynamical agents of the form:

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

where $\mathbf{x}_i = [x_j]_{d\times 1}$ is the d-dimensional state vector of the i^{th} system, $\mathbf{F}(\mathbf{x}_i) : \mathbb{R}^d \mapsto \mathbb{R}^d$ is the vector field of the i^{th} isolated system, σ is a scalar global coupling parameter, and $\mathbf{H}(\mathbf{x}_j) : \mathbb{R}^d \mapsto \mathbb{R}^d$ is the coupling function between nodes (see for example Figure 1[1]) The weighted, symmetric graph Laplacian is denoted by $\mathbf{L} = [l_{i,j}]_{n\times n}$, and is defined as a function of edge weights $w_{i,j}$:

$$l_{i,j} = \begin{cases} \sum_{j\in\mathcal{N}_i} w_{i,j} & \text{if } i = j \\ -w_{i,j} & \text{if } i \neq j \end{cases} \tag{2}$$

with $\mathcal{N}_i$ denoting the one-hop neighbourhood of node i, so that $\mathbf{L}$ is a square, symmetric matrix with zero row sum. The eigenvalues of the graph Laplacian are thus real, and can then be ordered as $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$.

As shown in the previous literature [24], [25] and briefly summarized in Appendix A, local transversal stability of the synchronization manifold in a network of identical nonlinear systems can be studied using the Master Stability Function (MSF) approach (see Appendix A for further details). According to this approach, the range of coupling strengths that render the synchronous solution locally transversally stable can be maximized by maximizing the algebraic connectivity λ_2 or minimizing the Laplacian eigenratio r according to the specific shape of the MSF for the systems of interest.

Therefore, if we wish to maximize the synchronizability of a network, then we must consider two cases, which can be formulated in the standard form of an optimization problem

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}} && f(\mathbf{w}) \\ &\text{subject to} && \mathbf{w} \in \mathcal{W} \end{aligned} \tag{3}$$

where $\mathbf{w} = [w_{i,j}]_{m\times 1}$ is the vector of all edge weights, and the objective function $f(\mathbf{w})$ is equal to $-\lambda_2(\mathbf{L})$ [Case 1] or $\lambda_n(\mathbf{L})/\lambda_2(\mathbf{L})$ [Case 2] according to the type of MSF considered. $\mathcal{W}$ is a set of feasible edge weights, which is both

[1]For instance, the one-hop connected dynamics of node 3 are $\dot{\mathbf{x}}_3 = \mathbf{F}(\mathbf{x}_3) - \sigma\left(w_{2,3}(\mathbf{H}(\mathbf{x}_2) - \mathbf{H}(\mathbf{x}_3)) + w_{3,4}(\mathbf{H}(\mathbf{x}_4) - \mathbf{H}(\mathbf{x}_4))\right)$.

closed and convex. For our examples in this paper, we will use the following feasible region,

$$\mathcal{W} = \{\mathbf{w} : \mathbf{w} \geq \mathbf{0} \wedge l_{i,i} \leq \kappa_i, \forall i\} \tag{4}$$

so as to allow only non-negative edge weights, and upper bound the weighted degree of each node by a constant κ_i (feasible edge weights lie in a closed polytope in the positive orthant of $\mathbb{R}^m$).

Remark 1: The feasible set of Equation (4) creates a convex, bounded set, in which the investigated objective function $f(\mathbf{w})$ has a unique minimizer. One can easily change the set $\mathcal{W}$ to include for instance negative lower bounds for $\mathbf{w}$, e.g. $\mathbf{w} \geq -\mathbf{u}$ for $\mathbf{u} \geq \mathbf{0}$. Moreover, it is for instance also possible to include the further constraint $\lambda_n(\mathbf{L}) \leq 1$, which would render the two minimization cases $f(\mathbf{w}) = -\lambda_2(\mathbf{L})$ and $f(\mathbf{w}) = \lambda_n/\lambda_2$ to be equivalent.

III. Description of the Method

To optimize the desired objective (maximizing λ_2 or minimizing r) we take advantage of the fact that $\lambda_2(\mathbf{L})$ is a concave function of the edge weights [14] [$-\lambda_2(\mathbf{L})$ is then a convex function], and $r(\mathbf{L})$ is a quasiconvex function of edge weights [26], thus for either function any local minimizer is a global minimizer. Then, as our set of feasible edge weights is a convex set, we can minimize $-\lambda_2(\mathbf{L})$ or $r(\mathbf{L})$ by gradient descent in a distributed fashion. This gradient descent method forms the top level in a hierarchy of distributed processes (see Figure 2) which is added to an underlying layer designed to estimate the gradient of the chosen objective. Note that Case 2 is an example of a single ratio quasi-convex fractional program, and thus can be transformed to a parameter-free convex program, as described in [26] (Proposition 8).

A. Gradient Descent with Logarithmic Barriers

First, let us describe the *Weight optimizer layer* in Figure 2. The goal of this layer is to minimize the objective function $f(\mathbf{w})$ (which may alternatively be $-\lambda_2$ or r depending on the case we are considering). This is achieved by forcing edge weights in the direction of steepest descent of a modified objective function $g(\mathbf{w})$, which enforces the boundary constraints of the feasible region through the use of logarithmic barriers [27]:

$$g(\mathbf{w}) = f(\mathbf{w}) - \frac{1}{q(t)} \left(\sum_{\{i,j\} \in \mathcal{E}} \log(w_{i,j}) + \sum_{i=1}^{n} \log(\kappa_i - l_{i,i}) \right) \tag{5}$$

For conciseness of notation we have used $\mathcal{E}$ to signify the edge set, so that the first summation is over all edges in the network. The severity of these barriers is determined by the function $q(t)$ which is chosen to be positive monotonic increasing and unbounded (further details on its choice will be given below). The barriers will then become steeper over time and in the case that the optimization problem is solved with at least one inequality constraint being tight, the stationary point will move progressively closer to the edge of the feasible set.

Each edge weight is then adapted in time using gradient descent according to:

$$\ddot{w}_{i,j} = -k_a \frac{\partial g(\mathbf{w})}{\partial w_{i,j}} - c_1 \dot{w}_{i,j} \tag{6}$$

where the sensitivity of the modified objective with respect to an edge weight can be computed as:

$$\frac{\partial g(\mathbf{w})}{\partial w_{i,j}} = \frac{\partial f(\mathbf{w})}{\partial w_{i,j}} - \frac{1}{q(t)} \left(\frac{1}{w_{i,j}} - \frac{1}{\kappa_i - l_{i,i}} - \frac{1}{\kappa_j - l_{j,j}} \right) \tag{7}$$

The constants k_a and c_1 are positive reals. These two degrees of freedom, k_a and c_1, can be used to easily control the transient dynamics which determine how quickly edge weights adapt, similar to [28]. [2]

Remark 2: As mentioned for remark 1, it is easily possible to change the constraints. For instance, considering a modified set to

$$\mathcal{W} = \{\mathbf{w} : \mathbf{w} \geq -\mathbf{u} \wedge \lambda_n(\mathbf{L}) \leq 1\}$$

for $\mathbf{u} \geq \mathbf{0}$, one can rewrite the objective function $g(\mathbf{w})$ (5) to state

$$g(\mathbf{w}) = f(\mathbf{w}) - \frac{1}{q(t)} \left(\sum_{\{i,j\} \in \mathcal{E}} \log(w_{i,j} + u_{i,j}) + \log(1 - \lambda_n(\mathbf{L})) \right).$$

Subsequently, this creates a more complex analysis for the partial derivative $\frac{\partial g(\mathbf{w})}{\partial w_{i,j}}$ due to an extended use of the chain rule. Instead, for the minimization of λ_n/λ_2, it was chosen to separate the local constraints from the global objective, which then needs to be estimated in a distributed manner.

Choice of steepness function: It is sufficient that $q(t)$ be a positive, monotonic increasing, unbounded function, that does not escape to infinity in finite time, for edge weights to converge to the constrained minimum of the desired objective $f(\mathbf{w})$. The simplest choice would simply be $q(t) = t$, but this does not regulate how rapidly the barriers should steepen. A reasonable method to achieve this is to use an approximation of how well the system has converged for the current value of $q(\mathbf{w})$. We can infer how closely the system has converged to the stationary point for a given $q(\mathbf{w})$ by observing the magnitude of the gradient of the modified objective function $||\nabla g(\mathbf{w})||_2$ (the stationary point for given $q(t)$ satisfies $||\nabla g(\mathbf{w})||_2 = 0$). As such, we propose to regulate the growth of $q(t)$ according to the second-order adaptive law:

$$\ddot{q} = \frac{k_b}{||\nabla g(\mathbf{w})||_2 + \delta} - c_2 \dot{q} \tag{8}$$

with k_b, δ and c_2 being positive real constants, that can be used for tuning the response. However, it can be seen that this equation is not fully distributed, violating our requirements ($||\nabla g(\mathbf{w})||_2$ is a global variable). To overcome this problem we assign a local $q_{i,j}$ for each weight, which is updated

[2] We found that this second order solution can be particularly useful for the numerical implementation of the algorithm in contrast to for instance to a first order law, $\dot{w}_{i,j} = -k_a \frac{\partial g(\mathbf{w})}{\partial w_{i,j}}$, which would also work but gives less freedom in tuning.

according to:

$$\ddot{q}_{i,j} = \frac{k_b}{\left|\frac{\partial g(\mathbf{w})}{\partial w_{i,j}}\right| + \delta} - c_2 \dot{q}_{i,j} \tag{9}$$

Then, the microscopic dynamics of each edge weight follows Equation (6), but with the new gradient:

$$\frac{\partial g(\mathbf{w})}{\partial w_{i,j}} = \frac{\partial f(\mathbf{w})}{\partial w_{i,j}} - \frac{1}{q_{i,j}(t)} \left(\frac{1}{w_{i,j}} - \frac{1}{\kappa_i - l_{i,i}} - \frac{1}{\kappa_j - l_{j,j}} \right) \tag{10}$$

It can clearly be seen that the forcing from the logarithmic barrier functions on the adaptive dynamics of a single edge requires only information local to that edge (its own weight $w_{i,j}$, the weighted degrees of its parents $l_{i,i}$ and $l_{j,j}$ and their maximum allowed weighted degrees κ_i and κ_j). Only the sensitivity of the chosen objective function, $\frac{\partial f(\mathbf{w})}{\partial w_{i,j}}$, with respect to the edge remains as a global parameter, which will be computed locally via a set of distributed estimators (in Case 1, by the λ_2 Estimator layer, shown in Figure 2). This will permit the optimization algorithm to be fully decentralized.

In what follows we describe how to implement the layer to obtain a decentralized estimation of the $\frac{\partial f(\mathbf{w})}{\partial w_{i,j}}$. We focus first on the problem of estimating the algebraic connectivity λ_2 and the largest Laplacian eigenvalue λ_n. Then we show how these local estimates can be combined to obtain an estimation of the sensitivities of the two cost functions under consideration with respect to the edge weights.

B. Case 1: Decentralized Estimation of λ_2

To estimate the sensitivities of the algebraic connectivity to variation of the edge weights, we use the distributed strategy by Yang et al. [23] to evaluate the algebraic connectivity of a weighted undirected network in a distributed fashion. The strategy can be implemented as two additional layers, the *Proportional-Integral (PI) Consensus layer* and the λ_2-*Estimator layer* shown in Fig. 2. The dynamics of these two layers can be described by the following set of differential equations inspired by power iteration (see [23] for further details) :

$$\dot{\mathbf{a}} = -k_1 \boldsymbol{\varphi_a} - k_2 \mathbf{L}\mathbf{a} - k_3(\boldsymbol{\psi_a} - \mathbf{1}) \circ \mathbf{a} \tag{11}$$

$$\dot{\boldsymbol{\varphi}}_{\mathbf{a}} = \gamma(\mathbf{a} - \boldsymbol{\varphi_a}) - k_P \mathbf{L}\boldsymbol{\varphi_a} - k_I \mathbf{L}\boldsymbol{\chi_a} \tag{12}$$

$$\dot{\boldsymbol{\chi}}_{\mathbf{a}} = k_I \mathbf{L}\boldsymbol{\varphi_a} \tag{13}$$

$$\dot{\boldsymbol{\psi}}_{\mathbf{a}} = \gamma(\mathbf{a}^2 - \boldsymbol{\psi_a}) - k_P \mathbf{L}\boldsymbol{\psi_a} - k_I \mathbf{L}\boldsymbol{\omega_a} \tag{14}$$

$$\dot{\boldsymbol{\omega}}_{\mathbf{a}} = k_I \mathbf{L}\boldsymbol{\psi_a} \tag{15}$$

For concise notation, component-wise product of vectors is signified by $\circ$, and squaring a vector is taken component-wise also, so that $\mathbf{a}^2 = \mathbf{a} \circ \mathbf{a}$. Here $\mathbf{a}$ is an estimate of the eigenvector associated with λ_2, and Equation (11) forms the λ_2 estimator layer, and requiring two further global variables: the arithmetic mean of $\mathbf{a}$, $\langle \mathbf{a} \rangle \triangleq 1/n \sum a_i$, and the mean of the squared components, $\langle \mathbf{a}^2 \rangle \triangleq 1/n \sum a_i^2$.

These global variables are estimated in a distributed manner using a further layer (see Figure 2) consisting of two Proportional-Integral (PI) consensus estimators [29], with $\boldsymbol{\varphi_a}$ being an estimate of $\langle \mathbf{a} \rangle \mathbf{1}$, and $\boldsymbol{\psi_a}$ being an estimate of $\langle \mathbf{a}^2 \rangle \mathbf{1}$.

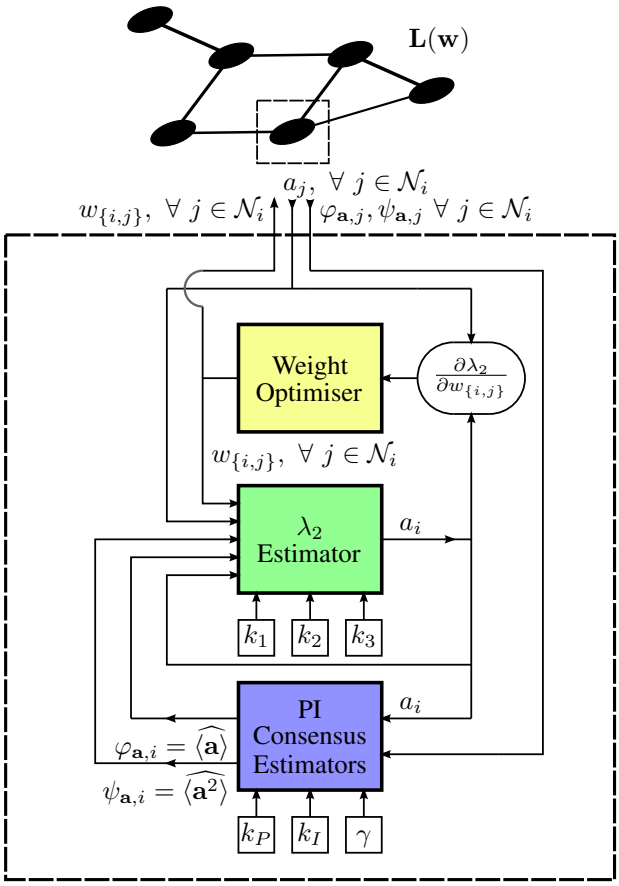

Figure 2. Schematic diagram of the distributed multilayer approach for λ_2 maximization proposed in the paper, with faster processes at the bottom, and slower processes built on top. The processes occurring in a single node are expanded, and all nodes follow identical rules.

The variables $\boldsymbol{\chi_a}$ and $\boldsymbol{\omega_a}$ are the integrator variables in the two PI consensus estimators.

The parameters k_1, k_2 and k_3 control three actions which can be summarized as deflation, direction update, and renormalization, respectively [23]. The result of these actions is that for $\mathbf{a}$, there are two stable stationary points, which together are global attractors if $k_1 > k_3 \geq k_2 \lambda_n$:

$$\mathbf{a}^* = \pm \mathbf{v_2}(\mathbf{L}) \sqrt{\frac{n(k_3 - k_2 \lambda_2)}{k_3}} \tag{16}$$

where $\mathbf{v_2}$ is the unit eigenvector associated with λ_2. Hence, λ_2 can be estimated by node i using Equations (11), (12), (14) and (16). Specifically, by rearranging and noting that $\psi_{\mathbf{a}i}$ is node i's estimate of $||\mathbf{a}||_2^2/n$, a local estimate of λ_2 at node i can be obtained:

$$\widehat{\lambda_2}^{(i)} = \frac{k_3}{k_2}(1 - \psi_{\boldsymbol{a}i}) \tag{17}$$

C. Case 2: Estimation of $\frac{\lambda_n}{\lambda_2}$

By reversing the sign of the direction update in (11), the distributed λ_2 estimator may be modified to form a distributed estimator for λ_n and the estimated eigenvector $\mathbf{b}$ associated

with it. Specifically, we can use the set of equations:

$$\dot{\mathbf{b}} = -k_1\boldsymbol{\varphi_b} + k_2\mathbf{L}\mathbf{b} - k_3(\boldsymbol{\psi_b} - \mathbf{1}) \circ \mathbf{b} \tag{18}$$

$$\dot{\boldsymbol{\varphi}}_\mathbf{b} = \gamma(\mathbf{b} - \boldsymbol{\varphi_b}) - k_P\mathbf{L}\boldsymbol{\varphi_b} - k_I\mathbf{L}\boldsymbol{\chi_b} \tag{19}$$

$$\dot{\boldsymbol{\chi}}_\mathbf{b} = k_I\mathbf{L}\boldsymbol{\varphi_b} \tag{20}$$

$$\dot{\boldsymbol{\psi}}_\mathbf{b} = \gamma(\mathbf{b}^2 - \boldsymbol{\psi_b}) - k_P\mathbf{L}\boldsymbol{\psi_b} - k_I\mathbf{L}\boldsymbol{\omega_b} \tag{21}$$

$$\dot{\boldsymbol{\omega}}_\mathbf{b} = k_I\mathbf{L}\boldsymbol{\psi_b} \tag{22}$$

so that $\mathbf{b}$ will converge to an eigenvector associated with λ_n. Following a similar argument to that made above for λ_2, node i can then compute a local estimate of λ_n as:

$$\widehat{\lambda_n}^{(i)} = \frac{k_3}{k_2}(\psi_{\boldsymbol{b}i} - 1) \tag{23}$$

This process can be seen as the mirror to the algebraic connectivity estimator presented in [23]. Instead of the term associated with k_2 contracting the state towards consensus so that the slowest mode dominates, it now expands away from consensus so that the fastest mode dominates. Again, k_1 ensures that the estimate is perpendicular to the consensus mode and k_3 acts to force the vector $\mathbf{b}$ towards a Euclidean ball of radius $\sqrt{n}$, stopping the estimate from diverging.

D. Inferring Sensitivity with respect to Edge Weights

These two methods which make decentralized estimates of λ_2 and λ_n can further be used by each node to obtain local estimates of the sensitivity of each objective function with respect to an edge weight $w_{i,j}$. We derive, in a similar manner to [30], the partial derivatives of a generic eigenvalue, say λ, of the weighted graph Laplacian with respect to the edge weight $w_{i,j}$. Imagine that we know an associated right eigenvector $\mathbf{v}$ so that we can define the unit eigenvector $\hat{\mathbf{v}} = \mathbf{v}/||\mathbf{v}||_2$. As $\mathbf{L}$ is symmetric, we know that the unit left eigenvector is the transpose $\hat{\mathbf{u}} = \hat{\mathbf{v}}^\top$. Then, pre-multiplying the eigenvector relation $\mathbf{L}\hat{\mathbf{v}} = \lambda\hat{\mathbf{v}}$ by the unit left eigenvector yields:

$$\hat{\mathbf{v}}^\top\mathbf{L}\hat{\mathbf{v}} = \lambda\hat{\mathbf{v}}^\top\hat{\mathbf{v}} = \lambda \tag{24}$$

Taking the component-wise derivative with respect to the edge weight $w_{i,j}$, we find that:

$$\frac{\partial\lambda}{\partial w_{i,j}} = \frac{\partial\hat{\mathbf{v}}^\mathsf{T}}{\partial w_{i,j}}\mathbf{L}\hat{\mathbf{v}} + \hat{\mathbf{v}}^\mathsf{T}\frac{\partial\mathbf{L}}{\partial w_{i,j}}\hat{\mathbf{v}} + \hat{\mathbf{v}}^\mathsf{T}\mathbf{L}\frac{\partial\hat{\mathbf{v}}}{\partial w_{i,j}}$$

As $\mathbf{L}$ is symmetric, it is ensured that

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

Thus,

$$\frac{\partial\lambda}{\partial w_{i,j}} = \hat{\mathbf{v}}^\mathsf{T}\frac{\partial\mathbf{L}}{\partial w_{i,j}}\hat{\mathbf{v}} \tag{25}$$

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

$$\begin{aligned}\frac{\partial\lambda}{\partial w_{1,2}} &= \frac{1}{||\mathbf{v}||_2^2}\mathbf{v}^\mathsf{T}\left(\begin{array}{cc|c} +1 & -1 & 0 \\ -1 & +1 & \\ \hline & 0 & 0 \end{array}\right)\mathbf{v} \\ &= \frac{1}{||\mathbf{v}||_2^2}\left(v_1 - v_2\right)^2\end{aligned} \tag{26}$$

where $\mathbf{v} = [v_1, v_2, \ldots, v_n]^\top$.

From the distributed estimation procedure outlined in Equations (11), (12) and (14), and the set of Equations (18), we can compute estimates for the eigenvectors associated with λ_2 and λ_n, respectively $\mathbf{a}$ and $\mathbf{b}$, and estimates of the mean of their squared components, respectively $\boldsymbol{\psi_a}$ and $\boldsymbol{\psi_b}$. Using the relabelling argument that any two nodes could be labelled 1 and 2, along with Equation (26), we arrive at node i's distributed estimates for the sensitivities:

$$\widehat{\frac{\partial\lambda_2}{\partial w_{i,j}}}^{(i)} = \frac{(a_i - a_j)^2}{n\psi_{\boldsymbol{a}i}} \tag{27}$$

$$\widehat{\frac{\partial\lambda_n}{\partial w_{i,j}}}^{(i)} = \frac{(b_i - b_j)^2}{n\psi_{\boldsymbol{b}i}} \tag{28}$$

Moreover, applying the quotient rule for differentiation, we can also estimate the sensitivity of r:

$$\widehat{\frac{\partial r}{\partial w_{i,j}}}^{(i)} = \frac{\widehat{\lambda_2}^{(i)}\widehat{\frac{\partial\lambda_n}{\partial w_{i,j}}}^{(i)} - \widehat{\lambda_n}^{(i)}\widehat{\frac{\partial\lambda_2}{\partial w_{i,j}}}^{(i)}}{\left(\widehat{\lambda_n}^{(i)}\right)^2} \tag{29}$$

E. Decentralized optimization

By using Equation (27) or Equation (29), it is now possible to estimate the gradient of either objective function in a fully decentralized manner. Hence, the edge weights can be adapted by steepest descent, also in a decentralized fashion and satisfying the feasibility constraints, using Equations (6), (9) and (10) where the term $\frac{\partial f(\mathbf{w})}{\partial w_{i,j}}$ is replaced by its corresponding estimate according to Equation (27) or Equation (29). For Equation (27), this also requires the use of two additional layers, the *Proportional-Integral (PI) Consensus layer* and the λ_2*-Estimator layer* as in Equations (11)-(15). Fig. 2 summarizes this. In the minimization of λ_n/λ_2 case, twice the number of additional layers are required, as indicated in Fig. 3. These are Equations (11)-(15) and also Equations (18)-(22).

It is worth noting that we require the estimators to converge faster than the weights adapt so that the sensitivities can be estimated with enough accuracy. By exploring the parameter space numerically, it appears that separating the time-scales between layers by an order of magnitude is sufficient for well-behaved convergence onto the solution. Note that it is always possible to tune the time-scale separation of the processes involved by tuning the gains of the PI layers, estimator layers, and those determining the response of the adaptive edge weight dynamics.

A proof of convergence of each of the layers involved in our approach can be found in [23], [29], [31]. It is evident from [23], [29], that the two fastest, lowest layers, the Proportional-Integral (PI) Consensus layer and the λ_2/λ_2-Estimator layer, i.e. Equations (11)-(15) and also Equations (18)-(22), are each exponentially stable. The right hand sides of these estimators each are analytic, as are the locations of the equilibria, which are also well-defined when the associated eigenvalues are distinct. The weight adaptation layer [31] of Equation (6) (together with Equation (9)) is asymptotically stable and the

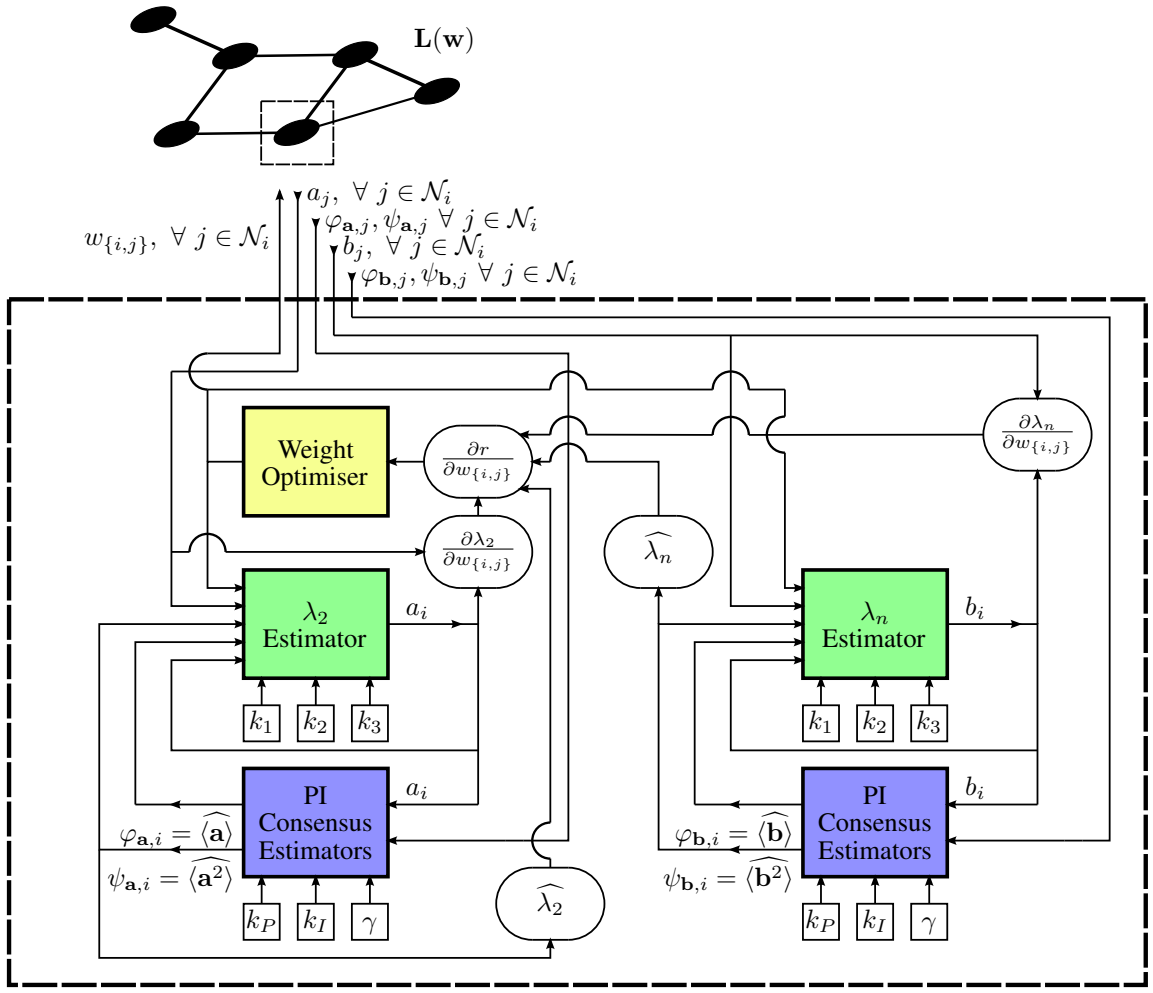

Figure 3. Schematic diagram of our distributed method for r minimization. The flow diagram structure is a schematic of how variables interact in the different estimation layers within a single node i. On the left of the diagram, the algebraic connectivity is being estimated (green block) which requires two PI average consensus estimators (blue block). From these blocks, distributed estimates are made for λ_2 and $\partial\lambda_2/\partial w_{i,j}$ for each edge that connects to node i. In a similar fashion, on the right hand side of the diagram, distributed estimates of λ_n and $\partial\lambda_n/\partial w_{i,j}$ are being made. Combining these estimates, a local estimate of $\partial r/\partial w_{i,j}$ is formed, and this is fed into the weight optimizer (yellow block), which uses this estimate and the local boundary constraints to inform the adaptation of each of the weights.

right hand side is again analytic for finite value of $q(t) > 0$. The lack of exponential stability in the slowest layer does not easily allude to singular perturbation arguments (e.g. [32]). Hence, this requires a more careful analytical investigation of the required time scale separation. A proof of convergence is currently under investigation and will be presented for space reasons elsewhere.

IV. Applications

A. Decentralized Constrained λ_2 Maximization

Firstly, we shall illustrate a representative application of our strategy to adapt the edges of a small undirected network for maximizing the algebraic connectivity λ_2, Case 1.

The optimization problem we wish to solve is to find the edge weights which yield the maximum λ_2 for a given connected, undirected graph, when no weighted degree at any node may exceed the number of neighbours which it connects to: $\kappa_i = \sum_{j\in\mathcal{N}_i} 1$ in Equation (4), and no edge weights may be negative. Through this choice of a feasible set, we can be sure that total weight in the network may not increase over time, and any improvement in λ_2 must be due to better distribution of edge weights, rather than absolute increase of their total value.

We have randomly generated a connected graph of twenty nodes, and set all weights at $w_{i,j}(0) = 1 - \epsilon$, where ϵ is small so that the initial edge weights lie in the interior of the feasible set. When all weights are equal to 1, the initial algebraic connectivity is found to be $\lambda_2(0) \approx 0.3344$. Edge weights are then adapted according to the algorithm in Figure 2, using Equations (11), (12) and (14) with (27) to estimate the gradient of the objective function, and (6) to adapt each edge weight according to gradient descent. In particular, the gains k_a and c_1 from Equation (6) are tuned to values of 0.01 and 1 respectively, so that the weight adaptation response is slow and over-damped. Moreover, we tune the control parameters so that the PI consensus estimator layer is approximately 10 times faster than the algebraic connectivity estimator, which in turn is chosen to be 10 times faster than the weight adaptation layer. This gives us sufficient separation in time scale between the layers.

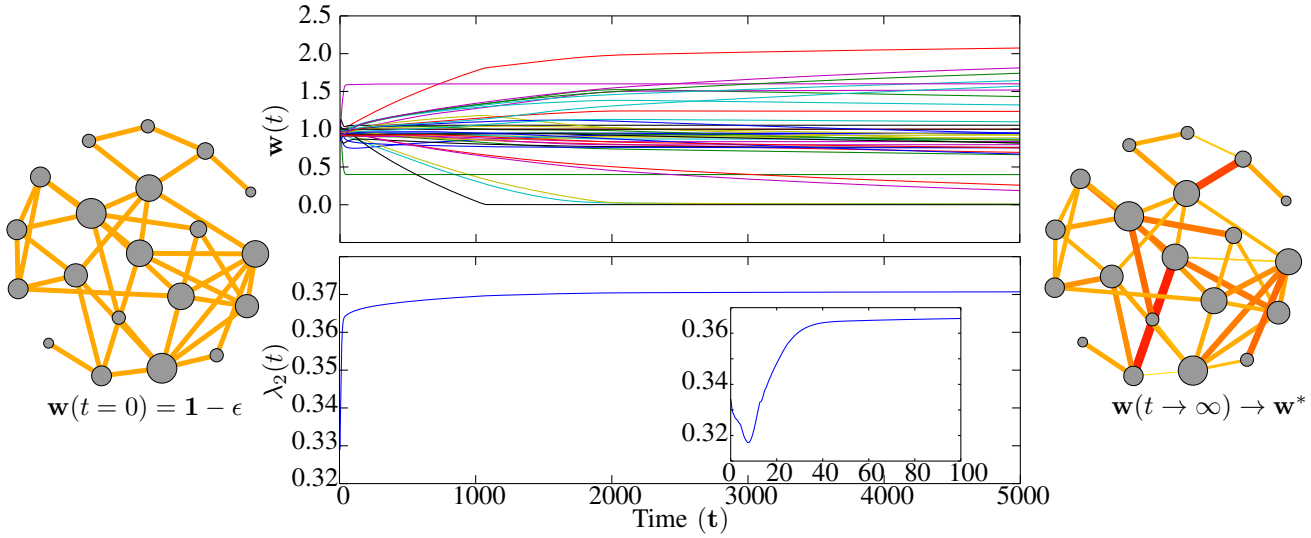

Figure 4. Edge weights are adapted in time according to our distributed algorithm. The algebraic connectivity λ_2 of the network increases over time and settles to a maximum value. To see the initial dip in connectivity while the layers of estimators take time to converge, see the inset. In the diagrams showing the initial and end state of the network, node diameter is proportional to maximum allowed weighted degree, κ_i, and edge thickness and colour is proportional to weight, higher weights are redder and thicker.

As edge weights are adapted, shown in Figure 4, the algebraic connectivity initially decreases until the estimator layers have properly converged, whereupon it increases rapidly, as the nadir of the potential well of the objective function for a given set of $q_{i,j}$ is reached. Finally, as the $q_{i,j}$ increase and the logarithmic barriers enforcing the feasible set become steeper, the edge weights slowly converge to their optimal values, see Figure 4.

It should be noticed that the algebraic connectivity converges more rapidly than the slowest converging edges due to weak sensitivity of λ_2 with respect to some edges. After 5000 seconds of simulated time, the algebraic connectivity has reached a value of $\lambda_2(5000) \approx 0.3707$, which is over 99.9% of the result for optimal λ_2 found using the method of [14]: $\lambda_2^* \approx 0.3708$. At this time, some of the edges have yet to converge, but it is known that three edges will tend to zero value, and could be removed from the network at no detriment to the algebraic connectivity.

B. Decentralized Constrained λ_n/λ_2 Minimization

Now we go on to Case 2: minimization of the ratio $r = \lambda_n/\lambda_2$, using the same network, initial weights and constraint conditions as the previous example. Through the use of Equations (11), (12) and (14) the eigenvector associated with λ_2 is estimated, and the set of Equations (18) are used to estimate the eigenvector associated with λ_n. Then, Equations (17) and (23) are used to infer the values of λ_2 and λ_n respectively, and Equations (27) to (29) to estimate the sensitivities, $\partial r/\partial w_{i,j}$.

Again, weight adaptation is controlled using the estimated sensitivity from Equation (6), where the partial derivative of the modified objective function is given by Equation (10) to account for the multiple $q_{i,j}$ which grow according to Equation (9). The interplay of these equations is summarized in the schematic diagram shown in Figure 3.

As shown in Figure 5, the eigenratio decreases rapidly from its initial value $r(0) = 26.8$, and converging onto $r = 20.6$ within 4000 seconds. However, it can be seen that the edge weights continue to exhibit small amplitude oscillations at steady state. These oscillations are due to the optimal Laplacian possessing non-distinct extremal eigenvalues.

Specifically, due to the limitation of the distributed estimator for λ_n only being able to estimate an associated eigenvector, rather than the entire eigenspace, weights will adapt to reduce λ_n so that it now becomes λ_{n-1}, and the distributed estimator takes time to converge onto the new λ_n. In this time where the estimator provides an incorrect value, edge weights may overshoot, and set up a persistent oscillation around the optimal value. As the speed of the distributed estimator layers is increased relative to the weight adaptation layers, oscillations become smaller in amplitude and higher in frequency.

The presence of these oscillations does not affect in any case convergence towards an optimal value. In practice each edge weight can be "locked" to its average steady-state value as is typically done in the practical implementation of adaptive controllers, for example in [33]. This result strongly contrasts the outcome of [22], where non-distinctness of eigenvalues damages the continuation of the algorithms.

C. Decentralized Control of λ_2

As opposed to maximizing or minimizing these spectral functions of the graph Laplacian, control can also be achieved with minor changes to the weight adaptation law. To show this, we demonstrate control of the algebraic connectivity to a desired reference value ρ using the weight adaptation law:

$$\ddot{w}_{i,j} = -k_a \frac{\partial g(\mathbf{w})}{\partial w_{i,j}} \left(\rho - \widehat{\lambda_2}^{(i)} \right) - c_1 \dot{w}_{i,j} \qquad (30)$$

Edge weights are simply forced in the direction of increasing λ_2 when the current estimated algebraic connectivity is less

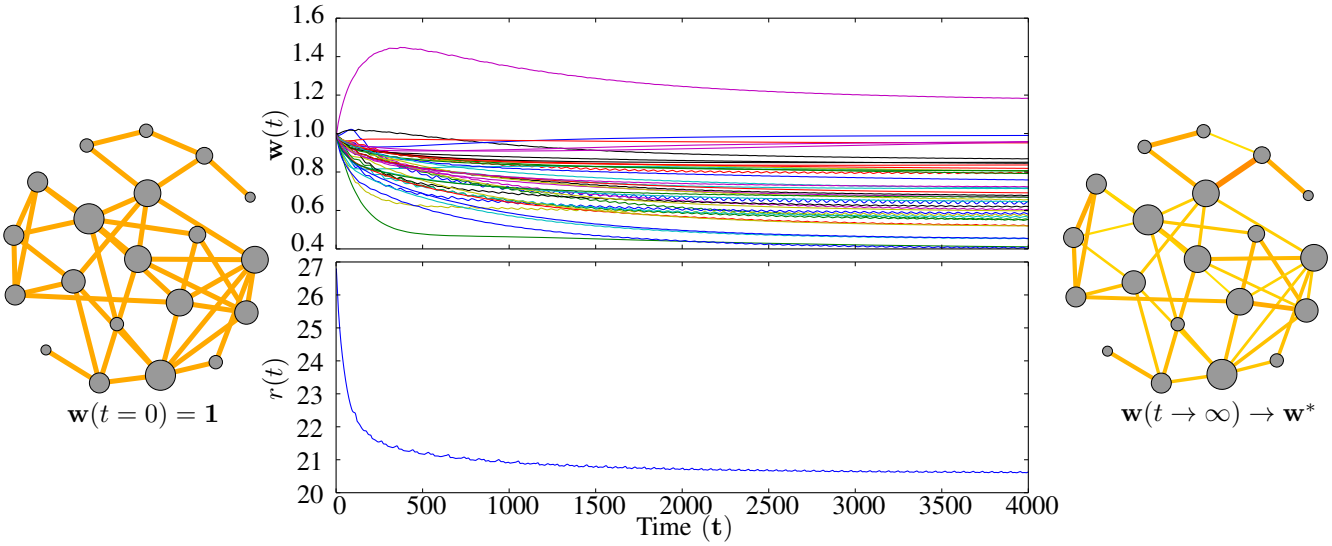

Figure 5. Edge weights are adapted in time according to our distributed algorithm for the minimization of $r = \lambda_n/\lambda_2$. It can be seen that the eigenratio r decreases over time settling into a persistent oscillation. Again, in the network diagrams, edge thickness and redness is proportional to edge weight, and node diameter is proportional to the maximum allowed weighted degree at each node.

than the desired value, and forced in the opposite direction in the case that the current estimate of the algebraic connectivity exceeds the desired value.

As an example, we use the case described in Section IV-A, but now force the λ_2 towards a value of $\rho = 0.25$, Figure 6. This is a feasible reference algebraic connectivity for the weighted network as it is less than the constrained maximum of 0.3708, but the solution is clearly not unique. To demonstrate one of the benefits of using an adaptive method, we simulate the loss of an edge between two nodes, by setting the weight to 0 at $t = 2000$ seconds. At this point, algebraic connectivity is temporarily reduced, but through the action of the controller, the set point of $\lambda_2 = 0.25$ is quickly recovered.

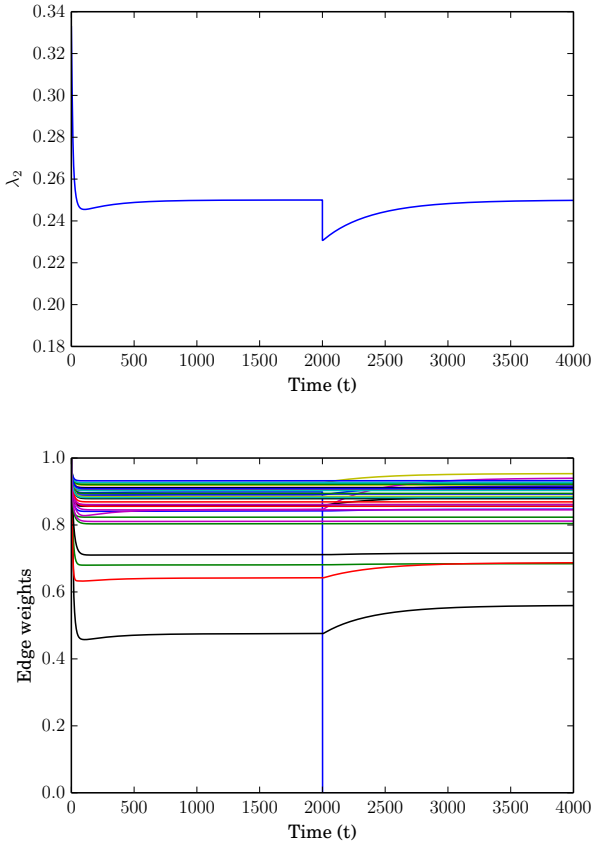

Figure 6. With a small change to the weight adaptation law, Equation (30), the chosen spectral function can be controlled. In this case we force λ_2 to the value of 0.25. We simulate an edge failing in the network at $t = 2000$ seconds by setting the weight to 0, at which point other edges adapt again to compensate for the loss of connectivity.

V. Conclusion

We have shown that a network of agents may cooperate together, exchanging only local information, and only maintaining a small number of states in local memory of each node (9 for λ_2 maximization, and 14 for r minimization, and not growing with the size of the network), and still come to agreement on a globally optimal network structure. This has been achieved through the use of a multi-layer weight adaptation algorithm. By means of such a multilayer approach, we showed that it is possible for nodes in the network to locally estimate the sensitivities of two global spectral functions of the graph Laplacian: the algebraic connectivity λ_2 and the eigenratio λ_n/λ_2. This information can then be used by the edges to locally adapt their weights so as to maximize the network synchronizability. It is therefore possible for the network to self-organize in order to steer some macroscopic observables, such the algebraic connectivity or eigenratio, by using only microscopic information. Moreover, the method can be used to evolve the network edge weights in order to control the value of the algebraic connectivity so as for it to achieve some desired value. In this way a fully decentralized adaptive control strategy is shown to steer in real-time a global macroscopic property of the graph without relying on any global information.

Future work will be aimed at investigating how to extend this approach to control other emerging properties of a network of interest and, hence, achieve control of other macroscopic properties of the network via local adaptive rules. We are also interested in formulating a discrete time implementation of the method with a rigorous complexity analysis of the algorithm, and bounds for control parameters to guarantee the stability of the algorithm.

REFERENCES

[1] Dean V Buonomano and Michael M Merzenich. "Cortical plasticity: from synapses to maps". In: *Annu. Rev. Neurosci.* 21.1 (1998), pp. 149–186.

[2] Dmitri B Chklovskii, BW Mel, and K Svoboda. "Cortical rewiring and information storage". In: *Nature* 431.7010 (2004), pp. 782–788.

[3] Florian Dorfler, Michael Chertkov, and Francesco Bullo. "Synchronization in complex oscillator networks and smart grids". In: *Proceedings of the National Academy of Sciences* 110.6 (2013), pp. 2005–2010.

[4] Anders Lyhne Christensen, Rehan O'Grady, and Marco Dorigo. "From fireflies to fault-tolerant swarms of robots". In: *IEEE T. Evolut. Comput.* 13.4 (2009), pp. 754–766.

[5] Dragoslav D Siljak. *Decentralized control of complex systems*. Courier Corporation, 2011.

[6] Mauricio Barahona and Louis M Pecora. "Synchronization in small-world systems". In: *Phys. Rev. Lett.* 89.5 (2002), p. 054101.

[7] Guanrong Chen and Zhisheng Duan. "Network synchronizability analysis: a graph-theoretic approach". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 18.3 (2008), p. 037102.

[8] Francesc Comellas and Silvia Gago. "Synchronizability of complex networks". In: *J. Phys. A* 40.17 (2007), p. 4483.

[9] Ming Zhao, Guan-rong Chen, Tao Zhou, et al. "Enhancing the network synchronizability". In: *Frontiers of Physics in China* 2.4 (2007), pp. 460–468.

[10] Luca Donetti, Pablo I Hurtado, and Miguel A Munoz. "Entangled networks, synchronization, and optimal network topology". In: *Phys. Rev. Lett.* 95.18 (2005), p. 188701.

[11] Ernesto Estrada, Silvia Gago, and Gilles Caporossi. "Design of highly synchronizable and robust networks". In: *Automatica* 46.11 (2010), pp. 1835–1842.

[12] Lin Xiao and Stephen Boyd. "Fast linear iterations for distributed averaging". In: *Systems & Control Letters* 53.1 (2004), pp. 65–78.

[13] Sharanya Eswaran, Archan Misra, Flavio Bergamaschi, et al. "Utility-based Bandwidth Adaptation in Mission-oriented Wireless Sensor Networks". In: *ACM Trans. Sen. Netw.* 8.2 (Mar. 2012), 17:1–17:26. DOI: 10.1145/2140522.2140530.

[14] Stephen Boyd. "Convex optimization of graph Laplacian eigenvalues". In: *Proceedings of the International Congress of Mathematicians*. Vol. 3. 2006, pp. 1311–1319.

[15] Mohammad Rafiee and Alexandre M. Bayen. "Optimal network topology design in multi-agent systems for efficient average consensus". In: *Proceedings of the 49th IEEE Conference on Decision and Control*. 2010, pp. 3877–3883.

[16] Ran Dai and Mehran Mesbahi. "Optimal topology design for dynamic networks". In: *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*. 2011, pp. 1280–1285.

[17] Thomas E Gorochowski, Mario di Bernardo, and Claire S Grierson. "Evolving enhanced topologies for the synchronization of dynamical complex networks". In: *Phys. Rev. E* 81.5 (2010), p. 056212.

[18] P.S. Skardal, D. Taylor, and J. Sun. "Optimal Synchronization of Complex Networks". In: *Phys. Rev. Lett.* 113 (2014), p. 144101.

[19] Adilson E Motter, CS Zhou, and Jrgen Kurths. "Enhancing complex-network synchronization". In: *Europhys. Lett.* 69.3 (2005), p. 334.

[20] Hasan Alihusain Poonawala, Aykut C Satici, Hazen Eckert, et al. "Collision-free Formation Control with Decentralized Connectivity Preservation for Nonholomic Wheeled Mobile Robots". In: *IEEE Trans. Control Netw. Syst.* 2.2 (2015), pp. 122–130.

[21] P. Di Lorenzo and S. Barbarossa. "Distributed Estimation and Control of Algebraic Connectivity Over Random Graphs". In: *IEEE Transactions on Signal Processing* 62.21 (2014), pp. 5615–5628.

[22] A. Bertrand and M. Moonen. "Topology-aware distributed adaptation of Laplacian weights for in-network averaging". In: *21st European Signal Processing Conference (EUSIPCO 2013)*. 2013, pp. 1–5.

[23] Peng Yang, Randy A Freeman, Geoffrey J Gordon, et al. "Decentralized estimation and control of graph connectivity for mobile sensor networks". In: *Automatica* 46.2 (2010), pp. 390–396.

[24] Louis M Pecora and Thomas L Carroll. "Master stability functions for synchronized coupled systems". In: *Phys. Rev. Lett.* 80.10 (1998), p. 2109.

[25] Liang Huang, Qingfei Chen, Ying-Cheng Lai, et al. "Generic behavior of master-stability functions in coupled nonlinear dynamical systems". In: *Phys. Rev. E* 80.3 (2009), p. 036204.

[26] Siegfried Schaible. "Fractional programming". In: *Zeitschrift für Operations Research* 27.1 (1983), pp. 39–54.

[27] Frank E. Curtis and Jorge Nocedal. "Flexible penalty functions for nonlinear constrained optimization". In: *IMA J. Numer. Anal.* 28.4 (2008), pp. 749–769.

[28] Pietro DeLellis, Mario di Bernardo, and Maurizio Porfiri. "Pinning control of complex networks via edge snapping". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 21.3 (2011), p. 033119.

[29] Randy A Freeman, Peng Yang, and Kevin M Lynch. "Stability and convergence properties of dynamic average consensus estimators". In: *Proceedings of the 45th IEEE Conference on Decision and Control*. 2006, pp. 398–403.

[30] Xiaoli Li, Shanwei Zhang, and Yugeng Xi. "Connected flocking of multi-agent system based on distributed eigenvalue estimation". In: *Proceedings of the 30th Chinese Control Conference*. IEEE. Yantai, 2011, pp. 6061–6066.

[31] Louis Kempton, Guido Herrmann, and Mario di Bernardo. "Adaptive weight selection for optimal consensus performance". In: *Proceedings of the 53rd Annual Conference on Decision and Control*. IEEE. 2014, pp. 2234–2239.

[32] Frank Hoppensteadt. "On systems of ordinary differential equations with several parameters multiplying the derivatives". In: *Journal of Differential Equations* 5.1 (1969), pp. 106 –116.

[33] Sebusang E.M. Sebusang and David P. Stoten. "Controller gain bounding in the minimal control synthesis algorithm". In: *Proceedings of the 30th Southeastern Symposium on System Theory*. IEEE. 1998, pp. 141–145.

APPENDIX A

In networks of identical, coupled, nonlinear systems, as described by Equation (1), the predominant technique for determining the local stability of the synchronous solution is the Master Stability Function (MSF) approach [24], [25]. The MSF $\Psi(\alpha)$ plots the Largest Lyapunov Exponent (LLE) of the system governed by Equation (31): the dynamics of virtual displacement $\delta\mathbf{y}$ between two trajectories in the vicinity of the synchronous solution $\mathbf{s}(t)$, as a function of the coupling strength α between them. The matrices $\mathbf{DF}$ and $\mathbf{DH}$ are the Jacobians of the vector field of the isolated system $\mathbf{F}(\mathbf{x})$ and coupling function $\mathbf{H}(\mathbf{x})$ respectively, evaluated on the synchronous solution $\mathbf{s}(t)$. For more details, please see for example [24], [25].

$$\frac{d\delta\mathbf{y}}{dt} = (\mathbf{DF}(\mathbf{s}) - \alpha\mathbf{DH}(\mathbf{s}))\delta\mathbf{y} \tag{31}$$

In intervals where α is such that the LLE of Equation (31) is negative (the MSF $\Psi(\alpha)$ is negative), any small distance between trajectories near to the synchronous solution will decay away, and thus the synchronous solution is locally transversally stable.

As an illustrative example, we will consider a network of coupled identical Rössler oscillators (as in [25]) with individual dynamics governed by:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \mathbf{F}(\mathbf{x}) \triangleq \begin{bmatrix}-x_2 - x_3\\ x_1 + \frac{1}{5}x_2\\ \frac{1}{5} + x_3(x_1 - 9)\end{bmatrix} \tag{32}$$

And we will inspect the MSFs under two separate diffusive coupling conditions given by:

$$\mathbf{H}_1(\mathbf{x}) \triangleq \begin{bmatrix}x_1\\ 0\\ 0\end{bmatrix},\ \mathbf{H}_2(\mathbf{x}) \triangleq \begin{bmatrix}0\\ x_2\\ 0\end{bmatrix} \tag{33}$$

Observing Figure 7, we can see that both MSFs are positive at $\alpha = 0$ indicating that the Rössler oscillator with parameters as given in (Equation (32)) is chaotic (trajectories with nearby initial condition diverge). Under the coupling strategy $\mathbf{H}_1$ the MSF initially decreases becoming negative first before rising again to cross the axis once more resulting in a negative interval of the MSF, $\alpha \in (0.186, 4.61)$. In the classification of [25] this characteristic shape with a single bounded negative interval is called Γ_2, and coupling that is too weak or too strong will result in desynchronization. Under the coupling strategy $\mathbf{H}_2$ a different characteristic shape is seen where the negative interval of the MSF $\Psi_2(\alpha)$ ($\alpha \in (0.156, \infty]$) is right unbounded. For any sufficiently strong coupling the sychronous solution will be stable. In the classification scheme of [25] this characteristic shape is deemed Γ_1.

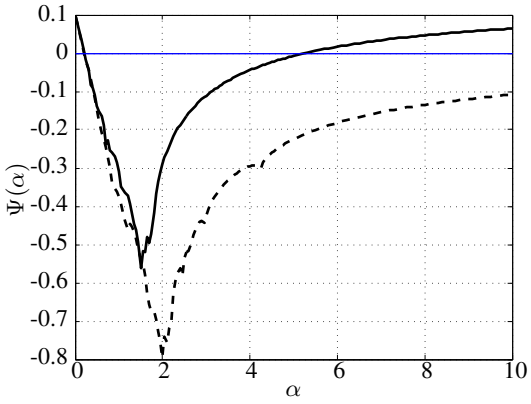

Figure 7. The two MSFs $\Psi_1(\alpha)$ (solid) and $\Psi_2(\alpha)$ (dashed) for the Rössler oscillator (Equation (32)) and two different couplings $\mathbf{H}_1$ and $\mathbf{H}_2$ respectively are given. The MSFs display class Γ_2 (solid) and Γ_1 (dashed) characteristic shapes, respectively. For the synchronous solution $\mathbf{s}(t)$ to be locally, transversally stable, the spectrum of $\sigma\mathbf{L}$ must lie inside the negative interval.

In general, for a given MSF $\Psi(\alpha)$, intervals in the scalar argument α for which its value is negative fall into two categories: right-unbounded $\Psi(\alpha) < 0$ for $\alpha \in (\alpha_1, \infty)$, or proper bounded $\Psi(\alpha) < 0$ for $\alpha \in (\alpha_1, \alpha_2)$. Using the classification system of Huang et al. [25], those MSFs which contain only one negative interval and that interval is right-unbounded are called Γ_1, and those for which their sole negative interval is proper bounded are called Γ_2. These two types of negative interval lead to two ideas of synchronizability, both of which have previously been presented in the literature [6], [8].

For systems with a Γ_1 MSF, the synchronous solution is stable if $\sigma\lambda_2(\mathbf{L}) > \alpha_1$ (Case 1). That is, networks with greater algebraic connectivity λ_2 require a lower global coupling strength σ, and are thus more easily synchronized. On the other hand, for systems with a Γ_2 MSF, the synchronous solution can only be stable if $\alpha_1 < \sigma\lambda_2 \leq \sigma\lambda_n < \alpha_2$ (Case 2). Thus, such systems can only admit a locally transversally stable synchronous solution if $\lambda_n/\lambda_2 < \alpha_2/\alpha_1$. Graphs with lower eigenratio $r = \lambda_n/\lambda_2$ are then deemed more synchronizable as a greater range of global coupling strengths will allow a stable synchronous solution [6]. Therefore, depending on the shape of the specific MSF for the network system, one of these two spectral functions of the graph Laplacian determines its synchronizability, and this is why we consider the two cases in this paper.