Decentralized Estimation of the Algebraic Connectivity for Strongly Connected Networks

Hasan A Poonawala and Mark W Spong

***Abstract*— The second smallest eigenvalue $\underline{\lambda}_2(\mathcal{L})$ of the Laplacian $\mathcal{L}$ of a network G is a parameter that captures important properties of the network. Applications such as synchronization of networked systems, consensus-based algorithms and network connectivity control may require one to regulate the magnitude of $\underline{\lambda}_2(\mathcal{L})$ in order to achieve suitable network performance. The problem of decentralized estimation of $\underline{\lambda}_2(\mathcal{L})$ for directed graphs is thus a relevant problem, yet it has received little attention thus far. We present an algorithm for its estimation and demonstrate its performance.**

I. Introduction

The eigenvalues of the graph Laplacian $\mathcal{L}(G)$ [1] of a network G contain important information regarding the network's properties and performance. The second smallest eigenvalue $\underline{\lambda}_2$ of $\mathcal{L}(G)$ is particularily significant. Note that for the case when $\mathcal{L}$ is asymmetric, the eigenvalues are ordered based on their absolute values. In the case of undirected networks, $\underline{\lambda}_2(\mathcal{L})$ is called the Fiedler value [2]. The Fiedler value can be used to determine whether an undirected network is connected or not. The value of $\underline{\lambda}_2(\mathcal{L})$ for a directed network can be used to deduce whether the network has a globally reachable node [3] or not. Therefore, $\underline{\lambda}_2(\mathcal{L})$ is called the algebraic connectivity of the network. For both directed and undirected networks, the rate of convergence of several consensus and synchronization algorithms depends on the magnitude of $\underline{\lambda}_2(\mathcal{L})$ [3], [4].

In the case of Multi-Robot Networks (MRN), the communication network is state dependent and hence $\underline{\lambda}_2(\mathcal{L})$ changes depending on the motion of the robots. This requires monitoring of $\underline{\lambda}_2(\mathcal{L})$ and possibly the adoption of control strategies to influence its value. Thus estimation of $\underline{\lambda}_2(\mathcal{L})$ is a relevant problem in several applications.

Several methods to estimate the algebraic connectivity for undirected graphs have been proposed, using power iterations [5], decentralized orthogonal iterations [6], Fast Fourier Transforms [7] and dynamical systems theory [8]. The methods in [6] and [7] are computationally expensive. The third method is highly effective and has been implemented in experiments [9], [10] where the objective is to preserve network connectivity. The method in [5] estimates the k^{th} power of a matrix C derived from the Laplacian, using power iterations [11]. The largest eigenvalue of C can then be obtained from the limit of the k^{th} root of the induced infinity norm of C^k as k increases. The estimate of the Fiedler value is then computed directly from the estimated largest eigenvalue of C. However, these methods do not apply to directed graphs, since in this case $\underline{\lambda}_2(\mathcal{L})$ may be complex, and the methods above only return real values.

The decentralized estimation of $\underline{\lambda}_2(\mathcal{L})$ for directed networks has received little attention thus far. One method for its estimation is given in [12]. The method proposes the construction of a matrix $\mathcal{A}_{L_1}$ similar to C in [5], however $\mathcal{A}_{L_1}$ may not be symmetric. Again, $\underline{\lambda}_2(\mathcal{L})$ can be computed directly from the largest eigenvalue of $\mathcal{A}_{L_1}$. Since $\mathcal{A}_{L_1}$ may be asymmetric, the power iteration only yields an estimate of the absolute values of the elements of the eigenvector ν corresponding to the largest eigenvalue of $\mathcal{A}_{L_1}$. The arguments of the elements of ν are obtained by finding the root of a nonlinear function. The Jacobian of this function turns out to be singular on an uncountable set (see Appendix) , which makes the numerical computation of the root unreliable, and in the authors' experience unsuccessful.

Another method was presented in [13], which can be used to obtain $|\underline{\lambda}_2(\mathcal{L})| \in \mathbb{R}$ even if $\underline{\lambda}_2(\mathcal{L})$ is complex. In contrast, the method presented in this paper obtains the complex number $\underline{\lambda}_2(\mathcal{L})$ and not just its modulus. Similar to [5], [12], the adjacency matrix A of the network is converted into a matrix Q, whose largest eigenvalue is a function of $\underline{\lambda}_2(\mathcal{L})$. A modified power iteration algorithm [14] can be used to compute the dominant eigenvalue of Q if it is complex. This algorithm combines a power iteration with the construction of a quadratic equation whose roots contain the largest eigenvalues of Q. Solving only a quadratic equation makes the method more reliable when compared to that in [12]. If the largest eigenvalue of Q is real, then standard power iterations can be used to estimate it.

The main contribution of this paper is to identify an iterative algorithm that allows each node in a directed graph to obtain an estimate of $\underline{\lambda}_2(\mathcal{L})$ through local communication only. The algorithm is shown to converge as the iterations proceed. A criterion is provided by which one can decide when to stop the iterations, while ensuring sufficient accuracy in the estimate. Examples are provided which show the performance of the estimation method.

II. Preliminaries

In this section we introduce various definitions and notations from graph theory that will be used in this paper.

A weighted directed graph G is a tuple consisting of a set of vertices $V = \{1, ..., n\}$ (also called nodes) and a set of edges E. An edge ϵ is an ordered pair (i, j) which indicates

Hasan A Poonawala is with the Erik Jonsson School of Engineering and Computer Science, University of Texas at Dallas, Richardson, Texas, USA `hasanp@utdallas.edu`

Mark W Spong is the Dean of the Erik Jonsson School of Engineering and Computer Science, University of Texas at Dallas, Richardson, Texas, USA `mspong@utdallas.edu`

that a connection exists which starts at node j and ends at node i. Thus (i,j) is an *in-edge* of node i and an *out-edge* of node j. We can define the set of neighbors $\mathcal{N}_i$ of agent i as

$$\mathcal{N}_i = \{j \in V | (i,j) \in E\}$$

which has cardinality n_i. We also define $\bar{\mathcal{N}_1} = \mathcal{N}_i \cup i$.

Each edge $(i,j) \in E$ is associated with a strength $w_{ij} \in (0,1]$, giving rise to the adjacency matrix $A = \{a_{ij}\} \in \mathbb{R}^{n \times n}$, given by

$$a_{ij} = \begin{cases} w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The Laplacian of the graph is then given by

$$\mathcal{L} = D - A$$

where D is a diagonal matrix whose i^{th} diagonal element is $\sum_{j=1}^{n} a_{ij}$. The Laplacian $\mathcal{L}$ always has an eigenvalue at 0, corresponding to a right eigenvector given by $\frac{1}{\sqrt{n}}\mathbf{1}_n$, where $\mathbf{1}_n \in \mathbb{R}^n, \mathbf{1}_n = [1, \dots, 1]^T$. The second smallest eigenvalue[1] of $\mathcal{L}$ is denoted by $\underline{\lambda}_2(\mathcal{L})$.

An undirected network is connected if a path exists between any two nodes, which is equivalent to the Fiedler value being non-zero. For directed graphs, there are multiple notions of connectivity. The most relevant one is that of strong connectivity. A directed network is strongly connected if there exists a directed path between any two nodes of the network.

We can now state the following properties [12]:

- $\underline{\lambda}_2(\mathcal{L})$ has multiplicity 1 if the graph is strongly connected
- $\underline{\lambda}_j(\mathcal{L}), j \in 2, \dots, n$ have positive real part if the graph is strongly connected

The graph Laplacian can be converted into a row-stochastic and non-negative matrix $S \in \mathbb{R}^{n \times n}$, by using the transformation

$$S = I_n - \varepsilon \mathcal{L} \tag{2}$$

where I_n is the identity matrix of size n and $\varepsilon > 0$ is a sufficiently small number. This was done in [15], with a view to analysis of discrete-time consensus protocols. It turns out that the same matrix is useful in the estimation of $\underline{\lambda}_2(\mathcal{L})$. Assuming that each edge weight w_{ij} is bounded above by 1, then selecting $\varepsilon \leq 1/n$ ensures S is non-negative. We now show that S is an irreducible matrix if G is strongly connected.

Lemma II.1. *Consider a graph G with adjacency matrix A and matrix $S \in \mathbb{R}^{n \times n}$ defined by* (2)*. If G is strongly connected, then S is irreducible.*

Proof. If the graph G is strongly connected, A is an irreducible matrix [16]. The matrix S is of the form $kA + B$ where k is a non-zero scalar and B is a diagonal matrix. Suppose that S is reducible. Then there exists a permutation matrix $P \in \mathbb{R}^{n \times n}$ such that P^TSP is in upper block triangular form. This implies that $kP^TAP + P^TBP$ is upper block triangular. Since B is diagonal, so is P^TBP. Thus, P^TAP must also be upper block triangular, which contradicts the fact that A is irreducible. Hence, S must be irreducible. □

[1]The n^{th} smallest and largest eigenvalues of a matrix A are denoted by $\underline{\lambda}_n$(A) and λ_n(A) respectively

The choice of $\varepsilon \leq 1/n$ also ensures that the diagonal elements are non-zero, implying that the trace of S is non-zero. Since S is irreducible and its trace is positive, it is a primitive matrix [17]. This fact is central to the success of the estimators presented in this paper.

III. Decentralized Estimation of $\underline{\lambda}_2(\mathcal{L})$

Consider a graph G with adjacency matrix A and graph Laplacian $\mathcal{L}$. We want each node to estimate $\underline{\lambda}_2(\mathcal{L})$. This estimation must be decentralized, meaning that each node may only receive information via communication with its neighbors. Another constraint is that each node only knows part of the matrices A and $\mathcal{L}$. In particular, node i only knows the variables a_{ij} for $j \in \{1, 2, \dots n\}$.

The estimation method presented in this paper requires the following assumption:

A1 The graph G is strongly connected.

As described in Section I, $\underline{\lambda}_2(\mathcal{L})$ can be estimated by estimating the largest eigenvalue of a matrix Q using a modified power iteration method. The matrix Q is given by

$$Q = S - \frac{1}{\|\gamma_1\|^2}\gamma_1\gamma_1^T \tag{3}$$

where S is the non-negative row stochastic matrix defined by (2), and γ_1 is the left eigenvector of S corresponding to the eigenvalue 1 such that $\gamma_1^T\mathbf{1}_n = 1$. If the eigenvalues of S are denoted by $\lambda_i(S)$ for $i \in \{1, 2, \dots, n\}$ such that $|\lambda_i(S)| \geq |\lambda_{i+1}(S)|$, then the eigenvalues of Q can be shown to be $\{0, \lambda_2(S), \lambda_3(S), \dots, \lambda_n(S)\}$ ([12], Lemma 4). Thus, $\lambda_1(Q) = \lambda_2(S)$, where $\lambda_1(Q)$ is the largest eigenvalue of Q. The value of $\underline{\lambda}_2(\mathcal{L})$ can be computed from $\lambda_2(S)$ using the relation

$$\underline{\lambda}_2(\mathcal{L}) = \frac{1}{\varepsilon}(1 - \lambda_2(S)) = \frac{1}{\varepsilon}(1 - \lambda_1(Q)) \tag{4}$$

The power iteration method involves the computation of a vector $z(k) \in \mathbb{R}^n$ given by $z(k) = Q^k z(0)$, where $z(0) \in \mathbb{R}^n$ is some initial condition. This computation can be implemented in an iterative manner using the equation

$$z(k+1) = Qz(k) \tag{5}$$

where $z(k)$ is obtained at the k^{th} iteration. Three consecutive values $z(k+1)$, $z(k)$ and $z(k-1)$ can be used to compute $\lambda_2(S) = \lambda_1(Q)$, for sufficiently large k. In a decentralized implementation of (5), node i computes the i^{th} component of $z(k+1)$. This computation requires each node to know γ_1 and $z(k)$.

It can be shown [15] that for any primitive stochastic matrix S,

$$\lim_{l \to \infty}(S)^l = \mathbf{1}_n\gamma_1^T \tag{6}$$

Suppose $\omega \in \mathbb{R}^n$ is a distributed vector, such that each node i holds the i^{th} element ω_i. The authors in [15] use (6) to design a consensus algorithm given by

$$w_i(l+1) = \sum_{k \in \mathcal{N}_i}^{n} s_{ik} w_k(l) \tag{7}$$

where $S = \{s_{ij}\}$. The authors show that $\lim_{l\to\infty} w_i(l) = \sum_i^n \gamma_{1,i} w_i(0)$, where $\gamma_{1,i}$ is the i^{th} element of γ_1. This readily implies a method to distribute the γ_1 to all agents, and through γ_1 any other vector $v \in \mathbb{R}^m$. In order to obtain γ_1, each agent implements n copies of (7), with the i^{th} robot's initial value for the j^{th} copy given by the kronecker delta product δ_i^j.

Once γ_1 is known, any distributed n dimensional vector v can be completely spread to any node by implementing n copies of (7) with the i^{th} robot's initial value for the j^{th} copy given by $v_i \delta_i^j$. The estimate at each node converges to an n dimensional vector ω such that the i^{th} element is $v_i \gamma_{1,i}$. The vector v is obtained as $\Gamma^{-1}\omega$, where $\Gamma = \text{diag}(\gamma_1)$. This method can be used to spread $z(k+1)$ to all nodes, once $z_i(k+1)$ has been computed at each node. This procedure can also be found in [12].

With these details in place, we are ready to present an estimator that allows each node to estimate $\underline{\lambda}_2(\mathcal{L})$ using communication with its neighbors only. This is the main result of this paper.

Theorem III.1. *Consider a row-stochastic matrix $S(G)$ corresponding to a strongly connected directed graph G, as defined in* (2). *Consider the following algorithm:*

Algorithm 1 *Let the i^{th} node of the graph G perform the following iterative estimation scheme*

E1 Initialize an estimate $z_i(0) = \frac{1}{\sqrt{(n)}}$

E2 Perform the iterations

$$\begin{aligned} z_i(k+1) = &\left(\sum_{j\in\mathcal{N}_i}^{n} s_{ij} z_j(k)\right) \\ &- \frac{1}{\|\gamma_1(k)\|^2} \gamma_{1,i}(k)(\gamma_1^T z(k)) \end{aligned} \tag{8}$$

where γ_1 and $z(k)$ are obtained using estimators defined by (7) *with appropriate initial conditions*

E3 Store the result of three consecutive iterations $z_p(k), z_p(k-1)$ and $z_p(k-2)$ for $p \in i \cup \mathcal{N}_i$

E4 Solve the $n_i + 1$ linear equations

$$z_p(k) + b z_p(k-1) + c z_p(k-2) = 0$$

for b and c using the method of least squares , where $p \in i \cup \mathcal{N}_i$.

E5 Compute the roots $x_{i,1}(k)$ and $x_{i,2}(k)$ of the equation

$$x^2 + bx + c = 0$$

E6 Compute

$$x_{i,3}(k) = z_i(k)/z_i(k-1)$$

Then, as $k \to \infty$, at least one of the estimates $x_{i,j}(k) \to \lambda_2(S)$, where $j \in \{1,2,3\}$.

Proof. When all n nodes implement steps $S1$ and $S2$, the resulting computation is the decentralized implementation of the power iteration step (5), as shown below:

$$\begin{aligned} z(k+1) &= Sz(k) - \frac{1}{\|\gamma_1\|^2}\gamma_1\gamma_1^T z(k) \\ &= \left(S - \frac{1}{\|\gamma_1\|^2}\gamma_1\gamma_1^T\right) z(k) \\ &= Qz(k) \end{aligned} \tag{9}$$

We now study the form of $z(k+1)$ as k increases. The eigenvectors v_i of Q are identical to those of S [12], and are independent. Let the v_i have indices such that their corresponding eigenvalues λ_i are ordered as

$$|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$$

Hence, the initial condition $z(0)$ for the power iteration can be expressed as

$$z(0) = \sum_{i=1}^{n} \beta_i v_i \tag{10}$$

so that

$$\begin{aligned} z(k) &= Q^k z(0) = \sum_{i=1}^{n} \beta_i Q^k v_i \\ &= \sum_{i=1}^{n} \beta_i \lambda_i^k v_i \end{aligned} \tag{11}$$

<u>Case 1:</u> $\lambda_1(Q) = \bar{\lambda}_2(Q)$ or $\lambda_1(Q), \lambda_2(Q) \in \mathbb{R}$

Let $b_0, c_0 \in \mathbb{R}$ be the coefficients of the monic second order polynomial whose roots are the two largest eigenvalues eigenvalues of Q (either both complex conjugates or both real numbers). Then

$$\lambda_j^2 + b_0\lambda_j + c_0 = 0, \; j \in \{1,2\} \tag{12}$$

Consider the sum $f(b,c) = z_k + bz_{k-1} + cz_{k-2}$. Using (11) we obtain

$$\begin{aligned} f(b,c) &= \sum_{i=1}^{n} \beta_i\lambda_i^k v_i + b(\sum_{i=1}^{n} \beta_i\lambda_i^{k-1} v_i) + c(\sum_{i=1}^{n} \beta_i\lambda_i^{k-2} v_i) \\ &= \sum_{i=1}^{n} \beta_i(\lambda_i^2 + b\lambda_i + c)\lambda_i^{k-2} v_i \\ &= \sum_{i=j-1}^{n} \beta_i(\lambda_i^2 + b\lambda_i + c)\lambda_i^{k-2} v_i + \epsilon_k \end{aligned} \tag{13}$$

where $\epsilon_k = \sum_{i=j}^{n} \beta_i(\lambda_i^2 + b\lambda_i + c)\lambda_i^{k-2} v_i$, and j is the index of the largest eigenvalue such that $|\lambda_j(Q)| < |\lambda_2(Q)|$. If $\lambda_1(Q)$ is complex, then $j = 3$ because of $A1$. Clearly, $f(b_0, c_0) = \epsilon_k$. Consider the solutions to $f(b,c) = 0$. The i^{th} node computes this by solving the $n_i + 1$ linear equations

$$z_p(k) + \begin{bmatrix} z_p(k-1) & z_p(k-2) \end{bmatrix} \begin{bmatrix} b \\ c \end{bmatrix} = 0 \tag{14}$$

where $p \in \mathcal{N}_i \cup \{i\}$. The relation $f(b_0, c_0) = \epsilon_k$ at node i is equivalent to the $n_i + 1$ linear equations given by

$$z_p(k) + \begin{bmatrix} z_p(k-1) & z_p(k-2) \end{bmatrix} \begin{bmatrix} b_0 \\ c_0 \end{bmatrix} = \epsilon_{k,p} \tag{15}$$

We can combine (14) and (15) to obtain the system of equations

$$\begin{bmatrix} z_p(k-1) & z_p(k-2) \end{bmatrix} \begin{bmatrix} b_0 - b \\ c_0 - c \end{bmatrix} = \epsilon_{k,p} \tag{16}$$

Since $|\lambda_2| > |\lambda_3|$, as $k \to \infty$, $|\epsilon_k| \ll |z_k|$. This implies that as $k \to \infty$, $(b, c) \to (b_0, c_0)$. Note that $z(k-1)$ and $z(k-2)$ are independent, since $\lambda_1(Q)$ is not real, and $v_2 = \bar{v}_1$. Thus, the matrix formed from $z(k-1)$ and $z(k-2)$ in (16) has rank 2. In turn, $(x_{i,1}, x_{i,2}) \to (\lambda_1(Q), \bar{\lambda}_1(Q)) = (\lambda_2(S), \bar{\lambda}_2(S))$.

<u>Case 2:</u> $|\lambda_1(Q)| > |\lambda_2(Q)|$

This case occurs when $\lambda_1(Q)$ is real. The estimate can be expressed as

$$x_{i,3}(k) = \frac{z_i(k)}{z_i(k-1)} = \frac{\beta_1 \lambda_1^k v_1 + \sum_{i=2}^n \beta_i \lambda_i^k v_i}{\beta_1 \lambda_1^{k-1} v_1 + \sum_{i=2}^n \beta_i \lambda_i^{k-1} v_i} \tag{17}$$

Since $|\lambda_1| > |\lambda_2|$, the terms in the sum must become negligible compared to the first term, as $k \to \infty$. Then clearly

$$\lim_{k \to \infty} \frac{z_i(k)}{z_i(k-1)} = \lambda_1 \tag{18}$$

This completes the proof. □

Remark 1. The Fiedler value $\underline{\lambda}_2(\mathcal{L})$ can now be obtained as $\frac{1}{\varepsilon}(1 - \lambda_2(S))$.

Remark 2. The Theorem states that one of three estimates converges to the true value of $\lambda_2(S)$. We propose a stopping criterion in the next section which is used to extract the estimated value of $\lambda_2(S)$ from the sequences of values $x_{i,j}(k)$.

IV. Numerical Implementation and Examples

An important issue that must be resolved before implementing the estimator in Section III is to decide at which iteration k the estimation may be stopped, and one of the $x_{i,j}(k), j \in \{1, 2, 3\}$ be accepted as the estimate of $\lambda_2(S)$ for each node i. The authors in [12] discuss methods to ensure convergence of the estimates of γ_1 and ω_i. We now describe a method to detect when the convergence of one of the $x_{i,j}$ has occurred.

Our stopping criterion relies on the observation that for large values of k, the error in the estimate decreases according to a power law. For the case when $\lambda_2(S)$ is real, this property can be seen from Equation (17). Consider the case when $\lambda_2(S)$ is complex. We note that

$$\epsilon_k = \sum_{i=3}^{n} \beta_i (\lambda_i^2 + b\lambda_i + c) \lambda_i^{k-2} v_i \tag{19}$$

which implies that for a sufficiently large k,

$$\epsilon_k \approx \beta_i (\lambda_i^2 + b\lambda_i + c) \lambda_i^{k-2} v_i \tag{20}$$

Equation (20) indicates that $\|\epsilon_k\|_2$ decreases according to a power law, for sufficiently large k. From (16), we see that the error in estimated values (b_0, c_0) will follow this power law. We can model this behavior as

$$\begin{aligned} |b(k) - b_0| &= \lambda |b(k-1) - \hat{b}_0| \\ |c(k) - c_0| &= \lambda |c(k-1) - \hat{c}_0| \end{aligned} \tag{21}$$

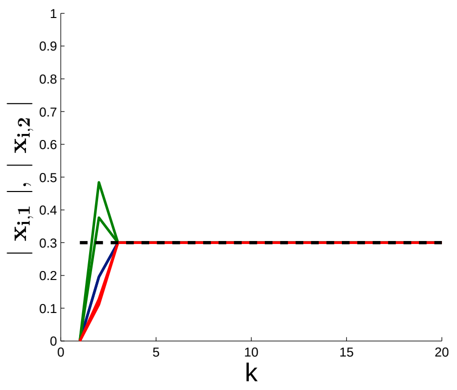


Fig. 1: Absolute values of the estimates of $\lambda_2(S_1)$ for each agent given by $x_{i,1}$ and $x_{i,2}$. The dashed black line indicates $|\lambda_2(S_1)|$.

It is straightforward to solve for $(\hat{b}_0, \hat{c}_0)$ using (21), given the current estimates $b(k)$ and $c(k)$ and a small number of previous estimates. As estimates $(b(k), c(k))$ converge to (b_0, c_0), the predictions $(\hat{b}_0, \hat{c}_0)$ approach (b_0, c_0). Thus, when the difference between the current estimate $(b(k), c(k))$ at iteration k and the predicted true value $(\hat{b}_0, \hat{c}_0)$ is smaller than some specified threshold, we can stop the iterations with the estimate at iteration k (or predicted value) accepted as our final estimate. An additional condition is imposed that requires the difference between the predicted values at iteration k and $k-1$ to be smaller than some threshold.

We now describe the convergence of all three estimates $x_{i,3}$. If $\lambda_2(S)$ is complex , then $(x_{i,1}, x_{i,2})$ converge to $(\lambda_2(S), \bar{\lambda}_2(S))$ but $x_{i,3}$ does not converge to any number. When $\lambda_2(S) \in \mathbb{R}$ and $\lambda_3(S)$ is complex, then $x_{i,3}$ converges to $\lambda_2(S)$ but $(x_{i,1}, x_{i,2})$ do not converge to any value. When $\lambda_2(S), \lambda_3(S) \in \mathbb{R}$ then $(x_{i,1},\ x_{i,2})$ converges to $(\lambda_2(S), \lambda_3(S))$ (or $(\lambda_3(S), \lambda_2(S))$) and $x_{i,3}$ converges to $\lambda_2(S)$.

In order to demonstrate the performance of the algorithm, we implement the estimation algorithm on three row-stochastic matrices. The first was used as an example in [12], and is presented to enable a comparison with their work.

Example 1. *Estimate* $\lambda_2(S_1)$, *where*

$$S_1 = \begin{bmatrix} 0.3 & 0 & 0.7 \\ 0.5 & 0.5 & 0 \\ 0.2 & 0.2 & 0.6 \end{bmatrix}$$

In Figure 1 we can observe that the estimate has converged to the true value of 0.3 by the fourth iteration (only the absolute value is plotted for representation). The stopping criterion is satisfied at the fourth iteration for all nodes (see Figure 2). We continue the iterations in order to facilitate a comparison with Figure 1*b* in [12], wherein the estimates do not settle at the true value. It is possible to accept the mean of several iterations as the estimate, however we submit that our algorithm has better performance.

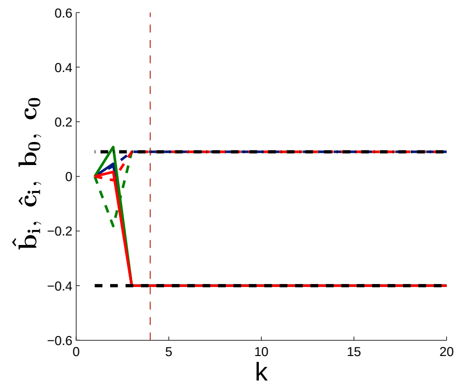
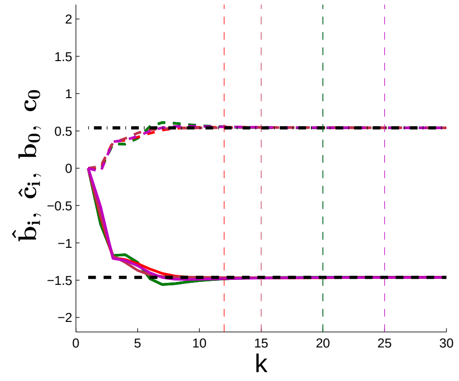
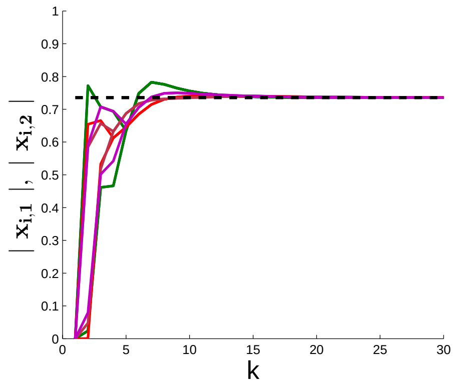


Fig. 2: The estimates of b and c for each node in Example 1. The dashed vertical line marks the iteration at which the stopping criterion is satisfied. The dashed horizontal black lines are the true values.



(a) The estimates of b and c for each node. The estimates and predictions converge as k increases. The dashed vertical lines indicate the iteration at which the stopping criterion is met for each node.



(b) Absolute values of the estimates of $\lambda_2(S_2)$ for each agent given by $x_{i,1}$ and $x_{i,2}$. The dashed black line indicates $|\lambda_2(S_2)|$.

Fig. 3: Simulation results for the second example. Plots of the same color corresponds to the same node. The sets $\bar{\mathcal{N}}_1$ and $\bar{\mathcal{N}}_2$ are identical, hence their estimates are identical and only four plots can be seen.

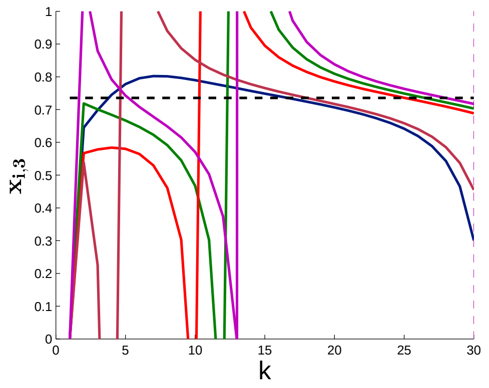


Fig. 4: The estimates $x_{i,3}$ of $\lambda_2(S_2)$ using (17) for all five nodes. The sequence of estimates fail to meet the stopping criterion for any k.

Example 2. *Estimate* $\lambda_2(S_2)$*, where*

$$S_2 = \begin{bmatrix} 0.7349 & 0.1190 & 0 & 0 & 0.1460 \\ 0.0878 & 0.8145 & 0 & 0 & 0.0977 \\ 0 & 0.1206 & 0.6035 & 0.1602 & 0.1157 \\ 0 & 0.1422 & 0.0848 & 0.7729 & 0 \\ 0.081744 & 0 & 0.1016 & 0.1858 & 0.6309 \end{bmatrix}$$

The matrix S_2 is generated by creating a random matrix in MATLAB. The matrix is scaled and the diagonal entries modified so that the matrix is row stochastic. Its second largest eigenvalue (in magnitude) is $0.73113 \pm 0.080893i$.

Figure 3 shows the results of the estimation. The estimates of b and c are seen to converge to their expected true values in Figure 3a. The dashed vertical lines mark the iterations at which the stopping criterion is met for the different nodes, which occurs when $|x_{i,2}|$ is nearly equal to $\lambda_2(S_2)$. The absolute values of the estimates $x_{i,1}$ and $x_{i,2}$ are seen to converge to that of $\lambda_2(S_2)$ for all five nodes, in Figure 3b.

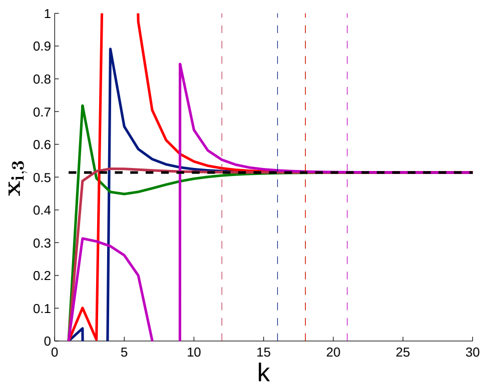
As expected, the estimate $x_{i,3}$ (which is always a real number) does not converge to $\lambda_2(S_2)$ since the latter is complex, as seen in Figure 4.

Example 3. *Estimate* $\lambda_2(S_3)$*, where*

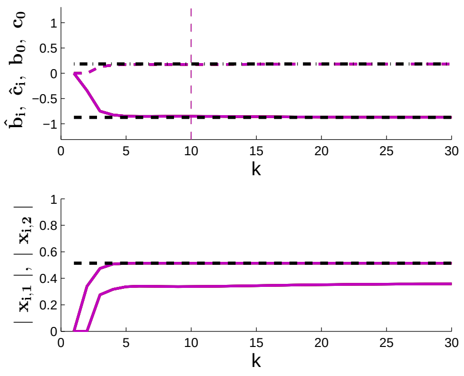
$$S_3 = \begin{bmatrix} 0.521 & 0.163 & 0.122 & 0.045 & 0.147 \\ 0.112 & 0.473 & 0.198 & 0.099 & 0.117 \\ 0.185 & 0.197 & 0.386 & 0.180 & 0.049 \\ 0.139 & 0.000 & 0.095 & 0.631 & 0.133 \\ 0.116 & 0.173 & 0.160 & 0.169 & 0.381 \end{bmatrix}$$

This matrix is generated in the same way as in the previous example. Its second largest eigenvalue is 0.5140. Figure 5a shows the results of the estimation. The estimates $x_{i,3}$ are seen to converge to $\lambda_2(S_3)$ for all five agents.

Since $\lambda_3(S_3)$ is real , $(x_{i,1},\ x_{i,2})$ converges to $(\lambda_2(S_3),\ \lambda_3(S_3))$. This can be seen in Figure 5b. Note that the stopping criterion is met at an earlier iteration for these estimates compared to the standard power iterations for real largest eigenvalues. Since each node is connected to all the remaining nodes, all five estimates are identical in Figure 5b.



(a) Absolute values of $x_{i,3}$ for each node. The dashed black line denotes $|\lambda_2(S_2)|$.



(b) The estimates of b and c for each node (above) and the resulting estimates $x_{i,1}$ and $x_{i,2}$ (below), for S_3.

Fig. 5: Simulation results for the third example.

V. Conclusion

In this paper we propose a decentralized method to estimate $\underline{\lambda}_2(\mathcal{L})$ of a strongly connected graph. The method combines existing decentralized methods to perform power iterations with a method to compute complex dominant eigenvalues of a real matrix. The algorithm requires each agent to communicate a vector whose size increases linearly with the number of agents in the network. The computations performed by each agent at every iteration are inexpensive, unlike other methods.

References

[1] M. De Gennaro and A. Jadbabaie, "Decentralized control of connectivity for multi-agent systems," in *Decision and Control, 2006 45th IEEE Conference on*, Dec 2006, pp. 3628–3633.

[2] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

[3] C. Wu, *Synchronization in Complex Networks of Nonlinear Dynamical Systems.* World Scientific Publishing Company, Incorporated, 2007. [Online]. Available: http://books.google.com/books?id=vMUexcIVnjIC

[4] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[5] R. Aragues, G. Shi, D. Dimarogonas, C. Sagues, and K. Johansson, "Distributed algebraic connectivity estimation for adaptive event-triggered consensus," in *American Control Conference (ACC), 2012*, June 2012, pp. 32–37.

[6] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *Journal of Computer and System Sciences*, vol. 74, no. 1, pp. 70 – 83, 2008, ¡ce:title¿Learning Theory 2004¡/ce:title¿.

[7] M. Franceschelli, A. Gasparri, A. Giua, and C. Seatzu, "Decentralized laplacian eigenvalues estimation for networked multi-agent systems," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, 2009, pp. 2717–2722.

[8] P. Yang, R. Freeman, G. Gordon, K. Lynch, S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity in mobile sensor networks," in *American Control Conference, 2008*, june 2008, pp. 2678 –2683.

[9] A. C. Satici, H. A. Poonawala, H. Eckert, and M. W. Spong, "Connectivity preserving formation control with collision avoidance for nonholonomic wheeled mobile robots," in *International Conference on Intelligent Robots and Systems, to appear in Proceedings of*, 2013.

[10] L. Sabattini, N. Chopra, and C. Secchi, "Decentralized connectivity maintenance for cooperative control of mobile robotic systems," vol. 32, no. 12, pp. 1411–1423, October 2013.

[11] G. H. Golub and C. F. Van Loan, *Matrix computations (3rd ed.).* Baltimore, MD, USA: Johns Hopkins University Press.

[12] C. Li and Z. Qu, "Distributed estimation of algebraic connectivity of directed networks," *Systems & Control Letters*, vol. 62, no. 6, pp. 517 – 524, 2013.

[13] F. Knorn, R. Stanojevic, M. Corless, and R. Shorten, "A framework for decentralised feedback connectivity control with application to sensor networks," *International Journal of Control*, vol. 82, no. 11, pp. 2095–2114, 2009. [Online]. Available: http://dx.doi.org/10.1080/00207170902912056

[14] A. Gourlay and G. Watson, *Computational Methods for Matrix Eigenproblems.* John Wiley & Sons, 1973.

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

[16] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences.* Society for Industrial Mathematics, Jan. 1987.

[17] H. Minc, *Nonnegative matrices.* Technion-Israel Institute of Technology, Dept. of Mathematics, 1974.

Appendix

Analysis of Jacobian in [12]

The method in [12] requires one to find the values of θ_2, ϕ_i and ϕ_l which are roots of the equation $f = 0$, where $f = [f_k, f_{k-1}, f_{k-2}]^T$ and f_k is given by

$$f_k(\theta_2, \phi_i, \phi_l) = \frac{\hat{v}_{2,i}(k+1)\hat{v}_{2,l}(k)}{\hat{v}_{2,l}(k+1)\hat{v}_{2,i}(k)} - \frac{\cos(k\theta_2 + \phi_i)\cos[(k-1)\theta_2 + \phi_l]}{\cos(k\theta_2 + \phi_l)\cos[(k-1)\theta_2 + \phi_i]}$$

The determinant of its Jacobian ∇f can be computed as

$$\begin{aligned}\det(\nabla f) = 8\cos\theta_2 \sec^2(\phi_i + (k-3)\theta_2)\sec(\phi_i \\ + (k-2)\theta_2)\sec(\phi_l + (k-2)\theta_2)\sec(\phi_i \\ + (k-1)\theta_2)\sec(\phi_l + (k-1)\theta_2)\sec^2(\phi_l \\ + k\theta_2)\sin^2(\phi_i - \phi_l)\sin^5\theta_2\end{aligned} \tag{22}$$

which vanishes at several points for any k. Thus, the selection of an appropriate initial condition is critical to being able to find a root. Since the set of points where $\det(\nabla f)$ vanishes is large, the initial condition needs to be close to the solution, which is hard to achieve without prior knowledge.