A Decentralized Algorithm for Balancing a Strongly Connected Weighted Digraph

Attilio Priolo[1], Andrea Gasparri[1], Eduardo Montijano[2] and Carlos Sagues[3]

***Abstract*— In this work we propose a decentralized algorithm for balancing a strongly connected weighted digraph. This algorithm relies on the decentralized estimation of the left eigenvector associated to the zero structural eigenvalue of the Laplacian matrix. The estimation is performed through the distributed computation of the powers of the Laplacian matrix itself. This information can be locally used by each agent to modify the weights of its incoming edges so that their sum is equal to the sum of the weights outgoing this agent, i.e., the weighted digraph is balanced. Simulation results are proposed to corroborate the theoretical results.**

I. Introduction

In the past decades, multi-agent systems have gained an increasing interest within the control theory community. A multi agent system consists of a set of agents which collaborate for a common objective. It can be effectively used to model for instance wireless sensor networks, teams of robots or social dynamics, [1]–[3]. Several interesting problems arise in the context of multi-agent systems, ranging from distributed estimation [4] to collaborative data fusion [5].

A common assumption in the multi-agent literature is to model the communication among the agents as an undirected communication graph (see [6], [7] and the references therein). This is founded on the assumptions that the communication is isotropic, i.e., the employed antenna is supposed to have the same radiation in all directions and that its range is the same for all the agents in the network. Therefore, if an agent can communicate to another one, the opposite is possible as well. However, these assumptions might not be adequate for some application scenarios due to several phenomena as varying interference and different power transmission levels.

In this work we propose a decentralized algorithm for balancing a strongly connected weighted digraph (SCWD). In particular, first we introduce a technique to estimate the left eigenvector associated to the zero eigenvalue of the Laplacian matrix encoding the communication digraph. Then we discuss how this information, locally available to each agent, can be used by the agents themselves to independently adjust the weights of their incoming edges without any prior knowledge of the out-neighborhood so that the weighted digraph is balanced.

To the best of the authors knowledge, little is available in the literature regarding the estimation of the left eigenvector associated with the zero eigenvalue of the Laplacian matrix for digraphs. The work by Qu et al. [8] possibly represents the best approach in this field. In this work, the authors propose a distributed approach for the estimation of the left eigenvector associated to the weight matrix and the expected consensus value. The estimate is then used in a control algorithm to improve the network convergence rate. However, the continuous time nature of the algorithm makes its implementation in a real world scenario particularly challenging.

In the context of cooperative control, where the agents are required to share information among themselves in order to achieve a common goal [9]–[12], the consensus algorithm represents an effective solution. The key idea is to let each agent update its value by performing a weighted linear combination of the previous values of its one-hop neighbors and its previous value in order to achieve an agreement. The Laplacian matrix represents the state matrix of the continuous time consensus protocol [13], [14]. In the case of undirected graphs, the Laplacian matrix is symmetric and sums up to zero both on the rows and the columns. This statement does not hold in the case of general digraphs due to the non-symmetric nature of their adjacency matrices.

Recently, a few works have appeared dealing with the problem of weight balancing a SCWD. In [15], the authors introduce two algorithms, a centralized and a decentralized one, to balance a weighted strongly connected digraph. However, within this work, the out-neighborhood of each agent is assumed to be explicitly known. In [16] a distributed algorithm to make the adjacency matrix doubly stochastic is presented. It relies on balancing the out and the in degrees of each node by adjusting the weights on the outgoing edges. However, this approach requires each agent to be able to send individual messages to the agents belonging to its out-neighborhood. Thus, a specific mechanism for the identification of the out-neighborhood is required. In [17], a doubly stochastic weight matrix is computed by an iterative procedure that adjusts the outgoing weights of each node. In fact, each agent computes a linear combination of its value and the ones from its neighbors. This approach relies on the assumption that each agent is able to choose a time invariant

[1] A. Priolo and A. Gasparri are with the Department of Computer Science and Automation (DIA), University of Rome “Roma Tre”, Italy. {gasparri,priolo}@dia.uniroma3.it

[2] E. Montijano is with Centro Universitario de la Defensa (CUD) and Instituto de Investigación en Ingeniería de Aragón (I3A), Zaragoza, Spain. emonti@unizar.es

[3] C. Sagues is with Departamento de Informática e Ingeniería de Sistemas - Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Spain. csagues@unizar.es

This work was partially supported by the Italian grant FIRB “Futuro in Ricerca”, project NECTAR “Networked Collaborative Team of Autonomous Robots”, code RBFR08QWUV, funded by the Italian Ministry of Research and Education (MIUR) and by the Spanish project Ministerio de Economía y Competitividad DPI2009-08126 and DPI2012-32100.

weight which sums to one together with the time invariant weights of its out-neighbors. In other terms, each agent is required to know its out-degree.

In this work, we release the assumption of having any prior knowledge of the out-neighborhood of an agent. This paves the way for the application of algorithms based on weight balanced digraphs for which the adopted communication scheme forbids to collect any information about the out-going edges.

Summarizing, the main contributions of this work are the following:

- A novel distributed algorithm for the estimation of the left eigenvector associated to the zero structural eigenvalue of the Laplacian Matrix of a SCWD.
- A procedure to let each agent adjust the weights of its incoming edges so that the resulting SCWD is balanced.

The rest of the paper is organized as follows. In Section II some preliminary tools coming from the algebraic graph theory are reviewed. In Section III the proposed algorithm for the estimation of the left eigenvector of a Laplacian matrix encoding a SCWD is described. In Section IV, a procedure to balance the SCWD is introduced. In Section V, some simulations to corroborate the theoretical results are provided. Finally, in Section VI conclusions are drawn.

II. Preliminaries

A. Notation

Let us consider a set of n agents whose communication network is described by a *digraph* $\mathcal{G}(\mathcal{V},\mathcal{E})$ where $\mathcal{V}=\{1,\cdots,n\}$ is the set of nodes and $\mathcal{E}\subseteq\mathcal{V}\times\mathcal{V}$ is the set of directed edges, i.e., ordered pairs of nodes. Let us define the *weighted adjacency matrix* $\mathcal{A}(\mathcal{G})\in\mathbb{R}^{n\times n}$ as follows:

$$\mathcal{A}_{ij}(\mathcal{G})=\begin{cases}\mathcal{A}_{ij}(\mathcal{G})>0 & \text{if } (j,i)\in\mathcal{E},\\ 0 & \text{otherwise.}\end{cases} \tag{1}$$

Note that $\mathcal{A}_{ij}(\mathcal{G})>0$ if the agent i can receive data from the agent j. In general, the existence of $(j,i)\in\mathcal{E}$ does not imply the existence of $(i,j)\in\mathcal{E}$. It is assumed that no self loops exist in the network, i.e., $(i,i)\notin\mathcal{E}$. The in-degree and the out-degree of a node k are given by $d^{\text{in}}(k)=\sum_j\mathcal{A}_{kj}(\mathcal{G})$ and $d^{\text{out}}(k)=\sum_j\mathcal{A}_{jk}(\mathcal{G})$ respectively. The *Laplacian matrix* is defined as follows:

$$\mathcal{L}(\mathcal{G})=\mathcal{D}(\mathcal{G})-\mathcal{A}(\mathcal{G}),$$

with $\mathcal{D}(\mathcal{G})$ the *diagonal in-degree matrix* defined as

$$\mathcal{D}(\mathcal{G})=\begin{bmatrix} d^{\text{in}}(1) & & \\ & \ddots & \\ & & d^{\text{in}}(n)\end{bmatrix},$$

and $\mathcal{D}_{ij}(\mathcal{G})=0$ whenever $i\neq j$. For the sake of readability, the dependency on the graph $\mathcal{G}$ will be omitted in the rest of the paper. Let us now review some important properties of the Laplacian matrix for a directed graph. Generally speaking, the Laplacian matrix is a non-symmetric weakly diagonal dominant matrix. It has a zero structural eigenvalue for which the corresponding right eigenvector is the vector of ones of appropriate size, i.e., $\mathcal{L}\mathbf{1}=\mathbf{0}$. Let us denote with $\mathcal{Q}\geq 0$, a nonnegative matrix, i.e., a matrix for which all elements are strictly nonnegative. Note that, a nonnegative matrix is not the same as a positive semidefinite matrix.

The following definitions will be used throughout the rest of the paper. For further details and proofs, the reader is referred to [18].

Definition 2.1: A digraph is called *strongly connected* if and only if any two distinct nodes in the vertex set can be connected by a path which follows the direction of the edges of the digraph.

Definition 2.2: A weighted digraph is said to be *balanced* if:

$$d^{\text{in}}(k)=d^{\text{out}}(k),\ \forall k\in\{1,\ldots,n\}\,.$$

Note that, this definition can be equivalently expressed in terms of the Laplacian matrix as follows: $\mathcal{L}\mathbf{1}=\mathbf{0}$ and $\mathbf{1}^T\mathcal{L}=\mathbf{0}^T$.

Definition 2.3: A non negative matrix $\mathcal{Q}\in\mathbb{R}^{r\times r}$, with $r\geq 2$, is *reducible* if there exists a permutation matrix $P\in\mathbb{R}^{r\times r}$ such that $P^T\mathcal{Q}P$ is an upper triangular matrix. Otherwise the matrix is said to be *irreducible*.

The link between the definition of an irreducible matrix and digraphs is given by the following proposition.

Proposition 2.1: The adjacency matrix $\mathcal{A}$ is an irreducible matrix if and only if its associated graph $\mathcal{G}$ is strongly connected.

Definition 2.4: A nonnegative matrix $\mathcal{Q}$ is said to be *primitive* if there exist a positive integer k s.t. $\mathcal{Q}^k>0$.

In the following, a proposition representing a sufficient condition for a matrix to be primitive is reported:

Proposition 2.2: If a nonnegative irreducible matrix $\mathcal{Q}$ has at least one positive diagonal element, then $\mathcal{Q}$ is primitive.

Eventually, from the Perron-Frobenius theorem it follows that for any primitive matrix $\mathcal{Q}$ the following holds:

$$\lim_{k\to\infty}\left(\frac{\mathcal{Q}}{\rho}\right)^k=\frac{v\,w^T}{w^T\,v}>0. \tag{2}$$

where v is the right eigenvector and w^T is the left eigenvector both associated with the spectral radius ρ.

B. Assumptions

Let the following assumptions be satisfied throughout the rest of the paper:

A1 The network topology of the considered multi-agent system is described by a static SCWD.

A2 The knowledge of the out-degree is not required by the agents.

A3 A unique identifier is associated to each agent i of the network, e.g., the IP address.

A4 Each agent can store n variables.

In **A1**, we assume that there is always a path from an agent towards another one. This allows the information to flow among all the agents. In **A2**, it is stated that each agent is not required to know the number of its out neighbors. In **A3**, we assume that each agent can distinguish the information coming from the other agents according to the identifier of the sender. Eventually, in **A4** it is assumed that each agent has enough storage size for the values coming from its in-neighbors.

III. Left Eigenvector Estimation

A. Decentralized Estimation of the Left Eigenvector

In this section, a decentralized algorithm for the estimation of the left eigenvector of a SCWD is proposed. To this aim, a distributed algorithm for the computation of the powers of a matrix is reviewed. Further details can be found in [19], [20].

Let us now introduce the concept of matrix compatibility with a digraph.

Definition 3.1: A matrix $\mathcal{Q} \in \mathbb{R}^{n\times n}$ is *compatible* with a digraph $\mathcal{G}$ if $\mathcal{Q}_{ij} = 0$ iff $(j,i) \notin \mathcal{E}$ and $j \neq i$.
Loosely speaking, this definition of compatibility guarantees that $\mathcal{Q}_{ij}$ is equal to 0 whenever the agent i cannot receive any data from the agent j.

Let each agent i have a variable $\delta_i(k) = [\delta_{i1}(k) \ \dots \ \delta_{in}(k)]^T$ with initial values

$$\delta_{ij}(0) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

and let $\mathcal{Q} = [\mathcal{Q}_{ij}]$ be a weighted matrix which is *compatible* with the communication graph. At each iteration, the agents update their variables as follows:

$$\delta_{ij}(k+1) = \sum_{j' \in \mathcal{N}_i \cup i} \mathcal{Q}_{ij'} \delta_{j'j}(k), \tag{4}$$

with $\mathcal{N}_i$ the in-neighborhood of agent i. Update rule (4) can be put in vectorial form as

$$\Delta(k+1) = \mathcal{Q}\Delta(k),$$

with:

$$\Delta(k) = \begin{bmatrix} \delta_1(k)^T \\ \vdots \\ \delta_n(k)^T \end{bmatrix}.$$

Let us denote with $\mathcal{I} \in \mathbb{R}^{n\times n}$ the identity matrix. Noting that $\Delta(0) = \mathcal{I}$, it is easy to see that at iteration k, the variable $\delta_i(k)$ contains exactly the value of the i^{th} row of the matrix $\mathcal{Q}^k$. Let us remark that the algorithm is fully distributed, in the sense that each agent only uses its previous value and the data sent by its in-neighbors to update the variable $\delta_i(k)$.

For the estimation of the left eigenvector associated to the zero eigenvalue of the Laplacian, the following matrix $\mathcal{C}$ compatible with the graph is used:

$$\mathcal{C} = \mathcal{I} - \beta\,\mathcal{L}, \quad 0 < \beta < \frac{1}{\Psi}, \tag{5}$$

where $\Psi = \max_i\{\sum_{j\neq i} \mathcal{A}_{ij}\}$. The matrix $\mathcal{C}$ is commonly referred to as the *Perron matrix*.

The eigenvalues $\lambda_{\mathcal{C}}$ of the matrix $\mathcal{C}$ and the eigenvalues of the Laplacian matrix $\lambda_{\mathcal{L}}$ are related as follows:

$$\lambda_{\mathcal{C}_i} = 1 - \beta\,\lambda_{\mathcal{L}_i},$$

where $\lambda_{\mathcal{C}_i}$ and $\lambda_{\mathcal{L}_i}$ are the i^{th} eigenvalues of the $\mathcal{C}$ and $\mathcal{L}$ matrices respectively. It follows that the two matrices also share the same set of eigenvectors. In particular for the eigenvalue of maximum modulus, namely $\lambda_{\mathcal{C}_1}$, the following holds:

$$\mathcal{C}\,\mathbf{1} = \lambda_{\mathcal{C}_1}\mathbf{1},$$
$$w^T\,\mathcal{C} = \lambda_{\mathcal{C}_1} w^T,$$

with w^T the left eigenvector associated to $\lambda_{\mathcal{C}_1}$ and $\lambda_{\mathcal{L}_1}$. The following proposition proves the convergence to the normalized left eigenvector:

Proposition 3.1: Assume that an initial value $\delta_{ij}(0)$, as defined in (3), is associated to each agent i. Then if the agents apply the update rule in (4) using the compatible with the graph matrix $\mathcal{C}$, as defined in (5), then

$$\lim_{k\to\infty} \Delta(k) = \frac{\mathbf{1}\ w^T}{w^T\mathbf{1}} \tag{6}$$

or, in other terms, $\delta_i(k)$ will tend to the normalized left eigenvector w of the Laplacian matrix encoding the digraph.

Proof: Using Proposition 2.2, it is possible to state that the matrix $\mathcal{C}$ is primitive. Then, the proof follows from the application of the Perron-Frobenius theorem reported in Section II-A. ■

Algorithm 1 shows the pseudo-code of the left eigenvector estimation algorithm. Note that the algorithm requires the digraph to be strongly connected. This ensures the Perron matrix built according to (5) to be irreducible and diagonally positive. Regarding the pseudo-code, the variable max_iter represents the predefined maximum value of the algorithm's iterations. The matrix $\Delta(max\_iter)$ is the output of the algorithm; each row i represents the estimation performed by the i^{th} agent. Line 1 describes the variables initialization. Each agent sets its corresponding component to 1 and the other components to 0. In the second step, the Perron matrix is computed according to its definition in (5). Lines 3-5 represent an implementation of the update rule introduced in (4).

Algorithm 1 Left Eigenvector Estimation Algorithm

```
Require: $\mathcal{G}$ SCWD, $\mathcal{L}$, $0 < \beta < \frac{1}{\Psi}$, $max\_iter$
Ensure: $\Delta_i(max\_iter) \to \mathbf{1w^T}$
1: $\delta_{ij}(0) = 1$ if $i = j$, $\delta_{ij}(0) = 0$ otherwise
2: $\mathcal{C} \leftarrow \mathcal{I} - \beta\mathcal{L}$
3: for $k = 0, \dots, max\_iter$ do
4:         $\delta_{ij}(k+1) = \sum_{j' \in \mathcal{N}_i \cup i} \mathcal{C}_{ij'} \delta_{j'j}(k)$
5: end for
```

IV. Balancing the Strongly Connected Weighted Digraph

In this section, we describe how each agent can use its estimate of the left eigenvector to modify the weights of the incoming edges to make the SCWD balanced. To this end, let us consider a SCWD composed of n nodes and let us denote with $\mathcal{L} \in \mathbb{R}^{n\times n}$ the related Laplacian matrix defined as:

$$\mathcal{L} = \begin{bmatrix} l_{11} & l_{12} & \dots & l_{1n} \\ l_{21} & l_{22} & \dots & l_{2n} \\ \vdots & & & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{bmatrix}. \tag{7}$$

Let us recall that by definition the rows of the Laplacian matrix $\mathcal{L}$ sum to zero, that is

$$\sum_{j=1}^{n} l_{ij} = 0, \quad i = 1, \dots, n.$$

Let w be the left eigenvector related to the zero eigenvalue $\lambda_{\mathcal{L}_1} = 0$. Therefore by definition we have:

$$w^T \mathcal{L} = \lambda_{\mathcal{L}_1} w^T = \mathbf{0}^T,$$

which can be written with respect to each column $l_i = [l_{1i} \;\; l_{2i} \;\; \dots \;\; l_{ni}]^T$ of the Laplacian matrix as:

$$w_1 l_{1i} + w_2 l_{2i} + \dots + w_n l_{ni} = 0.$$

Let us now introduce a modified Laplacian matrix $\bar{\mathcal{L}}$ where the i^{th} row is obtained by multiplying the i^{th} row of the original Laplacian matrix by the i^{th} component of the left eigenvector w, as follows:

$$\bar{\mathcal{L}} = \begin{bmatrix} w_1 l_{11} & w_1 l_{12} & \dots & w_1 l_{1n} \\ w_2 l_{21} & w_2 l_{22} & \dots & w_2 l_{2n} \\ \vdots & & & \vdots \\ w_n l_{n1} & w_n l_{n2} & \dots & w_n l_{nn} \end{bmatrix}. \tag{8}$$

Note that this matrix can always be obtained in a distributed fashion as each row i can be computed by the i^{th} agent once the estimation of the i^{th} coefficient of the left eigenvector is available to it. Recall that the i^{th} row is simply the set of coefficients used by the i^{th} agent to weight the incoming edges from its neighbors.

Proposition 4.1: The Laplacian matrix $\bar{\mathcal{L}}$ sums to zero also on the columns, that is:

$$\mathbf{1}^T \bar{\mathcal{L}} = \mathbf{0}^T.$$

Proof: for each column i of the Laplacian matrix $\bar{\mathcal{L}}$:

$$\begin{aligned} \mathbf{1}^T \bar{l}_i &= \bar{l}_{1i} + \bar{l}_{2i} + \dots + \bar{l}_{ni} \\ &= w_1 l_{1i} + w_2 l_{2i} + \dots + w_n l_{ni} \\ &= 0 \end{aligned}$$

where $\bar{l}_i$ is the i-th column of the matrix $\bar{\mathcal{L}}$. ■

Let us point out again the equivalence between the result of Proposition 4.1 and the definition of balancedness for a weighted digraph given in Definition 2.2. Note that the result stated in Proposition 4.1 could be equivalently derived by proving that the hypothesis required in [16, Theorem 3.2] hold true. However, Proposition 4.1 suffices for the objective of this paper and does not burden the reader with additional details. Algorithm 2 shows the pseudo-code of the algorithm to let the Laplacian matrix of any strongly connected digraph sum up to zero both on the rows and the columns. The input for the algorithm is the number of agents n, the Laplacian matrix $\mathcal{L}$ and the Perron matrix $\mathcal{C}$. Note that if n is not available to the agents, a labeling technique as the one in [20] can be used. The result of this algorithm is $\bar{\mathcal{L}}$ as defined in (8). Line 1 describes how the computation of the powers of the Perron matrix are carried out in a distributed fashion exploiting the algorithm given in Section III-A. Lines 2-4 show the computation of the i^{th} row of the modified Laplacian matrix $\bar{\mathcal{L}}$ locally carried out by the i^{th} agent.

Algorithm 2 Weight Balancing the SCWD

```
Require: $n, \mathcal{L}, \mathcal{C}$
Ensure: $\bar{\mathcal{L}}\mathbf{1} = \mathbf{0}$,   $\mathbf{1}^T\bar{\mathcal{L}} = \mathbf{0}^T$
1: Compute $\mathcal{C}^{max\_iter}$ using Algorithm 1.
2: for $i = 1 \dots n$ do
3:     $\bar{\mathcal{L}}_i = \mathcal{L}_i \cdot C_{ii}^{max\_iter}$
4: end for
```

V. Simulations

In this section two simulations with an increasing number of agents are proposed. In each simulation, all the agents first estimate the eigenvector associated to the zero eigenvalue of the Laplacian matrix, then perform a consensus using the modified Laplacian matrix. Let us assume that an initial value $x_i(0) \in \mathbb{R}$ is associated to each agent and $x_i(0)$ to be a scalar for the sake of simplicity and without loss of generality. Let us also denote with $x(0) = [x_1(0) \dots x_n(0)]^T$ the vector containing all the initial values associated to the agents. It is worthy to point out that using the modified Laplacian matrix, the consensus value coincides with the average of the initial conditions, i.e., $\mu = \frac{1}{n}\sum_i x_i(0)$, while using the original Laplacian it converges to $\mu_1 = \sum_i w_i x_i(0)$. The first simulation involves 6 six agents performing the average consensus algorithm after the estimation process of the left eigenvector of the Laplacian matrix. The second simulation involves 20 agents. In this case, only the estimation of the left eigenvector performed by a single agent is shown along with the iterations of the consensus algorithm. Let us consider the network topology to be described by the graph in Fig. 2(a). It is composed by six agents and the corresponding directed edges. The initial conditions of the agents are:

$$x(0) = [43 \;\, 93 \;\, 95 \;\, 6 \;\, 31 \;\, 80]^T,$$

while $\mu = 58$. In Fig. 1 the estimation process of the left eigenvector associated to the zero eigenvalue of the Laplacian matrix is illustrated. For every agent, the estimation process of each element of the vector is plotted. It can be noticed that the agents asymptotically achieve the same values. In Fig. 2(b) the execution of the consensus algorithm with the original Laplacian matrix in (7) is shown. It is worthy to notice that in this case the agents can achieve a consensus because the graph is a SCD, but the consensus value is different from μ. Instead, in Fig. 2(c) the execution of the consensus protocol using the modified Laplacian matrix is given. In this case, all the agents obtain μ as their consensus value. The same simulation, but considering 20 agents, is depicted in Fig. 3 and the following random initial conditions are used:

$$x(0) = [99\ 78\ 4\ 33\ 89\ 25\ 61\ 12\ 68 \\ 81\ 8\ 4\ 89\ 91\ 94\ 24\ 67\ 61\ 99\ 86]^T,$$

while $\mu = 58.65$. Note that also in this case the average consensus is reached. In Fig. 4, the corresponding left eigenvector estimation process is illustrated for a single agent. The agents achieve the same values of the estimated components of the left eigenvector w^T. Multiplying each row of the original Laplacian matrix by the corresponding element of the estimated left eigenvector leads to the executions of the consensus protocol in Fig. 3(c).

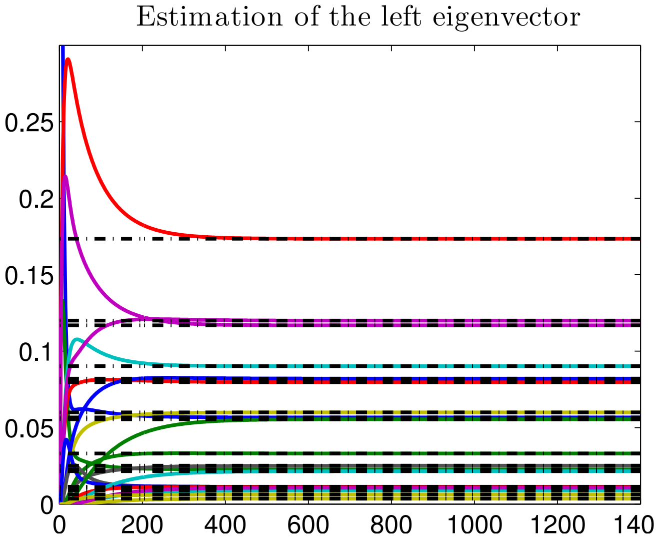


Fig. 4. Left eigenvector estimation in the case of 20 agents.

VI. Conclusion

In this work a decentralized algorithm to estimate the left eigenvector associated to the zero eigenvalue of the Laplacian matrix has been presented. In particular, first an algorithm for the computation of the powers of the Laplacian matrix which converges to a matrix where each row is the aforementioned left eigenvector has been proposed; successively a local policy to let each agent independently adjust the weights of its incoming edges so that the resulting weighted digraph is balanced has been described. A theoretical analysis of the proposed algorithm has been carried out along with simulations to corroborate the theoretical results. Future work will be mainly focused on the definition of a necessary and sufficient condition for the verification of the strongly connectivity of the digraph. Noting that the proposed approach requires the estimation algorithm and the digraph balancing policy to run sequentially, a modified version of this approach where the algorithm can run in parallel is currently under investigation.

References

[1] E. Montijano, J. Thunberg, X. Hu, and C. Sagues, “Multi-robot distributed visual coordination using epipoles,” in *50th IEEE Conference on Decision and Control and European Control Conference*, December 2011.

[2] E. Garone, A. Gasparri, and F. Lamonaca, “Clock synchronization for wireless sensor network with communication delay,” in *American Control Conference (ACC)*, 2013.

[3] R. Hegselmann and U. Krause, “Opinion dynamics and bounded confidence, models, analysis and simulation,” *Journal of Artificial Societies and Social Simulation*, vol. 5, no. 3, pp. 1–33, 2002.

[4] S. Stankovic, M. Stankovic, and D. Stipanovic, “Decentralized parameter estimation by consensus based stochastic approximation,” *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 531–543, March 2011.

[5] A. Gasparri, F. Fiorini, M. Di Rocco, and S. Panzieri, “A networked transferable belief model approach for distributed data aggregation,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 391 – 405, April 2012.

[6] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*, ser. Princeton Series in Applied Mathematics. Princeton University Press, 2010.

[7] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*, ser. Applied Mathematics Series. Princeton University Press, 2009, electronically available at http://coordinationbook.info.

[8] Z. Qu, C. Li, and F. Lewis, “Cooperative control based on distributed estimation of network connectivity,” in *American Control Conference*, June 2011, pp. 3441 – 3446.

[9] Z. Qu, J. Wang, and R. A. Hull, “Cooperative control of dynamical systems with application to autonomous vehicles,” *IEEE Transactions on Automatic Control*, vol. 53, no. 4, pp. 894–911, 2008.

[10] A. Jadbabaie, J. L. J. Lin, and A. S. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,” *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[11] 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, no. 9, pp. 1520–1533, 2004.

[12] E. Montijano, J. I. Montijano, and C. Sagues, “Chebyshev Polynomials in Distributed Consensus Applications,” *IEEE Transactions on Signal Processing*, vol. 61, no. 3, pp. 693–706, March 2013.

[13] B. Mohar, “The laplacian spectrum of graphs,” *Graph Theory Combinatorics and Applications*, vol. 2, pp. 871–898, 1991.

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

[15] C. Hadjicostis and A. Rikos, “Distributed strategies for balancing a weighted digraph,” in *20th Mediterranean Conference on Control Automation (MED)*, 2012, pp. 1141 –1146.

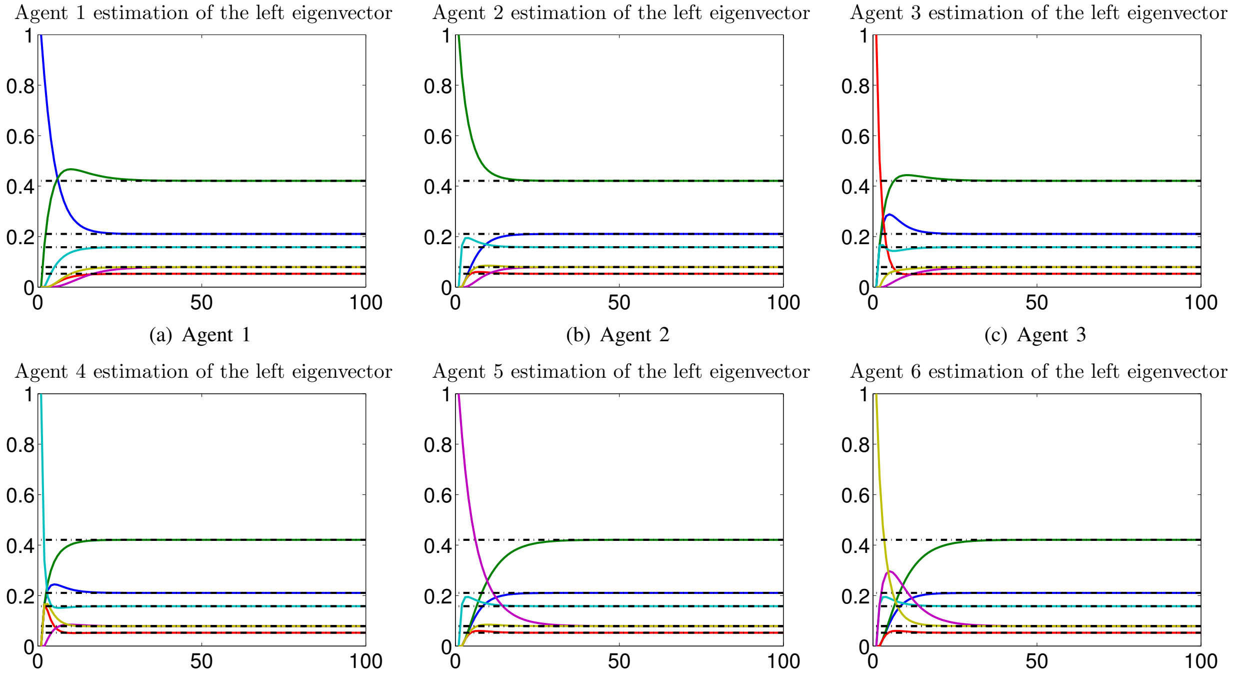
[16] B. Gharesifard and J. Cortes, “Distributed strategies for generating weight-balanced and doubly stochastic digraphs,” *European Journal of Control*, vol. 18, no. 6, pp. 539–557, 2012.

[17] A. D. Dominguez-Garcia and C. N. Hadjicostis, “Distributed strategies for average consensus in directed graphs,” *IEEE Conference on Decision and Control and European Control Conference*, pp. 2124–2129, December 2011.

[18] C. D. Meyer, Ed., *Matrix analysis and applied linear algebra.* Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000.

[19] R. Aragues, E. Montijano, and C. Sagues, “Consistent data association in multi-robot systems with limited communications,” in *Robotics: Science and Systems*, June 2010.

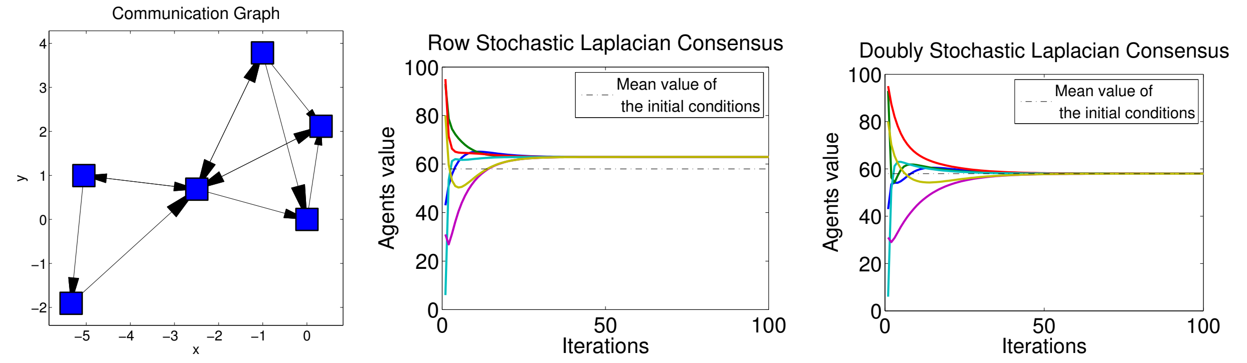
[20] R. Aragues, G. Shi, D. V. Dimarogonas, C. Sagues, , and K. H. Johansson, “Distributed algebraic connectivity estimation for adaptive event-triggered consensus,” in *American Control Conference*, 2012.



(a) Agent 1 (b) Agent 2 (c) Agent 3

(d) Agent 4 (e) Agent 5 (f) Agent 6

Fig. 1. Left Eigenvector Estimate for the 6 agents.

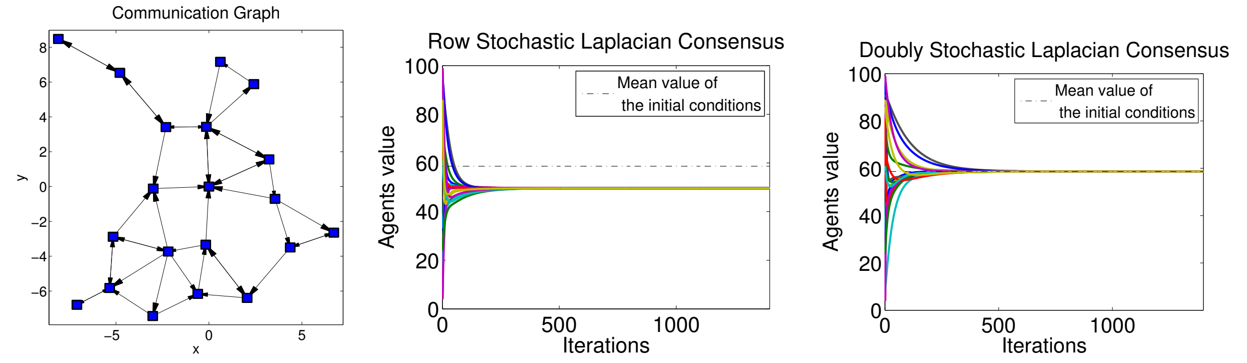


(a) The underlying communication graph describing the interaction among the six agents.

(b) Consensus protocol using the original Laplacian matrix with six agents. The dotted line represents the average value.

(c) Consensus protocol using the modified Laplacian matrix with six agents. The dotted line represents the average value.

Fig. 2. Iterations of the consensus protocol both with the original Laplacian matrix and the modified one with six agents.



(a) The underlying communication graph describing the interaction among the twenty agents.

(b) Consensus protocol using the original Laplacian matrix with twenty agents. The dotted line represents the average value.

(c) Consensus protocol using the modified Laplacian matrix with twenty agents. The dotted line represents the average value.

Fig. 3. Iterations of the consensus protocol both with the original Laplacian matrix and the modified one with twenty agents.