Distributed Estimation of Graph Laplacian Eigenvalues by the Alternating Direction of Multipliers Method

Thi-Minh Dung Tran, Alain Y. Kibangou

Distributed Estimation of Graph Laplacian Eigenvalues by the Alternating Direction of Multipliers Method

Thi Minh Dung Tran, Alain Y. Kibangou

Gipsa-Lab, CNRS, University Joseph Fourier,
11 rue des Mathématiques, Grenoble Campus, France.
(e-mail: thi-minh-dung.tran@gipsa-lab.fr, alain.kibangou@ujf-grenoble.fr).

Abstract: This paper presents a new method for estimating the eigenvalues of the Laplacian matrix associated with the graph describing the network topology of a multi-agent system. Given an approximate value of the average of the initial condition of the network state and some intermediate values of the network state when performing a Laplacian-based average consensus, the estimation of the Laplacian eigenvalues is obtained by solving the factorization of the averaging matrix. For this purpose, in contrast to the state of the art, we formulate a convex optimization problem that is solved in a distributed way by means of the Alternating Direction Method of Multipliers (ADMM). The main variables in the optimization problem are the coefficients of a polynomial whose roots are precisely the inverse of the distinct nonzero Laplacian eigenvalues. The performance of the proposed method is evaluated by means of simulation results.

Keywords: Graph Laplacian, Eigenvalues, Distributed optimization, Alternating Direction of Multipliers Method, Average Consensus.

1. INTRODUCTION

Investigating the relationship between the structure of a graph and its eigenvalues is the central topic in the field of algebraic graph theory, Merris (1994); Chung (2012). In particular, the spectrum of the Laplacian matrix has a direct connection to the behavior of several networked dynamical processes. For example, the second smallest eigenvalue of a Laplacian matrix, i.e., the graph algebraic connectivity of the graph, Friedler (1973), is known to have the main role in the convergence time of various distributed algorithms. It is also a critical parameter that influences on the performance and robustness properties of dynamical systems operating over an information network. From the spectrum of the graph Laplacian matrix, we can also infer bounds on the graph diameter and state the connectedness of the graph, Friedler (1973). Moreover, it can be utilized for checking the controllability and observability of the system, Franceschelli et al. (2010). A comprehensive survey on properties of Laplacian of undirected graphs can be found in Merris (1994).

In the recent literature devoted to multi-agent dynamic systems, several issues are formulated as a consensus problem, which is to design a network protocol based on the local information obtained by each agent, such that all agents finally reach an agreement on certain quantities of interest. The network protocol is an interaction rule, which ensures that the whole group can achieve a consensus on the shared data in a distributed manner, i.e. without the coordination of a central authority. In the study of consensus problems, the speed of convergence is usually an important index to evaluate the efficiency of the corresponding protocol. For networks modeled with undirected graphs, when using Laplacian-based consensus matrices, also known as constant edge weights protocol, it has been shown that, for speeding up the convergence in an average consensus problem, the optimal consensus matrix depends on both the largest and the smallest nonzero Laplacian eigenvalues, Xiao and Boyd (2004). More recently, it has been shown that all the spectrum of the Laplacian matrix can also be used for designing consensus matrices in order to achieve the average consensus in a finite number of steps, Kibangou (2011, 2012). Such a conclusion is based on the fact that the averaging matrix, $\mathbf{J}_N = \frac{1}{N}\mathbf{1}\mathbf{1}^T$, can be written as a product of Laplacian-based consensus matrices parameterized by the inverse of the Laplacian eigenvalues. Therefore, computing the eigenvalues of the Laplacian matrix is an important issue and solving this problem in a distributed way is particularly challenging.

During the current decade, various studies have been carried out on decentralized algorithms for estimating the Laplacian eigenvalues. For instance, in Yang et al. (2008), the second smallest Laplacian eigenvalue was estimated by resorting to a decentralized power iteration method. In Aragues et al. (2012), the algebraic connectivity was estimated by counting on the distributed computation of the powers of a deflated Laplacian matrix. In Sahai et al. (2012) and Franceschelli et al. (2013), Fast Fourier Transform (FFT)-based methods were suggested. The main idea in these works is to make the state of each agent oscillate only at frequencies corresponding to the eigenvalues of the Laplacian matrix associated with the network topology. The problem is then mapped into a signal processing one that can be efficiently and independently solved by each agent in applying the Fast Fourier Transform (FFT). The approach in Franceschelli et al. (2013) involved twice communication burden compared to that in Sahai et al. (2012). However, both methods inherit the limitations of the FFT-based algorithm. In particular, the resolution of the estimated eigenvalues is strongly dependent on that of the FFT-based method and the

accuracy depends on the amount of stored data. In contrast, in Kibangou and Commault (2012), the authors resort to an algebraic method using observability properties of the network. With this method, the eigenvalues of the network matrix can be recovered by solving a local eigenvalue decomposition on an appropriately constructed matrix of observed data. In addition, this approach also provides multiplicities of Laplacian eigenvalues. However, this method is only applicable to networks having nodes with sufficient storage and computation capabilities. In Tran and Kibangou (2013), the authors have proposed a method for estimating the distinct nonzero Laplacian eigenvalues by solving the factorization of the averaging matrix as a product of Laplacian-based consensus matrices. Precisely, an equality constrained consensus problem, which turns to be a non-convex optimization problem, was formulated. A gradient backpropagation method was proposed. Despite the efficiency of that method, only convergence towards local minima can in fact be guaranteed. In addition, speed of convergence was particularly slow. It is worth noting that the authors assumed that the number of the distinct nonzero Laplacian eigenvalues was known, which is a strong assumption fulfilled by few families of graphs such as strongly regular graphs (two nonzero distinct Laplacian eigenvalues) and distance regular graph (the number of nonzero distinct Laplacian eigenvalues equals the diameter of the graph), Kibangou (2014).

In this paper, we propose a novel method to deal with the distributed estimation of the Laplacian eigenvalues. Herein, the number of distinct Laplacian eigenvalues is unknown. Furthermore, the proposed method is shown to be efficient for solving the problem when the exact average value is not known. The proposed solution results on solving a convex optimization problem in a distributed way. The Alternating Direction of Multipliers Method (ADMM), that has received a great attention from the community due to its easy implementation, fast convergence, and good accuracy, is adopted to solve the problem, Boyd et al. (2011); Erseghe et al. (2011).

The remainder of this paper is organized as follows: in Section 2, we recall some properties of the graph Laplacian and its connections with the average consensus problem. Then, the proposed method is described in Section 3 for solving a constrained optimization problem. The performance of the proposed method is evaluated in Section 4 by means of simulation results before concluding the paper.

2. LAPLACIAN SPECTRUM AND AVERAGE CONSENSUS

2.1 Average consensus

Consider a network of N agents characterized by local values $x_i \in \Re$, $i = 1, \cdots, N$. The interactions between these agents are modeled by means of a connected undirected graph $G(V,E)$, where V and E denote the node set (vertex set) and the link set (edge set) respectively. Average consensus algorithms can be seen as the distributed solution of an optimization algorithm whose goal is the minimization of the disagreement between the nodes in a given network, Olfati-saber et al. (2010). In other words, average consensus resorts to minimizing the cost function

$$\Phi_G(\mathbf{x}) = \mathbf{x}^T \mathbf{L}\mathbf{x} = \frac{1}{2} \sum_{(i,j)\in E} (x_j - x_i)^2, \tag{1}$$

where $\mathbf{x} = [x_1, x_2, ..., x_N]^T$ denotes the network state whereas $\mathbf{L}$ stands for the Laplacian matrix, which is a double stochastic matrix whose entries are given by $l_{ii} = d_i$, $l_{ij} = -1$ if $(i,j) \in E$, and $l_{ij} = 0$ elsewhere, where d_i stands for the degree of agent i, i.e. the number of agents that are adjacent to agent i.

Starting from an initial state $x_i(0) = x_i$ and using a steepest descent method, the following linear iteration scheme is obtained:

$$x_i(t) = x_i(t-1) - \alpha \sum_{j\in N_i} (x_i(t-1) - x_j(t-1)), \tag{2}$$

where $N_i = \{j \in V : (i,j) \in E\}$ denotes the neighborhood of node i. In matrix form, we get:

$$\mathbf{x}(t) = (\mathbf{I}_N - \alpha \mathbf{L})\mathbf{x}(t-1), \tag{3}$$

or equivalently

$$\mathbf{x}(t) = (\mathbf{I}_N - \alpha \mathbf{L})^t \mathbf{x}(0) = \sum_{i=0}^{t} \binom{t}{i} (-1)^i \alpha^i \mathbf{q}(i), \tag{4}$$

where $\mathbf{q}(i) = \mathbf{L}^i \mathbf{x}(0)$. By appropriately selecting the stepsize $\alpha, (0 < \alpha < \frac{1}{d_{max}})$, with $d_{max} = \max\{d_i\}$, all nodes converge asymptotically to the same value that is the average of the initial ones, Xiao and Boyd (2004):

$$\lim_{t\to\infty} \mathbf{x}(t) = \bar{\mathbf{x}} = \frac{1}{N} \mathbf{1}\mathbf{1}^T \mathbf{x}(0).$$

Let $Sp(\mathbf{L}) = \{\lambda_1^{m_1}, \lambda_2^{m_2}, \cdots, \lambda_{D+1}^{m_{D+1}}\}$ be the Laplacian spectrum (the set of Laplacian eigenvalues), where m_i stands for the multiplicity of the i-th eigenvalue, $\sum_{i=1}^{D+1} m_i = N$, $\lambda_1 = 0$, $m_1 = 1$, and $\lambda_1 < \lambda_2 < \cdots < \lambda_{D+1}$. It is well known that the speed of convergence of (3) depends on the smallest nonzero Laplacian eigenvalue λ_2. In Xiao and Boyd (2004), it was shown that by selecting the step-size α of the average consensus algorithm as $\frac{2}{\lambda_2 + \lambda_N}$, the fastest Laplacian-based consensus algorithm is obtained.

Let $\Lambda = \{\lambda_2, \lambda_3, \cdots, \lambda_{D+1}\}$ be the set of nonzero distinct Laplacian eigenvalues. It has been shown that the averaging matrix can be factorized as, Kibangou (2012):

$$\mathbf{J} = \frac{1}{N}\mathbf{1}\mathbf{1}^T = \prod_{t=1}^{D} (\mathbf{I} - \alpha_t \mathbf{L}), \ \alpha_{t_1} \neq \alpha_{t_2} \text{ if } \quad t_1 \neq t_2, \quad \frac{1}{\alpha_t} \in \Lambda,$$

as a product of D Laplacian-based consensus matrices parameterized by the inverse of a given Laplacian eigenvalue. Therefore, the averaging matrix is a polynomial of the Laplacian matrix:

$$\mathbf{J} = \mathscr{P}_{\mathbf{c}}(\mathbf{L}) = \sum_{t=0}^{D} c_t \mathbf{L}^t,$$

with

$$c_r = (-1)^r \sum_{i<j<\ldots<r} \alpha_i \alpha_j \ldots \alpha_r, r = 0, \ldots, D \tag{5}$$

and specifically $c_0 = 1$ and $c_D = (-1)^D \prod_{t=1}^{D} \alpha_t$. As a consequence, by following the linear iteration scheme $\mathbf{x}(t) = (\mathbf{I} - \alpha_t \mathbf{L})\mathbf{x}(t-1)$, with α_t given by the inverse of Laplacian eigenvalues, average consensus is achieved in D steps and

$$\mathbf{x}(D) = \sum_{j=0}^{D} c_t \mathbf{q}(t) = \mathbf{J}\mathbf{x}(0) = \bar{\mathbf{x}}. \tag{6}$$

To summarize, a constant step-size α gives rise to an average consensus protocol with asymptotic convergence, which can be accelerated by using $\min\{\Lambda\}$ and $\max\{\Lambda\}$, the smallest and the greatest nonzero Laplacian eigenvalues, whereas all the elements of Λ yield a finite-time average consensus protocol.

2.2 Inferring the Laplacian spectrum from average consensus measurements

Equation (6) gives some insights on how estimating Laplacian eigenvalues from average consensus measurements. Indeed, assuming that the $(D+1)$ measurements $\mathbf{q}(t)$, $t = 0,1,\cdots,D$, are stored along with the final value $\bar{\mathbf{x}}$ of a standard average consensus algorithm, the coefficients of the polynomial $\mathscr{P}_{\mathbf{c}}$ can be retrieved by solving $\mathbf{Qc} = \bar{\mathbf{x}}$, where $\mathbf{Q} = (\mathbf{q}(0)\ \mathbf{q}(1)\ \cdots\ \mathbf{q}(D)) \in \Re^{N\times(D+1)}$. Then the Laplacian eigenvalues can be deduced as the inverse of the roots of the polynomial $\mathscr{P}_{\mathbf{c}}$. Obviously, the initial condition $\mathbf{q}(0) = \mathbf{x}(0)$ should be selected so that $\mathbf{Q}$ be full column rank. For instance, the entries of $\mathbf{x}(0)$ should not be all equal. Assuming that $\mathbf{x}(0)$ is randomly generated from a continuous distribution then one can deduce that almost surely $\mathbf{Q}$ is full column rank.

In these conditions, a distributed least square algorithm can be devised to solve the problem. However, this presupposes that the following assumptions are fulfilled: D, the number of distinct nonzero Laplacian eigenvalues, is known and the actual average consensus value $\bar{\mathbf{x}}$ is also known. The later assumption means that several iterations of the average consensus protocol have been run prior to the distributed estimation of the Laplacian eigenvalues. Based on these assumptions, in Tran and Kibangou (2013), the authors proposed to solve the following problem in a distributed way:

$$\begin{aligned} \min_{\boldsymbol{\alpha}_t\in\Re^{N\times 1}, t=1,2,\ldots,D} \quad & \frac{1}{2}\sum_{t=1}^{D}\sum_{i\in V}\sum_{j\in N_i}(\alpha_{t,j}-\alpha_{t,i})^2. \\ \text{subject to} \quad & \mathbf{x}(D) = \bar{\mathbf{x}}. \end{aligned} \tag{7}$$

This problem is a constrained consensus one and allows obtaining directly the inverse of the Laplacian eigenvalues. However, as formulated, it is a non-convex optimization problem. Solving this problem with a descent gradient approach suffers slow convergence, strong dependence of convergence with the initialization of the algorithm and no guarantee of convergence to the global minimum. In what follows, we propose an alternative formulation giving rise to a convex optimization problem while relaxing the conditions on the perfect knowledge of D and $\bar{\mathbf{x}}$.

3. CONVEX DISTRIBUTED OPTIMIZATION APPROACH FOR LAPLACIAN SPECTRUM ESTIMATION

The only thing we know about the number D is the fact that it is lower bounded by the diameter[1] $d(G)$ of the graph and upper bounded by $N-1$. The lower bound comes from the fact that $d(G)$ characterizes the time necessary for a given information to reach all the agents in the network while the upper bound is derived from the fact that the $N\times N$ Laplacian matrix $\mathbf{L}$ of a connected graph has at most N eigenvalues including the simple eigenvalue 0. Therefore, the averaging matrix cannot be factored with a number of matrices lower than $d(G)$.

Let us recall the following lemma:

Lemma 1. (Tran and Kibangou (2013)).
Let $\lambda_2,\cdots,\lambda_{D+1} \neq 0$ be the D distinct nonzero eigenvalues of the graph Laplacian matrix $\mathbf{L}$, then, up to permutation, the sequence $\{\alpha_i\}_{i=1,\cdots,D}$, with $\alpha_i = \frac{1}{\lambda_{i+1}}$, $i = 1,2,\cdots,D$, is the unique sequence, which allows getting the minimal factorization of the averaging matrix as $\frac{1}{N}\mathbf{1}\mathbf{1}^T = \prod_{i=1}^{D}(\mathbf{I}_N - \alpha_i\mathbf{L})$.

As a consequence, if the factorization of the averaging matrix with $h > D$ factor matrices succeeds then the obtained set of stepsizes includes necessarily the D inverse of Laplacian eigenvalues. Therefore, by taking $h = N-1$, we intend to find a larger set of values that includes the Laplacian eigenvalues. In subsection 3.2, we describe the way of retrieving the Laplacian eigenvalues from the obtained larger set.

Let us assume that the standard average consensus protocol (2) for obtaining $\bar{\mathbf{x}}$ is stopped after a finite number of iterations $M > N$. For a sufficiently large value of M, $\bar{\mathbf{x}}(M)$ can be viewed as a reasonable approximation of $\bar{\mathbf{x}}$. Therefore, the problem under study is:

Consider a network performing the average consensus protocol (3). Given the number of nodes N, the state of the network $\mathbf{x}(M)$ *at iteration* $M > N$*, and the N first consecutive measurements* $\mathbf{q}(t)$, $t = 0,1,\cdots,N-1$, *estimate the nonzero distinct Laplacian eigenvalues.*

3.1 Convex optimization problem formulation

Instead of solving the non-convex optimization problem (7) as in Tran and Kibangou (2013), we first reformulate the problem into a convex one and then solve it in a distributed way.

We know that if $\{\alpha_t\}_{t=1}^{h}$ contains the D inverses of the nonzero Laplacian eigenvalues, then $\bar{\mathbf{x}} = \prod_{t=1}^{h}(\mathbf{I}-\alpha_t\mathbf{L})\mathbf{x}(0)$ or equivalently,

$$\bar{\mathbf{x}} = \sum_{r=0}^{h} c_r\mathbf{L}^r\mathbf{x}(0) = \sum_{r=0}^{h} c_r\mathbf{q}(r),$$

Assuming that $\mathbf{x}(M)$ is a reasonable approximation of $\bar{\mathbf{x}}$, our aim is to:

(i)- find the coefficients c_r, $r = 0,1,\cdots,h$, that minimize the quadratic error on the final consensus value:

$$E(\mathbf{c}) = \left\| \sum_{r=0}^{h} c_r\mathbf{q}(r) - \mathbf{x}(M) \right\|^2; \tag{8}$$

(ii)- compute the set $S_1 = \{\boldsymbol{\alpha}_t\}_{t=1,\cdots,h}$ of the roots of the polynomial with coefficients c_r;

(iii)- deduce from S_1, the set S_2 of inverse of Laplacian eigenvalues.

In order to solve the problem in a distributed way, we first define by $c_{i,r}$ the coefficients of the polynomial of interest as estimated by node i. The optimization problem can be reformulated as follows:

$$\begin{aligned} \min_{\mathbf{c}_i\in\Re^{(h+1)}, i=1,\ldots,N} \quad & \frac{1}{2}\sum_{i=1}^{N}\Big(\sum_{r=0}^{h} c_{i,r}q_{i,r} - x_i(M)\Big)^2, \\ \text{subject to} \quad & c_{i,r} = c_{j,r}, \quad i = 1,\ldots,N; j\in N_i; \\ & \qquad\qquad r = 0,\ldots,h. \\ & \mathbf{c}_i = (c_{i,0}, c_{i,1},\cdots,c_{i,h})^T \in C_i, \end{aligned} \tag{9}$$

where C_i stands for the constrained set with respect to node i defined as $C_i = \{\boldsymbol{\gamma}\in\Re^{h+1} : \mathbf{Q}_i\boldsymbol{\gamma} = x_i(M)\mathbf{1},\ \mathbf{Q}_i\in\Re^{(d_i+1)\times(h+1)}\}$ where $\mathbf{Q}_i$ is formed with the rows of $\mathbf{Q}$ corresponding to node i and its d_i neighbors. This constraint set allows to enforce

[1] The diameter $d(G)$ of a graph is defined as $d(G) = \max_{i,j} dist(i,j)$, where given two vertices i and j, the distance $dist(i,j)$ is the length of the shortest path between i and j.

not only the local coefficients vector $\mathbf{c}_i$ to be equal in a given neighborhood but also to enforce the scalar product $\mathbf{q}_i^T\mathbf{c}_i$ to be globally equal, where $\mathbf{q}_i^T$ stands for the row of $\mathbf{Q}$ associated with node i.

In the literature, there exist several methods dealing with the distributed optimization problem (9), including distributed descent algorithms, Nedic et al. (2010), dual averaging methods, Duchi et al. (2012), and the ADMM, Boyd et al. (2011); Erseghe et al. (2011). However, we make use of the ADMM method, which has become a popular approach for solving convex minimization problems by means of parallelization. The positive features of this method is that it is guaranteed to converge for all tuning parameters Boyd et al. (2011). ADMM has only one single penalty parameter ρ that can influence on the speed of convergence. ADMM ensures in general very good convergence speed when this parameter is appropriately chosen. There exist some works dealing with penalty parameter selection for accelerating the convergence rate of ADMM Ghadimi et al. (2012); Boley (2013); Teixeira et al. (2013). The convergence analysis is also studied in Boyd et al. (2011); Erseghe et al. (2011); Boley (2013). In what follows, we describe the derivation of the ADMM algorithm for solving problem (9) with a constant penalty parameter.

For this purpose, one can note that by introducing the auxiliary variables $\{\mathbf{z}_{ij}\}$, the following constrained convex optimization problem is absolutely equivalent to problem (9):

$$\min_{\mathbf{c}_i\in\Re^{(h+1)},i=1,\ldots,N} \quad \frac{1}{2}\sum_{i=1}^{N}(\sum_{r=0}^{h} c_{i,r}q_{i,r}-x_i(M))^2. \tag{10}$$

$$\begin{aligned} \text{subject to} \quad & c_{i,r}=z_{ij,r}, \quad i=1,\ldots,N; j\in N_i \\ & z_{ji,r}=z_{ij,r}, \quad r=0,\ldots,h. \qquad (11)\\ & \mathbf{c}_i\in C_i. \end{aligned}$$

Since the graph is assumed to be connected, the constraints (11) allow enforcing $\mathbf{c}_i=\mathbf{c}_j, j\in N_i$. We can then write the associated augmented Lagrangian as:

$$\begin{aligned} L_\rho(\mathbf{c},\mathbf{z},\mathbf{y}) &= \sum_{i=1}^{N}\left(\frac{1}{2}\sum_{r=0}^{h}(c_{i,r}q_{i,r}-x_i(M))^2\right. \\ &\quad \left.+\sum_{r=0}^{h}\sum_{j\in N_i} y_{ij,r}(c_{i,r}-z_{ij,r})+\sum_{j\in N_i}\frac{\rho}{2}\|\mathbf{c}_i-\mathbf{z}_{ij}\|^2\right). \\ &= \sum_{i=1}^{N}\left(\frac{1}{2}(\mathbf{q}_i^T\mathbf{c}_i-x_i(M))^2+\sum_{j\in N_i}\mathbf{y}_{ij}^T(\mathbf{c}_i-\mathbf{z}_{ij})\right. \\ &\quad \left.+\sum_{j\in N_i}\frac{\rho}{2}\|\mathbf{c}_i-\mathbf{z}_{ij}\|^2\right). \end{aligned} \tag{12}$$

The ADMM solution is obtained by solving iteratively three sub-optimization problems. At step k, given $\mathbf{c}_i[k]$, $\mathbf{z}_{ik}[k]$, and $\mathbf{y}_{ij}[k]$, $j\in N_i$, $i=1,\cdots,N$, the following problems are to be solved:

- Minimization with respect to the polynomial coefficients $\mathbf{c}_i$, $i=1,\cdots,N$:
$$\mathbf{c}_i[k+1]=\Omega_{C_i}[\text{argmin} \quad L_\rho(\mathbf{c}_i,\mathbf{z}_{ij}[k],\mathbf{y}_i[k])]. \tag{13}$$
where $\Omega_{C_i}[\cdot]$ stands for the projection onto the constraints set of the vector in argument.
- Minimization with respect to the auxiliary variables $\mathbf{z}_{ij}$ with the constraint $\mathbf{z}_{ji}=\mathbf{z}_{ij}$:
$$\mathbf{z}_{ij}=\text{argmin} \quad L_\rho(\mathbf{c}_i[k+1],\mathbf{z}_{ij},\mathbf{y}_i[k]). \tag{14}$$
- Update of Lagrange multipliers:
$$\mathbf{y}_{ij}[k+1]=\mathbf{y}_{ij}[k]+\rho(\mathbf{c}_i[k+1]-\mathbf{z}_{ij}[k+1]). \tag{15}$$

Solving the sub-optimization problem (13) acts in two steps. First the augmented Lagrangian is minimized with respect to the vector of polynomial coefficients $\mathbf{c}_i$:

$$\hat{\mathbf{c}}_i[k+1]=\mathbf{\Psi}_i(x_i(M)\mathbf{q}_i+\rho\sum_{j\in N_i}\mathbf{z}_{ij}[k]-\sum_{j\in N_i}\mathbf{y}_{ij}[k]), \tag{16}$$

with $\mathbf{\Psi}_i=(\mathbf{q}_i\mathbf{q}_i^T+\rho d_i\mathbf{I}_{h+1})^{-1}$. Then, the obtained solution is projected onto the constraints set:

$$\mathbf{c}_i[k+1]=\Omega_{C_i}[\hat{\mathbf{c}}_i[k+1]]$$

that yields:

$$\mathbf{c}_i[k+1]=\tilde{\mathbf{Q}}_i\bar{\mathbf{x}}_{i,M}+(\mathbf{I}_{h+1}-\tilde{\mathbf{Q}}_i\mathbf{Q}_i)\hat{\mathbf{c}}_i[k+1], \tag{17}$$

with $\tilde{\mathbf{Q}}_i=\mathbf{Q}_i^T(\mathbf{Q}_i\mathbf{Q}_i^T)^{-1}$. This solution is well defined if and only if $\mathbf{Q}_i$ is a full row rank matrix. One way to guarantee this property is to have a different initial condition at each neighborhood meaning that $x_i(0)\neq x_j(0)$ for any pair (i,j).

One can note that performing this updating task does not need information exchange between neighbors since building matrices $\mathbf{Q}_i$ is carried out beforehand.

Next, solving the sub-optimization problem (14) in the same way yields

$$\mathbf{z}_{ij}[k+1]=\frac{1}{2}(\mathbf{c}_i[k+1]+\mathbf{c}_j[k+1])+\frac{1}{2\rho}(\mathbf{y}_{ij}[k]+\mathbf{y}_{ji}[k]). \tag{18}$$

Unlike the previous step, now nodes have to share their local polynomial coefficients vector $\mathbf{c}_i[k+1]$ and the Lagrange multipliers $\mathbf{y}_{ij}[k]$.

The overall ADMM algorithm for estimating the polynomial coefficient vector or equivalently performing the factorization of the averaging matrix is described in Algorithm 1.

Algorithm 1: ADMM-based distributed factorization of the averaging matrix

(1) Given the number of agents N, the intermediate measurements $\{q_i(t)\}, t=0,1,\cdots,h$ with $h=N-1$, and the value $x_i(M)$ of the state at time $M>N$, each node forms the vector $\mathbf{q}_i$, and the matrices $\mathbf{Q}_i$ and $\tilde{\mathbf{Q}}_i$.
(2) Initialization:
- Penalty parameter ρ,
- Random initial values of the coefficients $\{c_{i,r}\},\{z_{ij,r}\},\{y_{ij,r}\}$, $r=0,\ldots,h$, at each node i, $i=1,\ldots,N$.
- Set $k=0$;

(3) Update Process:
  (a) Set $k:=k+1$;
  (b) Compute $\mathbf{c}_i[k+1]$ using (16) and (17).
  (c) Each node i sends a message containing its local polynomial coefficients vector $\mathbf{c}_i[k+1]$ and the current Lagrange multipliers $\mathbf{y}_{ij}[k]$ to its neighbors $j\in N_i$.
  (d) Compute $\mathbf{z}_{ij}[k+1]$ using (18).
  (e) Update the Lagrange multipliers $\mathbf{y}_{ij}[k+1]$ using (15).
  (f) Return to (3a) or stop the iterations if a stopping criterion is reached.

(4) Each agent build a polynomial $\mathscr{P}_{\mathbf{c},i}$ with the set of coefficients $c_{i,r}[k+1]$.
(5) Each agent compute the set of the stepsizes $\{\alpha_{i,t}\}$ as the roots of the polynomial $\mathscr{P}_{\mathbf{c},i}$.

3.2 Laplacian eigenvalues retrieving.

At each node, the set S_1 of the step-size obtained by means of Algorithm 1 contains the set S_2 of the inverse of Laplacian eigenvalues. Let $\hat{\hat{x}}_i$ be the final consensus value reconstructed by node i as $\hat{\hat{x}}_i = \sum_{r=0}^{h} c_{i,r} q_{i,r} = \mathscr{P}_{\mathbf{c}_i,i}(\mathbf{q}_i)$ with the coefficients obtained by means of Algorithm 1. The idea is to reduce, step-by-step, the degree of the polynomial $\mathscr{P}_{\mathbf{c}_i,i}(\mathbf{q}_i)$ by removing one element of S_1. We know that if the removed element is also an element of S_2 then $\hat{\hat{x}}_i \neq \tilde{\mathscr{P}}_{\tilde{\mathbf{c}}_i,i}(\mathbf{q}_i)$ where $\tilde{\mathscr{P}}_{\tilde{\mathbf{c}}_i,i}$ is the polynomial with reduced degree. Such a simple test allows to conclude if the selected element is an inverse of a Laplacian eigenvalue or not. Algorithm 2 describes the proposed procedure. We can note that it is strictly local.

Algorithm 2: Laplacian eigenvalues retrieving
Given the set of step size S_1, the final consensus value $\hat{\hat{x}}_i$, and the measurements vector $\mathbf{q}_i$.

(1) Initialization: $S_2 = \{\}$ and $S_3 = S_1$.
(2) while $S_3 \neq \emptyset$, select an element α_j of S_3.
(3) From the set $S_3 \backslash \{\alpha_j\}$ compute the coefficients c_i using (5) and then form $\tilde{\mathbf{c}}_i$.
(4) If $\hat{\hat{x}}_i \neq \tilde{\mathbf{c}}_i^T \mathbf{q}_i$, include α_j in S_2 so that $S_2 = S_2 \bigcup \{\alpha_j\}$, set $S_3 = S_3 / \{\alpha_j\}$, and return to 2.
(5) If $\hat{\hat{x}}_i = \tilde{\mathbf{c}}_i^T \mathbf{q}_i$, set $S_3 = S_3 / \{\alpha_j\}$, and return to 2.
(6) If $S_3 = \emptyset$, deduce the Laplacian eigenvalues as the inverse of the elements in S_2.

In practice, the test $\hat{\hat{x}}_i = \tilde{\mathbf{c}}_i^T \mathbf{q}_i$ is replaced by $\left(\hat{\hat{x}}_i - \tilde{\mathbf{c}}_i^T \mathbf{q}_i\right)^2 < \varepsilon$, where ε is a sufficiently small positive coefficient.

The overall distributed estimation scheme is summarized in Fig. 1

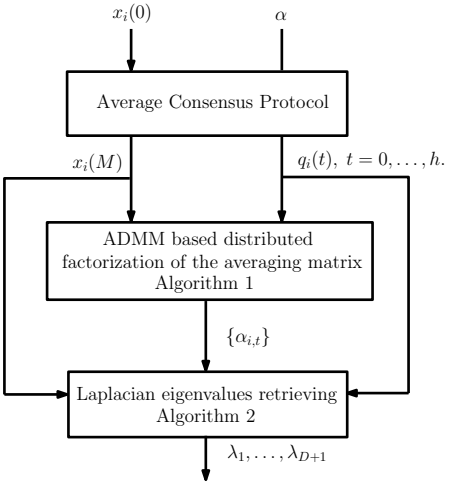

Fig. 1. Block diagram of the proposed distributed Laplacian eigenvalues estimation method.

4. SIMULATION RESULTS

In order to evaluate the performance of the proposed distributed estimation method, we consider a network of 6 nodes modeled as a cycle graph. The performance is evaluated by means of the mean square error (MSE) between the estimated Laplacian eigenvalues $\hat{\lambda}_{j,i}$ and the actual ones.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{D} (\lambda_j - \hat{\lambda}_{j,i})^2. \qquad (19)$$

The Laplacian matrix associated with this graph has 3 distinct nonzero eigenvalues $\Lambda = \{1,3,4\}$ and, therefore, the corresponding inverse of nonzero distinct eigenvalues are $\{1, 0.3333, 0.25\}$.

As depicted in Fig. 1, we first run an average consensus protocol using a constant edge weights consensus protocol with $\alpha = 0.2$ as a stepsize and

$$\mathbf{x}(0) = [0.6179, 0.0702, 0.0693, 0.1360, 0.7889, 0.0924]^T$$

as an initial condition. Fig. 2 depicts the trajectory of the state of each node. We can note that a reasonable agreement is obtained

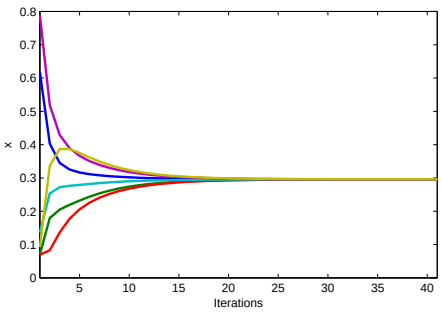

Fig. 2. Trajectory of the network state during average consensus protocol.

from iteration 10.

Then, the ADMM algorithm for computing the factorization of the averaging matrix was performed for different values for the number of iterations M of the average consensus algorithm. For instance, Fig. 3 depicts the trajectories of the estimated polynomial coefficients. We can note the good agreement between nodes on the estimation of these polynomial coefficients, which are then used to build a 5th-order polynomial whose roots contain the inverse of the Laplacian eigenvalues.

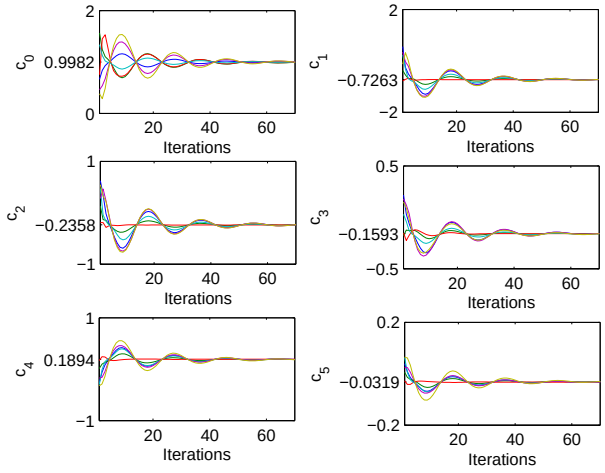

Fig. 3. Estimation of the polynomial coefficients for (M=15).

For $M \leq 10$, $\mathbf{x}(M)$ was too far from $\bar{\mathbf{x}}$. As a consequence, the obtained polynomial gave rise to negative or complex-valued roots meaning that the coefficients $\tilde{\mathbf{c}}$ created from these roots do not satisfy $\hat{\hat{x}}_i = \tilde{\mathbf{c}}_i^T \mathbf{q}_i$ in Algorithm 2. For $M > 10$, Algorithm 2 was successfully performed for retrieving the Laplacian eigenvalues. Fig. 4 depicts the MSE on the estimation of the Laplacian eigenvalues and the corresponding standard deviation. As expected, we can note that the closer $\mathbf{x}(M)$ is to

$\bar{\mathbf{x}}$, the higher the precision on the computation of the Laplacian eigenvalues is. In addition, the tight bounds around the MSE inform us about the agreement on the estimated Laplacian eigenvalues.

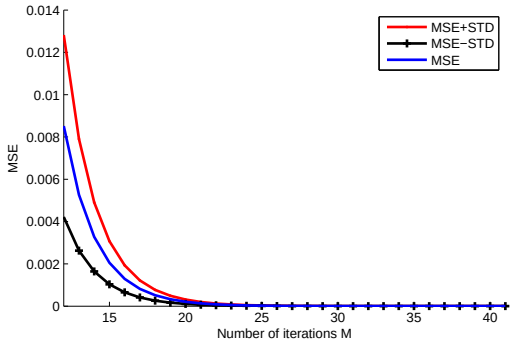

Fig. 4. Performance of Laplacian eigenvalues estimation for different values of the number of iterations of the standard average consensus algorithm.

5. CONCLUSION

In this paper, we have proposed an improved distributed method to estimate the Laplacian nonzero distinct eigenvalues of a connected undirected graph representing interactions between nodes of a given network. A convex optimization problem has been proposed. It consists in finding the optimal coefficients of a given polynomial whose roots are precisely the stepsizes $\alpha_t, t = 1, \ldots, D$, allowing to factorize the averaging matrix as a product of Laplacian-based average consensus matrices. The stepsizes are the inverse of the Laplacian eigenvalues. To solve this problem in a distributed way, we have devised an ADMM approach that exhibits in general fast convergence. The drawback of the proposed method concerns its scalability. Indeed, for very large graphs the roots of very high degree polynomials are to be computed. Furthermore, the proposed algorithms are implemented perfectly if the average consensus value $\bar{x}$ is estimated. In stead of solving two tasks, which are average consensus protocol and Laplacian eigenvalues estimation, we can combine them into one problem that simultaneously estimates the average consensus value $\bar{x}$ and nonzero distinct Laplacian eigenvalues. This problem is still under investigation.

REFERENCES

Aragues, R., Shi, G., Dimarogonas, D., Sagues, C., and Johansson, K. (2012). Distributed algebraic connectivity estimation for adaptive event-triggered consensus. In *Proc. of American Control Conf. (ACC), 2012*, 32–37.

Boley, D. (2013). Local linear convergence of ADMM on quadratic or linear programs. *SIAM J. on Optimization*, 23(4), 2183–2207.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1), 1–122.

Chung, F.R.K. (2012). *Spectral Graph Theory*. American Mathematical Society, Providence, RI.

Duchi, J., Agarwal, A., and Wainwright, M. (2012). Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Trans. on Automatic Control*, 57(3), 592–606.

Erseghe, T., Zennaro, D., Dall'Anese, E., and Vangelista, L. (2011). Fast consensus by the alternating direction multipliers method. *IEEE Trans. on Signal Processing*, 59(11), 5523–5537.

Franceschelli, M., Gasparri, A., Giua, A., and Seatzu, C. (2013). Decentralized estimation of Laplacian eigenvalues in multi-agent systems. *Automatica*, 49(4), 1031–1036.

Franceschelli, M., Martini, S., Egerstedt, M., Bicchi, A., and Giua, A. (2010). Observability and controllability verification in multi-agent systems through decentralized Laplacian spectrum estimation. In *Proc. of the 49th IEEE Conf. on Decision and Control (CDC)*, 5775–5780. Atlanta, Georgia, USA.

Friedler, M. (1973). Algebraic connectivity of graphs. *Czechoslovak Mathematical J.*, 23, 298–305.

Ghadimi, E., Teixeira, A., Shames, I., and Johansson, M. (2012). On the optimal step-size selection for the alternating direction method of multipliers. In *the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'12)*, volume 3, 139–144. Santa Barbara, California.

Kibangou, A. (2011). Finite-time average consensus based protocol for distributed estimation over AWGN channels. In *the 50th IEEE Conf. on Decision and Control and European Control Conf. (CDC-ECC)*, 5595–5600. Orlando, Fl, USA.

Kibangou, A. (2012). Graph Laplacian based matrix design for finite-time distributed average consensus. In *Proc. of American Control Conf. (ACC)*, 1901–1906. Montréal, Canada.

Kibangou, A. (2014). Step-size sequence design for finite-time average consensus in secure wireless sensor networks. *Systems and Control Letters*, 67, 19–23.

Kibangou, A. and Commault, C. (2012). Decentralized Laplacian eigenvalues estimation and collaborative network topology identification. In *the 3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'12)*, 7–12. Santa Barbara, USA.

Merris, R. (1994). Laplacian matrices of a graph: a survey. *Linear Algebra and its Applications*, 197, 143–176.

Nedic, A., Ozdaglar, A., and Parrilo, P. (2010). Constrained consensus and optimization in multi-agent networks. *IEEE Trans. on Automatic Control*, 55(4), 922–938.

Olfati-saber, R., Fax, J.A., and Murray, R.M. (2010). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 98(7), 1353–1354.

Sahai, T., Speranzon, A., and Banaszuk, A. (2012). Hearing the cluster of a graph: A distributed algorithm. *Automatica*, 48(1), 15–24.

Teixeira, A., Ghadimi, E., Shames, I., Sandberg, H., and Johansson, M. (2013). Optimal scaling of the ADMM algorithm for distributed quadratic programming. In *Proc. of the 52nd IEEE Conf. on Decision and Control (CDC)*, 6868 – 6873. Florence, Italy.

Tran, T. and Kibangou, A. (2013). Consensus-based distributed estimation of Laplacian eigenvalues of undirected graphs. In *Proc. of the European Control Conf. (ECC 2013)*, 227–232. Zurich, Switzerland.

Xiao, L. and Boyd, S. (2004). Fast linear iterations for distributed averaging. *Systems Control Letters*, 53, 65–78.

Yang, P., Freeman, R.A., Gordon, G., Lynch, K., Srinivasa, S., and Sukthankar, R. (2008). Decentralized estimation and control of graph connectivity in mobile sensor networks. In *Proc. of American Control Conf. (ACC)*, 2678–2683. Seattle, Washington, USA.