Automatica ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Hearing the clusters of a graph: A distributed algorithm☆

Tuhin Sahai [1], Alberto Speranzon, Andrzej Banaszuk

United Technologies Research Center, East Hartford, CT 06108, USA

ARTICLE INFO

Article history:
Received 20 June 2010
Received in revised form
15 April 2011
Accepted 9 June 2011
Available online xxxx

Keywords:
Decentralized systems
Graph theory
Decomposition methods

ABSTRACT

We propose a novel distributed algorithm to cluster graphs. The algorithm recovers the solution obtained from spectral clustering without the need for expensive eigenvalue/eigenvector computations. We prove that, by propagating waves through the graph, a local fast Fourier transform yields the local component of every eigenvector of the Laplacian matrix, thus providing clustering information. For large graphs, the proposed algorithm is orders of magnitude faster than random walk based approaches. We prove the equivalence of the proposed algorithm to spectral clustering and derive convergence rates. We demonstrate the benefit of using this decentralized clustering algorithm for community detection in social graphs, accelerating distributed estimation in sensor networks and efficient computation of distributed multi-agent search strategies.

1. Introduction

In recent years, there has been great interest in the analysis of large interconnected systems, such as sensors networks, social networks, the Internet, biochemical networks, power networks, etc. These systems are characterized by complex behavior that arises due to interacting subsystems. Graph theoretic methods have recently been applied and extended to study these systems. In particular, spectral properties of the Laplacian matrix associated with such graphs provide useful information for the analysis and design of interconnected systems. The computation of eigenvectors of the graph Laplacian is the cornerstone of spectral graph theory (Chung, 1997; von Luxburg, 2007), and it is well known that the sign of the second (and successive) eigenvectors can be used to cluster graphs (Fiedler, 1973, 1975).

The problem of graph (or data, in general) clustering arises naturally in applications ranging from social anthropology (Kottak, 1991), gene networks (Speer, Fröhlich, Spieth, & Zell, 2005), protein sequences (Paccanaro, Casbon, & Saqi, 2006), sensor networks (Akyildiz, Su, Sankarasubramaniam, & Cayirci, 2002; Ghiasi, Srivastava, Yang, & Sarrafzadeh, 2002; Muhammad & Jadbabaie, 2007), computer graphics (Herman, Melançon, & Marshall, 2000) and Internet routing algorithms (Kempe & McSherry, 2008).

The basic idea behind graph decomposition is to cluster nodes into groups with strong intra-connections but weak inter-connections. If one poses the clustering problem as a minimization of the inter-connection strength (sum of edge weights between clusters), it can be solved exactly and quickly (Stoer & Wagner, 1997). However, the decomposition obtained is often unbalanced (some clusters are large and others small) (von Luxburg, 2007). To avoid unbalanced cuts, size restrictions are typically placed on the clusters, i.e., instead of minimizing inter-connection strength, we minimize the ratio of the inter-connection strength to the size of individual clusters. This, however, makes the problem NP-complete (Wagner & Wagner, 1993). Several heuristics to partition graphs have been developed over the past few decades (Porter, Onnela, & Mucha, 2009) including the Kernighan–Lin algorithm (Kernighan & Lin, 1970), Potts method (Reichardt & Burnholdt, 2004), percolation based methods (Palla, Derényi, Farkas, & Vicsek, 2005), horizontal–vertical decomposition (Varigonda, Kalmar-Nagy, Labarre, & Mezic, 2004) and spectral clustering (Fiedler, 1973, 1975).

1.1. Spectral clustering

Spectral clustering has emerged as a powerful tool of choice for graph decomposition purposes (see von Luxburg, 2007 and references therein). The method assigns nodes to clusters based on the signs of the elements of the eigenvectors of the Laplacian corresponding to increasing eigenvalues (Chung, 1997; Fiedler, 1973, 1975). In Spielman and Teng (2004), the authors have developed a distributed algorithm for spectral clustering of graphs. The algorithm involves performing random walks, and at every step neglecting probabilities below a threshold value. The nodes are then ordered by the ratio of probabilities to node degree and

☆ The material in this paper was partially presented at the 49th IEEE Conference on Decision and Control, December 15–17, 2010, Atlanta, Georgia, USA. This paper was recommended for publication in revised form by Associate Editor Andrea Serrani under the direction of Editor Miroslav Krstic.

E-mail addresses: SahaiT@utrc.utc.com (T. Sahai), SperanA@utrc.utc.com (A. Speranzon), BansaszA@utrc.utc.com (A. Banaszuk).

[1] Tel.: +1 860 610 7763; fax: +1 860 622 0666.

0005-1098/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2011.09.019

grouped into clusters. Since this algorithm is based on random walks, it suffers, in general, from slow convergence.

Since the clustering assignment is computed using the eigenvectors/eigenvalues of the Laplacian matrix, one can use standard matrix algorithms for such computation (Golub & Loan, 1996). However, as the size of the matrix (and thus the corresponding network) increases, the execution of these standard algorithms becomes infeasible on monolithic computing devices. To address this issue, algorithms for distributed eigenvector computations have been proposed (Kempe & McSherry, 2008). These algorithms, however, are also (like the algorithm in Spielman & Teng, 2004) based on the slow process of performing random walks on graphs.

1.2. Wave equation method

In a theme similar to Mark Kac's question "Can one hear the shape of a drum?" (Kac, 1966), we demonstrate that by evolving the wave equation in the graph, nodes can "hear" the eigenvectors of the graph Laplacian using only local information. Moreover, we demonstrate, both theoretically and on examples, that the wave equation based algorithm is orders of magnitude faster than random walk based approaches for graphs with large mixing times. The overall idea of the wave equation based approach is to simulate, in a distributed fashion, the propagation of a wave through the graph and capture the frequencies at which the graph "resonates". In this paper, we show that by using these frequencies one can compute the eigenvectors of the Laplacian, thus clustering the graph. We also derive conditions that the wave must satisfy in order to cluster graphs using the proposed method.

The paper is organized as follows: in Section 2 we describe current methodologies for distributed eigenvector/clustering computation based on the heat equation. In Section 3 the new proposed wave equation method is presented. In Section 4 we determine bounds on the convergence time of the wave equation. In Section 5 we show some numerical clustering results for a few graphs, including a large social network comprising of thousands of nodes and edges. We then show, in Section 6, how the wave equation can be used to accelerate distributed estimation in a large-scale environment such as a building. In Section 7 we show how the proposed distributed clustering algorithm enables one to efficiently transform a centralized search algorithm into a decentralized one. Finally, conclusions are drawn in Section 8.

2. From heat to wave equation: related work

Let $\mathcal{G} = (V, E)$ be a graph with vertex set $V = \{1, \ldots, N\}$ and edge set $E \subseteq V \times V$, where a weight $\mathbf{W}_{ij} \geq 0$ is associated with each edge $(i, j) \in E$, and $\mathbf{W}$ is the $N \times N$ weighted adjacency matrix of $\mathcal{G}$. We assume that $\mathbf{W}_{ij} = 0$ if and only if $(i, j) \notin E$. The (normalized) graph Laplacian is defined as,

$$\mathbf{L}_{ij} = \begin{cases} 1 & \text{if } i = j \\ -\mathbf{W}_{ij} \Big/ \sum_{\ell=1}^{N} \mathbf{W}_{i\ell} & \text{if } (i, j) \in E \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

or equivalently, $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}$ where $\mathbf{D}$ is the diagonal matrix with the row sums of $\mathbf{W}$.

Note that in this work we only consider undirected graphs. The smallest eigenvalue of the Laplacian matrix is $\lambda_1 = 0$, with an associated eigenvector $\mathbf{v}^{(1)} = \mathbf{1} = [1, 1, \ldots, 1]^T$. Eigenvalues of $\mathbf{L}$ can be ordered as, $0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_N$ with associated eigenvectors $\mathbf{1}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)}, \ldots, \mathbf{v}^{(N)}$ (von Luxburg, 2007). It is well known that the multiplicity of λ_1 is the number of connected components in the graph (Mohar, 1991). We assume in the following that $\lambda_1 < \lambda_2$ (the graph does not have disconnected clusters). We also assume that there exist unique cuts that divide the graph into k clusters. In other words, we assume that there exist k distinct eigenvalues close to zero (Luxburg, Bousquet, & Belkin, 2004).

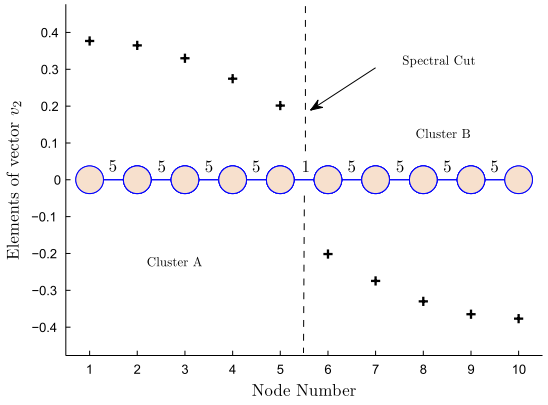


Fig. 1. Spectral clustering: The sign of the i-th element of eigenvector v_2 determines the cluster assignment of the i-th vertex, demonstrated on a simple line graph example (shown in the center). With $+$ we plot the value of the components of v_2.

Given the Laplacian matrix $\mathbf{L}$, associated with a graph $\mathcal{G} = (V, E)$, spectral clustering divides $\mathcal{G}$ into two clusters by computing the sign of the N elements of the second eigenvector $\mathbf{v}^{(2)}$, or Fiedler vector (Fiedler, 1975; von Luxburg, 2007). This process is depicted in Fig. 1 for a line graph where one edge (the edge $(5, 6)$) has lower weight than other edges.

More than two clusters can be computed from signs of the elements of higher eigenvectors, i.e. $\mathbf{v}^{(3)}, \mathbf{v}^{(4)}$, etc. von Luxburg (2007). Alternatively, once the graph is divided into two clusters, the spectral clustering algorithm can be run independently on both clusters to compute further clusters. This process is repeated until either a desired number of clusters is found or no further clusters can be computed. This method can also be used to compute a hierarchy of clusters.

There are many algorithms to compute eigenvectors, such as the Lanczos method or orthogonal iteration (Golub & Loan, 1996). Although some of these methods are distributable, convergence is slow (Golub & Loan, 1996) and the algorithms do not consider/take advantage of the fact that the matrix for which the eigenvalues and eigenvectors need to be computed is the adjacency matrix of the underlying graph. In Kempe and McSherry (2008), the authors propose an algorithm to compute the first k largest eigenvectors (associated with the first k eigenvalues with greatest absolute value)[2] of a symmetric matrix. The algorithm in Kempe and McSherry (2008) emulates the behavior of orthogonal iteration. To compute the first k eigenvectors of a given matrix $\mathbf{J}$, at each node in the network, matrix $\mathbf{V}_i = \sum_{j \in \mathcal{N}(i)} \mathbf{J}_{ij}\mathbf{Q}_j$ is computed, where $\mathbf{Q}_j \in \mathbb{R}^{N \times k}$ is initialized to a random matrix and $\mathcal{N}(i)$ is the set of neighbors of node i (including node i itself). Orthonormalization is achieved by the computation of matrix $\mathbf{K}_i = \mathbf{V}_i^T\mathbf{V}_i$ at every node, followed by computation of matrix $\mathbf{K}$, which is the sum of all the $\mathbf{K}_i$ matrices in the network. Once matrix $\mathbf{K}$ is computed, $\mathbf{Q}_i = \mathbf{V}_i\mathbf{R}^{-1}$ is updated at each node, where $\mathbf{R}$ is a unique matrix such that $\mathbf{K} = \mathbf{R}^T\mathbf{R}$ (Cholesky decomposition). The above iteration is repeated until $\mathbf{Q}_i$ converges to the i-th eigenvector. The sum of all the matrices $\mathbf{K}_i$ is done in a decentralized way, using gossip (Shah, 2009), which is a deterministic simulation of a random walk on the network. In particular, at each node one computes the matrix $\mathbf{K}$ as

[2] Note that in the case of spectral clustering we desire to compute the smallest k eigenvectors of $\mathbf{L}$. The algorithm is still applicable if we consider the matrix $\mathbf{I} - \mathbf{L}$.

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019

follows,

$$\mathbf{S}_i(t+1) = \sum_{j\in\mathcal{N}(i)} \mathbf{B}_{ji}\mathbf{S}_j(t), \tag{2}$$

$$\boldsymbol{\pi}_i(t+1) = \sum_{j\in\mathcal{N}(i)} \mathbf{B}_{ji}\boldsymbol{\pi}_j(t), \tag{3}$$

for $t \geq \tau$ steps, where τ is the mixing time for the random walk on the graph (Kempe & McSherry, 2008). Here $\mathbf{K} = \mathbf{S}_i/\boldsymbol{\pi}_i$, $\mathbf{S}_i(0) = \mathbf{K}_i$ and $\boldsymbol{\pi}_i(0) = 1$ for only one index i and zero for other indices. The values $\mathbf{B}_{ij}$ are transition probabilities of the Markov chain associated with the graph. A natural choice is $\mathbf{B}_{ij} = 1/\deg(i)$, where $\deg(i)$ is the degree of node i. Note that matrix $\mathbf{B} = [\mathbf{B}_{ij}]$ is the normalized adjacency matrix (given by $\mathbf{D}^{-1}\mathbf{W}$). This algorithm converges after $O(\tau \log^2 N)$ iterations (Kempe & McSherry, 2008).

The slowest step in the distributed computation of eigenvectors is the simulation of a random walk on the graph (defined by Eqs. (2) and (3)). It is clear from Eq. (1) that successive multiplications by the adjacency matrix $\mathbf{B}$ in Eqs. (2) and (3) are equivalent to successive multiplications by matrix $\mathbf{I} - \mathbf{L}$. This procedure is equivalent to evolving the discretized heat equation on the graph and can be demonstrated as follows. The heat equation is given by

$$\frac{\partial u}{\partial t} = \Delta u,$$

where u is a function of time and space, $\partial u/\partial t$ is the partial derivative of u with respect to time, and Δ is the Laplace operator (Evans, 1998). When the above equation is discretized (see Belkin & Niyogi, 2008, Chung, 1997, Hein, 2006, and Hein, Audibert, & von Luxburg, 2005 for details) on a graph $\mathcal{G} = (V, E)$ one gets the following equation:

$$\mathbf{u}_i(t+1) = \mathbf{u}_i(t) - \sum_{j\in\mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t),$$

for $i, j \in V$. Here $\mathbf{u}_i(t)$ is the scalar value of u on node i at time t. The graph Laplacian $\mathbf{L} = [\mathbf{L}_{ij}]$ appears due to the discretization of the Δ operator (Hein et al., 2005). The above iteration can be re-written, in matrix form, $\mathbf{u}(t+1) = (\mathbf{I} - \mathbf{L})\mathbf{u}(t)$ where $\mathbf{u}(t) = (\mathbf{u}_1(t), \ldots, \mathbf{u}_N(t))^T$. The solution of this iteration is,

$$\mathbf{u}(t) = C_0\mathbf{1} + C_1(1-\lambda_2)^t\mathbf{v}^{(2)} + \cdots + C_N(1-\lambda_N)^t\mathbf{v}^{(N)}, \tag{4}$$

where constants C_j depend on the initial condition $\mathbf{u}(0)$. It is interesting to note that in Eq. (4), the dependence of the solution on higher eigenvectors and eigenvalues of the Laplacian decays with increasing iteration count. Thus, it is difficult to devise a fast and distributed method for clustering graphs based on the heat equation. Next, we derive a novel algorithm based on the idea of permanent excitation of the eigenvectors of $\mathbf{I} - \mathbf{L}$. We note that the above connection between spectral clustering and the heat equation is not new and was pointed out in Nadler, Lafon, Coifman, and Kevrekidis (2006a,b).

Before discussing the details of wave-equation based eigenvector computation, we remark that in Franceschelli, Gasparri, and Seatzu (2009) the authors have independently developed a decentralized algorithm to compute the eigenvalues of the Laplacian. Compared to our approach, their algorithm involves solving a fourth order partial differential equation on the graph. This imposes twice the cost of communication, computation and memory on every node in the graph.

3. Wave equation based computation

Consider the wave equation,

$$\frac{\partial^2 u}{\partial t^2} = c^2\Delta u. \tag{5}$$

Analogous to the heat equation case (Eq. (4)), the solution of the wave equation can be expanded in terms of the eigenvectors of the Laplacian. However, unlike the heat equation where the solution eventually converges to the first eigenvector of the Laplacian, in the wave equation all the eigenvectors remain eternally excited (Evans, 1998) (a consequence of the second derivative of u with respect to time). Here we use this observation to develop a simple, yet powerful, distributed eigenvector computation algorithm. The algorithm involves evolving the wave equation on the graph and then computing the eigenvectors using local FFTs. Note that some properties of the wave equation on graphs have been studied in Friedman and Tillich (2004). Here we construct a graph decomposition/partitioning algorithm based on the discretized wave equation on the graph, given by

$$\mathbf{u}_i(t) = 2\mathbf{u}_i(t-1) - \mathbf{u}_i(t-2) - c^2\sum_{j\in\mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t-1), \tag{6}$$

where $\sum_{j\in\mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t-1)$ originates from the discretization of Δu in Eq. (5), see Hein et al., 2005 for details. The rest of the terms originate from discretization of $\partial^2 u/\partial t^2$. To update $\mathbf{u}_i$ using Eq. (6), one needs only the value of $\mathbf{u}_j$ at neighboring nodes and the connecting edge weights (along with previous values of $\mathbf{u}_i$).

The main steps of the algorithm are shown as Algorithm 3.1. Note that at each node (node i in the algorithm) one only needs nearest neighbor weights $\mathbf{L}_{ij}$ and the scalar quantities $\mathbf{u}_j(t-1)$ also at nearest neighbors. We emphasize, again, that $\mathbf{u}_i(t)$ is a scalar quantity and `Random` ([0, 1]) is a random initial condition on the interval [0, 1]. The vector $\mathbf{v}_i^{(j)}$ is the i-th component of the j-th eigenvector, $T_{\max}$ is a positive integer derived in Section 4, `FrequencyPeak(Y, j)` returns the frequency at which the j-th peak occurs and `Coefficient`(ω_j) return the corresponding Fourier coefficient.

Algorithm 3.1 Wave equation based eigenvector computation algorithm for node i. At node i one computes the sign of the i-th component of the first k eigenvectors. The cluster assignment is obtained by interpreting the vector of k signs as a binary number.

$\mathbf{u}_i(0) \leftarrow$ `Random` ([0, 1])
$\mathbf{u}_i(-1) \leftarrow \mathbf{u}_i(0)$
$t \leftarrow 1$
while $t < T_{max}$ **do**
  $\mathbf{u}_i(t) \leftarrow 2\mathbf{u}_i(t-1) - \mathbf{u}_i(t-2) - c^2\sum_{j\in\mathcal{N}(i)} \mathbf{L}_{ij}\mathbf{u}_j(t-1)$
  $t \leftarrow t+1$
end while
$Y \leftarrow$ FFT $([\mathbf{u}_i(1), \ldots\ldots, \mathbf{u}_i(T_{max})])$
for $j \in \{1, \ldots, k\}$ **do**
  $\omega_j \leftarrow$ `FrequencyPeak` (Y, j)
  $\mathbf{v}_i^{(j)} \leftarrow$ `Coefficient`(ω_j)
  if $\mathbf{v}_i^{(j)} > 0$ **then**
    $A_j \leftarrow 1$
  else
    $A_j \leftarrow 0$
  end if
end for
ClusterNumber $\leftarrow \sum_{j=1}^{k} A_j 2^{j-1}$

Proposition 3.1. *The wave equation iteration (6) is stable on any graph if the wave speed satisfies the following inequality,*

$$0 < c < \sqrt{2},$$

with an initial condition of $\mathbf{u}(-1) = \mathbf{u}(0)$.

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019

Proof. For analysis of the algorithm, we consider Eq. (6) in vector form,

$$\mathbf{u}(t) = -\mathbf{u}(t-2) + (2\mathbf{I} - c^2\mathbf{L})\mathbf{u}(t-1). \tag{7}$$

We stress again that, in practice, the algorithm is distributed and at every node one updates the state based on Eq. (6). The update equations given by Eq. (6) (and Eq. (7)) correspond to discretization of Eq. (5) with Neumann boundary conditions (Coifman, Shkolnisky, Sigworth, & Singer, 2008).

One can write iteration Eq. (7) in matrix form,

$$\underbrace{\begin{pmatrix}\mathbf{u}(t)\\ \mathbf{u}(t-1)\end{pmatrix}}_{\mathbf{z}(t)} = \underbrace{\begin{pmatrix}2\mathbf{I}-c^2\mathbf{L} & -\mathbf{I}\\ \mathbf{I} & 0\end{pmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix}\mathbf{u}(t-1)\\ \mathbf{u}(t-2)\end{pmatrix}}_{\mathbf{z}(t-1)}. \tag{8}$$

This implies that,

$$\mathbf{z}(t) = \mathbf{M}^t\mathbf{z}(0), \tag{9}$$

where $\mathbf{z}(0) = (\mathbf{u}(0), \mathbf{u}(-1))^T$. We now analyze the solution to Eq. (9) in terms of the eigenvalues and eigenvectors of the graph Laplacian $\mathbf{L}$.

We can compute the eigenvectors of $\mathbf{M}$ by solving for a generic vector $(\mathbf{a}_j, \mathbf{b}_j)^T$,

$$\mathbf{M}\begin{pmatrix}\mathbf{a}_j\\ \mathbf{b}_j\end{pmatrix} = \alpha_j\begin{pmatrix}\mathbf{a}_j\\ \mathbf{b}_j\end{pmatrix}.$$

This implies that the eigenvectors of $\mathbf{M}$ are given by

$$\mathbf{m}^{(j)} = \begin{pmatrix}\alpha_j\, v^{(j)}\\ \mathbf{v}^{(j)}\end{pmatrix}, \tag{10}$$

with eigenvalues

$$\alpha_{j_{1,2}} = \frac{2-c^2\lambda_j}{2} \pm \frac{c}{2}\sqrt{c^2\lambda_j^2 - 4\lambda_j}. \tag{11}$$

It is evident from Eq. (11) that stability is obtained if and only if,

$$\left|\frac{2-c^2\lambda_j}{2} \pm \frac{\sqrt{(2-c^2\lambda_j)^2-4}}{2}\right| \le 1.$$

The absolute value from the above equation is plotted for various values of $\theta_j = 2 - c^2\lambda_i$, in Fig. 2. The above stability condition is satisfied for $-2 \le \theta_j \le 2$, which yields the following bound on c:

$$0 \le c \le \frac{2}{\sqrt{\lambda_i}}.$$

The above equation must hold true for all eigenvalues of $\mathbf{L}$. The most restrictive of which is $c \le 2/\sqrt{\lambda_N}$. Since $\lambda_N \le 2$ for all graphs,

$$0 \le c \le \sqrt{2},$$

guarantees that all the eigenvalues of $\mathbf{M}$ have absolute value equal to one. However, Eq. (6) will be unstable if any of the eigenvalues of $\mathbf{M}$ have geometric multiplicity strictly less than the algebraic multiplicity with an initial condition that has non-zero projection on the unstable generalized eigenvectors. We now derive conditions so that these instabilities do not arise.

From Eq. (11) it is evident that there are three cases to analyze.

Case (i): Since $\mathbf{L}$ always has an eigenvalue at 0, this implies that $\mathbf{M}$ always has an eigenvalue at 1 with algebraic multiplicity two. It can be shown that the geometric multiplicity of this eigenvalue is equal to one. The corresponding eigenvector is $\mathbf{1}_{2N\times 1}$, with a generalized eigenvector $(\mathbf{1}, -\mathbf{1})^T$. To avoid instability, the initial conditions must be of the form $(\mathbf{u}(0), \mathbf{u}(0))^T$. In other words, we set $\mathbf{u}(-1) = \mathbf{u}(0)$ to ensure that the initial condition is orthogonal to $(\mathbf{1}, -\mathbf{1})^T$.

Case (ii): If $\mathbf{L}$ has k repeated eigenvalues, it implies that $\mathbf{M}$ has k repeated eigenvalues. In this case, however, the geometric and algebraic multiplicities are equal. One can show that the matrix $\mathbf{L}$ is similar to the symmetric matrix,

$$\mathbf{L}_{\text{sym}} = \mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{W})\mathbf{D}^{-1/2},$$

(in particular, $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{L}_{\text{sym}}\mathbf{D}^{1/2}$), implying that $\mathbf{L}$ is diagonalizable. Thus, the eigenvectors of $\mathbf{L}$ associated with the repeated eigenvalues are linearly independent. Since matrix $\mathbf{M}$ has eigenvectors of the form shown in Eq. (10), the repeated eigenvalues of $\mathbf{M}$ have eigenvectors that are linearly independent.

Case (iii): The matrix $\mathbf{M}$ has a repeated eigenvalue at -1 if $c^2 = 2$ and $\lambda_N = 2$. This repeated eigenvalue has an associated eigenvector $(-\mathbf{v}_N, \mathbf{v}_N)^T$ and a generalized eigenvector $(\mathbf{v}_N, \mathbf{v}_N)^T$. Clearly, in this case the initial condition would need to be orthogonal to both the vector $(\mathbf{1}, -\mathbf{1})^T$ and the vector $(\mathbf{v}_N, \mathbf{v}_N)^T$. This can be achieved if and only if $\mathbf{u}(0) \perp \mathbf{v}_N$ and $\mathbf{u}(-1) = \mathbf{u}(0)$. This is an undesirable condition, as it requires prior knowledge of $\mathbf{v}_N$. We avoid this situation by setting $c < \sqrt{2}$.

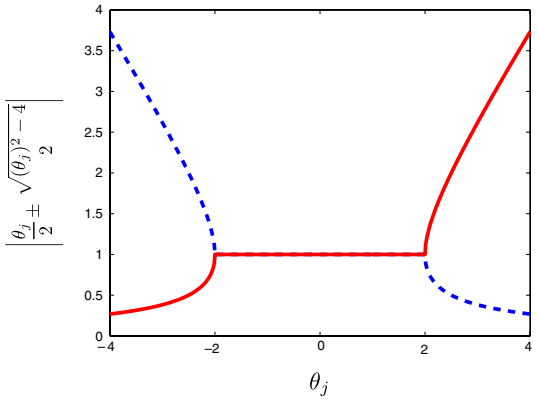


Fig. 2. Plot of functions $|\theta_j/2 \pm 1/2\sqrt{(\theta_j)^2-4}|$. Blue (dashed) line is the function with a negative second term. Red (solid) line is the function with a positive second term. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Thus, we can guarantee stability of the wave equation iteration on any graph (given by Eq. (6)), as long as $0 < c < \sqrt{2}$ and the initial condition has the form $\mathbf{u}(-1) = \mathbf{u}(0)$.

Notice that the condition $\mathbf{u}(-1) \neq \mathbf{u}(0)$ is analogous to a non-zero initial derivative condition on u for the continuous PDE, which is known to give a solution that grows in time (Evans, 1998). □

Remark 3.2. Although we call c the wave speed, it only controls the extent to which neighbors influence each other and not the speed of information propagation in the graph.

Proposition 3.3. *The clusters of graph $\mathcal{G}$, determined by the signs of the elements of the eigenvectors of $\mathbf{L}$, can be computed using the frequencies and coefficients obtained from the Fast Fourier Transform of $(\mathbf{u}_i(1), \ldots, \mathbf{u}_i(T_{\max}))$, for all i and some $T_{\max} > 0$. Here $\mathbf{u}_i$ is governed by the wave equation on the graph with the initial condition $\mathbf{u}(-1) = \mathbf{u}(0)$ and $0 < c < \sqrt{2}$.*

Proof. We can write the eigenvectors $\mathbf{m}^{(j)}$ of $\mathbf{M}$ as,

$$\mathbf{m}^{(j)} = \mathbf{p}^{(j)} \pm i\mathbf{q}^{(j)},$$

where,

$$\mathbf{p}^{(j)} = \begin{pmatrix} \text{Real}(\alpha_j)\mathbf{v}^{(j)} \\ \mathbf{v}^{(j)} \end{pmatrix}, \qquad \mathbf{q}^{(j)} = \begin{pmatrix} \text{Imag}(\alpha_j)\mathbf{v}^{(j)} \\ 0 \end{pmatrix}.$$

Using $\alpha_j = e^{i\omega_j}$, we can represent the solution of the update equation (Eq. (6)), or equivalently,

$$\mathbf{z}(t) = \mathbf{M}^t \mathbf{z}(0), \tag{12}$$

by expanding $\mathbf{z}(0)$ in terms of $\mathbf{p}^{(j)}$ and $\mathbf{q}^{(j)}$. Recall, that $\mathbf{z}(0) = (\mathbf{u}(0), \mathbf{u}(0))^T$ is orthogonal to the generalized eigenvector $(\mathbf{1}, -\mathbf{1})^T$. Thus, $\mathbf{z}(0)$ is represented as a linear combination of $(\mathbf{1}, \mathbf{1})^T$ and $\mathbf{m}^{(j)}$ for $j \geq 2$. This implies that the solution to Eqs. (8) and (9) is given by

$$\mathbf{z}(t) = \sum_{j=1}^{N} C_{j_1} \left[\mathbf{p}^{(j)} \cos(t\omega_j) - \mathbf{q}^{(j)} \sin(t\omega_j)\right] + C_{j_2} \left[\mathbf{p}^{(j)} \sin(t\omega_j) + \mathbf{q}^{(j)} \cos(t\omega_j)\right], \tag{13}$$

where

$$C_{j_1} = \mathbf{z}(0)^T \mathbf{p}^{(j)}, \qquad C_{j_2} = \mathbf{z}(0)^T \mathbf{q}^{(j)}. \tag{14}$$

It is easy to see that at every node, say the i-th node, one can locally perform an FFT on $(\mathbf{u}_i(1), \ldots, \mathbf{u}_i(T_{\max}))$ (where each value is computed using the update law in Eq. (6)) to obtain the eigenvectors. At the i-th node of the graph, one computes the i-th component of every eigenvector from the coefficients of the FFT. More precisely, for node i, the coefficient of $\cos(t\omega_j)$ is given $(C_{j_1} + C_{j_2})\mathbf{v}_i^{(j)}$. The sign of the coefficients of the eigenvector(s) provide the cluster assignment(s). □

Remark 3.4. The above algorithm assumes that one excites every frequency (or depending on the number of clusters, at least the first k frequencies). This is achieved if $\mathbf{z}(0)$ is not orthogonal to $\mathbf{p}^{(j)}$ and $\mathbf{q}^{(j)}$ (C_{j_1} and C_{j_2} must be non-zero). As mentioned before, an initial condition of the form $\mathbf{z}(0) = (\mathbf{u}(0), \mathbf{u}(0))^T$ prevents linear growth of the solution, however, $\mathbf{u}(0)$ should also not be orthogonal to $\mathbf{v}^{(2)}, \mathbf{v}^{(3)} \cdots \mathbf{v}^{(k)}$. This is easy to guarantee (with probability one) by picking a random initial condition at each node.

Remark 3.5. Note that the wave equation can also be used as a distributed algorithm for eigenvector and eigenvalue computation of $\mathbf{L}$. From the FFT we can compute ω_j which in turn allows us to compute the eigenvalues λ_j. The eigenvector components are computed using the coefficients of $\cos(t\omega_j)$ (or equivalently $\sin(t\omega_j)$).

Remark 3.6. The algorithm is also attractive from a communication point of view. In Kempe and McSherry (2008) entire matrices need to be passed from one node to another. In our algorithm only scalar quantities $\mathbf{u}_j$ need to be communicated.

Remark 3.7. Peak detection algorithms based on the FFT are typically not very robust because of spectral leakage. As we are only interested in the frequencies corresponding to peaks, algorithms like multiple signal classification (Schmidt, 1986) can overcome these difficulties. The investigation of such algorithms, as well as windowing methods, is the subject of future work.

4. Performance analysis

An important quantity related to the wave equation based algorithm is the time needed to compute the eigenvalues and eigenvectors components. The distributed eigenvector algorithm proposed in Kempe and McSherry (2008) converges at a rate of $O(\tau \log^2(N))$, where τ is the mixing time of the Markov chain associated with the random walk on the graph. We derive a similar convergence bound for the wave equation based algorithm.

It is evident from Eq. (13) that one needs to resolve the lowest frequency to cluster the graph. Let us assume that one needs to wait for η cycles of the lowest frequency to resolve it successfully (i.e. the number of cycles needed for a peak to appear in the FFT).[3] The time needed to cluster the graph based on the wave equation is,

$$T_{\max} = \frac{\eta}{\omega_2}. \tag{15}$$

From Eq. (11) it is easy to see that $\cos(\omega_2) = \text{Real}(\alpha_2) = (2 - c^2\lambda_2)/2$. Note that in Boyd, Diaconis, and Xiao (2004) it was shown that $\tau = -(\log|1 - \lambda_2|)^{-1}$. Thus, it follows that,

$$\omega_2 = \arccos\left(\frac{2 + c^2(e^{-1/\tau} - 1)}{2}\right).$$

Hence, the convergence of the wave equation based eigenvector computation depends on the mixing time of the underlying Markov chain on the graph, and is given by

$$T_{\max} = O\left(\arccos\left(\frac{2 + c^2(e^{-1/\tau} - 1)}{2}\right)^{-1}\right). \tag{16}$$

In the wave equation based clustering computation, one can at the i-th node, compute the i-th component of every eigenvector (along with all the eigenvalues) of the graph Laplacian, thus assigning every node to a cluster.

If one uses the wave equation to compute eigenvectors, to ensure that at every node one has entire eigenvectors, an extra communication step needs to be added. As a final step, locally computed eigenvectors components are transmitted to all other nodes. The cost of this step is $O(N)$ (worst case). Thus, convergence of the distributed eigenvectors computation scales as,

$$T_{\max} = O\left(\arccos\left(\frac{2 + c^2(e^{-1/\tau} - 1)}{2}\right)^{-1}\right) + O(N). \tag{17}$$

Note that simple analysis shows that for large τ our algorithm has a convergence rate of $\sqrt{\tau}/c$ (as $O(N)$ gets dominated by τ). It is interesting to note that in the discretized wave equation, though the constant c loses the meaning of wave speed (that it has in the continuous case) it does impact the speed of convergence.

The convergence of wave equation based clustering is compared to convergence of distributed spectral clustering in Fig. 3, for $c^2 = 1.99$. In particular, the figure shows that wave equation based clustering has, in general, better scaling, with respect to τ, than (Kempe & McSherry, 2008).

Note that the proximity of ω_3 to ω_2 (or the proximity of λ_3 to λ_2) will influence the constant in Eq. (16). The resolution of the FFT is $O(1/K)$, where K is the number of samples. Thus, K has to exceed $1/|\omega_3 - \omega_2|$, to enable computation of two separate peaks. The closer λ_3 is to λ_2, the greater are the number of samples that each node needs to store in order to obtain a good estimate of ω_2 using the FFT. A similar constant depending on the ratio of λ_2 and λ_3 arises in distributed spectral clustering (Kempe & McSherry, 2008) and any power iteration based scheme for eigenvector computation (Golub & Loan, 1996).

Practically, if the lowest frequency of the FFT does not change for a pre-defined length of time, we assume that convergence has been achieved.

[3] The constant η is related to the FFT algorithm and independent of the graph. Typically 6–7 cycles of the lowest frequency are necessary to discriminate it.

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019

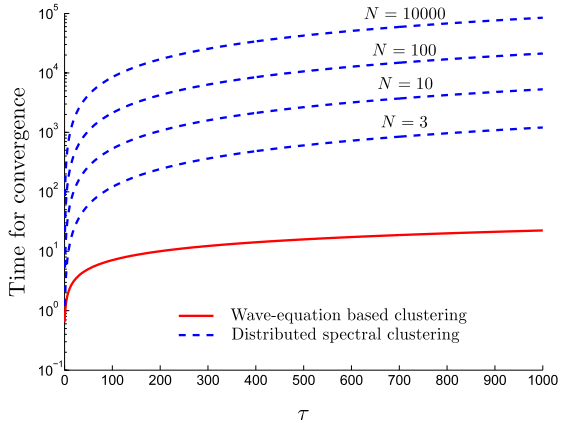


Fig. 3. Comparison of convergence rates between the distributed algorithm in Kempe and McSherry (2008) and our proposed wave equation algorithm for $c^2 = 1.99$. The wave equation based algorithm has better scaling with τ for graphs of any size (given by N). The plots are upper bounds on the convergence speed.

Fig. 4. The ring graph $\mathcal{C}_N$ with N nodes. Every edge has a weight of 1.

From Eq. (16) it seems that the proposed clustering algorithm is independent of the size of the graph (since $\sqrt{\tau}/c$ dominates $O(N)$). This, however, is not true. Larger graphs with low connectivity tend to have higher mixing times. Take for example, a cyclic graph $\mathcal{C}_N$ shown in Fig. 4. We use the cyclic graph as a benchmark as one can explicitly compute the mixing time as a function of N and make a comparison with (Kempe & McSherry, 2008). Of course, no unique spectral cut exists for such a graph. The second eigenvalue of the Laplacian for $\mathcal{C}_N$ is given by

$$\lambda_2 = 1 - \cos\left(\frac{2\pi}{N}\right). \tag{18}$$

Thus, the mixing time of the Markov chain is given by

$$\tau = -\frac{1}{\ln(\cos(2\pi/N))} \approx \left(\frac{N}{2\pi}\right)^2. \tag{19}$$

From Eq. (16), one can show that the time for convergence of the wave equation is,

$$T_{\max} = \frac{\eta}{\arccos(1 + 0.5c^2(\cos(2\pi/N) - 1))} \approx \eta\frac{N}{2\pi}. \tag{20}$$

As expected, Eq. (20) predicts that as the graph becomes larger, the convergence time for the wave equation based algorithm increases. We numerically compute and compare the convergence times for random walks and wave equation on the cyclic graph (by explicitly running the iterations for both processes and checking for convergence). The results are shown in Fig. 5.

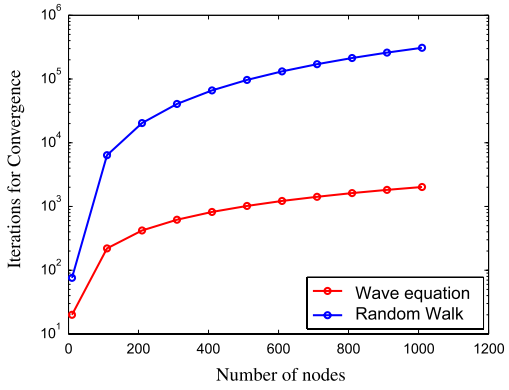


Fig. 5. Convergence of random walk and wave equation on the cyclic graph $\mathcal{C}_N$ as a function of number of nodes, N.

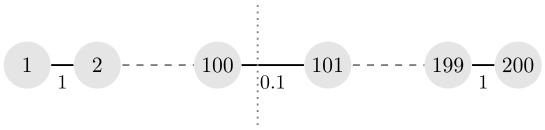


Fig. 6. A line graph with nearest neighbor coupling. The edge between 100 and 101 is a weak connection with weight 0.1, all other edges have weight 1.0. Vertical line shows the predicted cut.

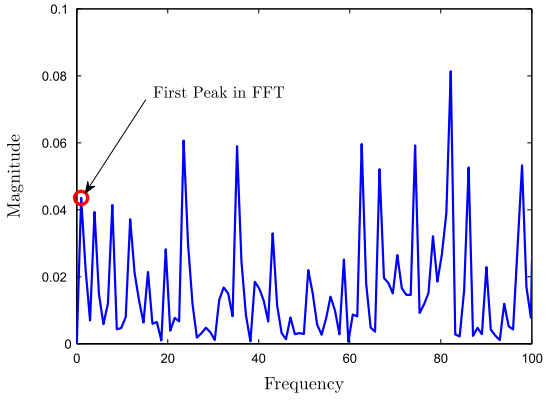


Fig. 7. FFT of $[\mathbf{u}_i(1)\cdots\mathbf{u}_i(T)]$ for any node i of the line graph. Red circle marks the lowest frequency. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5. Numerical results

Since our algorithm should predict the same partitions as spectral clustering, we demonstrate the algorithm on illustrative examples. Our first example, is the simple line graph shown in Fig. 6. Nodes 1–100 and 101–200 are connected to their nearest neighbors with edge weight 1. The edge between nodes 100 and 101 has weight 0.1. As expected, spectral clustering predicts a cut between nodes 100 and 101. We propagate the wave on the graph using update Eq. (6) at every node. At each node, one then performs an FFT on the local history of $\mathbf{u}$. The FFT frequencies are the same for all nodes (evident from Eq. (13)) and shown in Fig. 7. The sign of the coefficients of the lowest frequency in the FFT are shown in Fig. 8. It is evident from this figure that the sign of the coefficients change sign exactly at the location of the weak connection, predicting a cut between nodes 100 and 101 (consistent with spectral clustering).

We now demonstrate our distributed wave equation based clustering algorithm on the Zachary Karate club graph (Zachary, 1977) and on a Fortunato benchmark example (Lancichinetti, Fortunato, & Radicchi, 2008). These social networks are defined by the adjacency matrix that is determined by social interactions. We assume that all the edges have weight 1.

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019

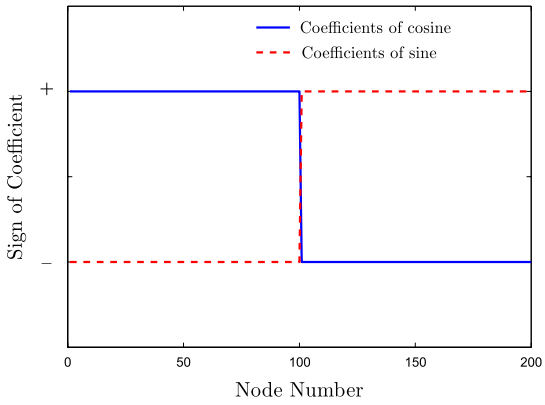


Fig. 8. Signs of the coefficients of the lowest frequency for the line graph.

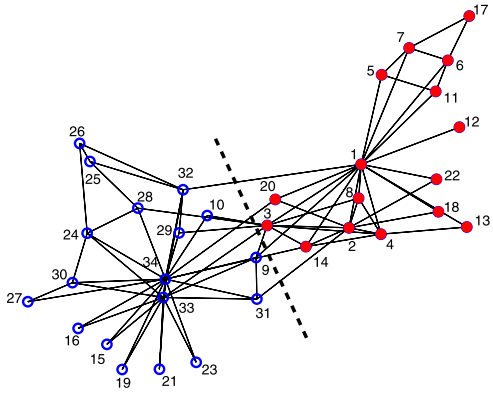


Fig. 9. Graph decompositions predicted by spectral and wave equation based clustering algorithms. Both methods predict the same graph cut.

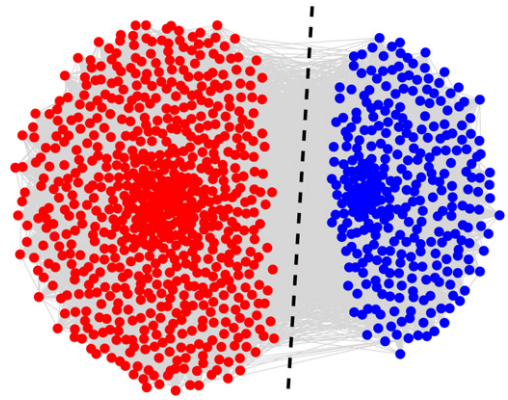


Fig. 10. A Fortunato community detection benchmark with 1000 nodes and 99 084 edges. Wave equation based clustering computes the graph cut exactly.

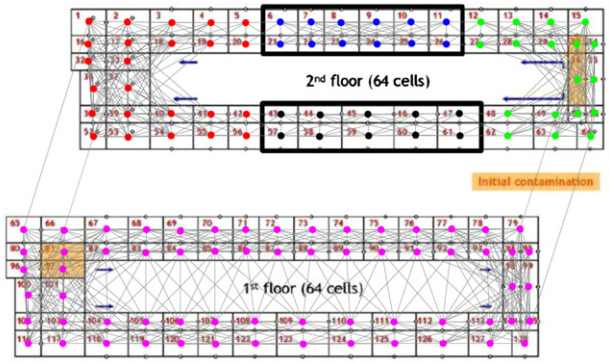


Fig. 11. A two floor building subdivided into 64 cells/rooms for each floor. In each room there is a sensor node capable of communicating with neighbors within a radius of 10 m. The thick black line, depicts walls that degrade communication strength.

Zachary, a sociologist, was studying friendships at a Karate club when it split into two. As expected, members picked the club with more friends. This example serves as an ideal test bed for clustering algorithms. Any effective clustering algorithm is expected to predict the observed schism. Community detection and graph clustering algorithms are routinely tested on this example, see Girvan and Newman (2002), Newman (2006), Porter et al. (2009) and Rosvall and Bergstrom (2007) for a few such demonstrations.

We first apply spectral clustering on this example, then run our wave equation based clustering algorithm, and compare the results in Fig. 9. As expected, we find that both algorithms partition the graph into exactly the same clusters.

We also demonstrate our algorithm on a large Fortunato benchmark with 1000 nodes and 99084 edges. The graph has two natural clusters with 680 and 320 nodes respectively. These clusters are shown in Fig. 10. The wave equation based clustering computes the graph cut exactly.

Thus, wave equation based eigenvector computation can be used to partition both abstract graphs on parallel computers, or physical networks such as swarms of unmanned vehicles, sensor networks, embedded networks or the Internet. This clustering can aid communication, routing, estimation and task allocation.

We now show how clustering can be effectively used to accelerate distributed estimation and search algorithms.

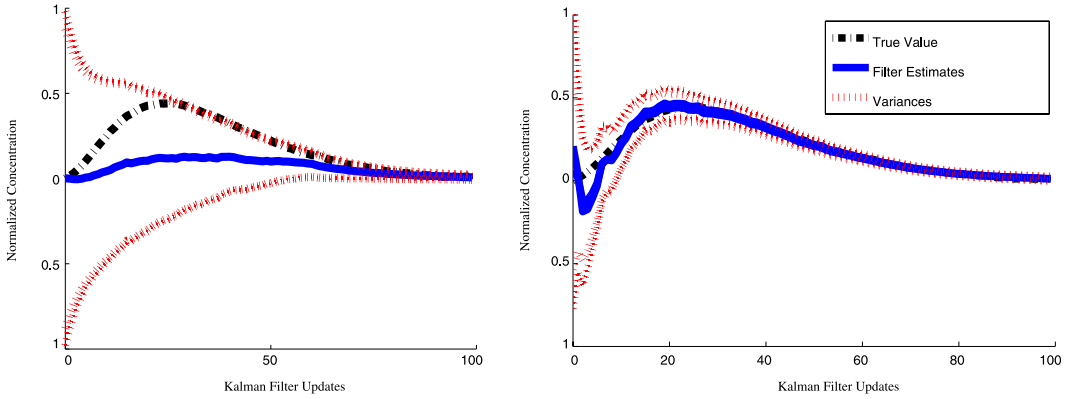
6. Distributed estimation over clusters

Distributed estimation has recently received significant attention see Alrikson and Rantzer (2007), Carli, Chiuso, Schenato, and Zampieri (2008), Olfati-Saber (2007) and Speranzon, Fischione, Johansson, and Sangiovanni-Vincentelli (2008) and references therein. Distributed estimation algorithms require the entire network of sensors to exchange (through nearest neighbor communication) data about the measured variables in order to obtain an overall estimate, which is asymptotically (in the number of iterations) optimal. This results in estimators with error dynamics that converge to zero very slowly. It is well known that these type of algorithm can be accelerated using multi-scale approaches, see for example Kim, West, Lall, Scholte, and Banaszuk (2008b); Kim, West, Scholte, and Narayanan (2008a) and Selle and West (2009). The key idea in these multi-scale approaches is to partition the sensor network into clusters, solve the distributed problem in each cluster and fuse the information between clusters.

As the overall estimation process is distributed, it is desirable that the multi-scale speedup is achieved through a distributed process as well. This means that the clustering must be computed, in a bottom-up fashion, from the structure of the network. We show in the following a simple yet illustrative example, where the wave equation based clustering algorithm can be used to accelerate distributed estimation computation by exploiting properties of the overall sensor network.

We consider the contaminant transport problem in a building (Kim et al., 2008a) with two floors, each divided into 64 cells/rooms (see Fig. 11). A sensor, to detect the contaminant, is present in each cell. Sensors can communicate if their relative distance is less than 10 m. However, we assume that only four sensors can communicate between floors, namely those placed within common staircases connecting the two floors. On the first floor, sensors can communicate across the empty space in between (we assume that windows are present), whereas on the second floor we assume that there are walls that reduce the communication range. We further assume that walls marked with a thick black line, see

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019



(a) Two clusters. (b) Five clusters.

Fig. 12. Concentration estimates for room 49. Concentration estimates (solid line) are compared to the true value (dash–dot line). The dashed lines give the $\pm 3\sigma$ curves around the estimate. As it is clear from the plots, the strategy in which the consensus step is run using five clusters (b) is much better than using two clusters (a).

Fig. 11, degrade communication between the nodes that are inside the area to those outside.

As in Kim et al. (2008a), we assume that the contaminant is produced in four rooms, two on the first floor and two on the second. Under the simplifying assumption of perfect mixing within each cell/room volume, the contaminant propagates within the building (see Kim et al., 2008a for details) according to:

$$\rho_i V_i \frac{dC_i}{dt} = \sum_{i\sim j} F_{ji} C_j - \sum_{i\sim j} F_{ij} C_i + G_i - R_i C_i,$$

ρ : Density C : Contaminant concentration

V : Volume F_{ji} : Mass flow rate from node j to i

G : Contaminant generation rate

R : Mass removal rate.

A constant inward flow of air is introduced at a corner of the second floor, and outflow openings exist wherever windows are open to the outside. We consider a distributed Kalman filter as one that uses consensus to average the estimates and covariance matrices between Kalman filter updates, see Kim et al. (2008a) for details.

The idea of using the wave equation for distributed clustering is to "discover" in a bottom-up fashion, the presence of clusters and exploit strong inter-cluster connectivity to accelerate computation. In particular, we demonstrate the benefit of using the "bottom-up" approach. In the building example there are two main clusters (first and second floors), which a filter and network designer can a-priori assume to know. The four clusters on the second floor, however, would not be known to the designer unless extensive communication measurements are carried out.

In order to determine the four clusters based on SNR, on the second floor, the wave equation based clustering was run for 600 steps. The clustering clearly needs to be run only once, unless there is very strong variation of SNR or the network. In this particular example we assume that the SNR and the network do not vary.

6.1. Numerical results

Numerical results are obtained by running the Kalman filter interleaved with the consensus step, see Kim et al. (2008a). We fix 10 iterations for the consensus step in each cluster. Fig. 12 shows the estimation result for 100 updates of the Kalman filter[4] for both clustering strategies described previously. In particular, Fig. 12(a), and (b) show the estimate (solid line) of the concentration (the true value is shown with dash–dot line) in room 49 made by all the sensors in the building. It can be clearly seen that the estimate in Fig. 12(a) is not as accurate as the one in Fig. 12(b). The reason is that the consensus step for the case of four clusters on the second floor converges much faster to the true covariance compared to the case of two clusters. In comparison, if consensus is run for the case of two clusters, it requires more than 500 iterations in each consensus step to converge to the accuracy of Fig. 12(b).

In the 5 cluster case, all the nodes in the building have accurate estimates of the contaminant concentration for rooms located on the first floor. This is because sensors on the first floor are strongly connected to one another and 10 iterations are enough to converge to the true covariance (with only slight corruption by the "unconverged" averaging on the second floor).

These simulations show that the wave equation based clustering provides an efficient distributed bottom-up methodology for partitioning sensor networks and accelerating distributed estimation algorithms.

7. Mobile sensor networks

We demonstrate the utility of distributed partitioning for computing the trajectories of mobile sensors/vehicles for the purpose of efficiently searching a large area. In Mathew and Mezic (2009) the authors have developed an algorithm to optimally search a region, given a prior distribution that models the likelihood of finding the target in any given location (see for example Fig. 13). The trajectories are computed using a set of ordinary differential equations given by

$$\dot{x}_j(t) = u_j(t). \tag{21}$$

The above equation describes the dynamics of the j-th vehicle, where $x_j(t)$ and $u_j(t)$ are the position and the control input of the j-th vehicle at time t respectively. The authors prove that the control law

$$u_j(t) = -u_{\max} \frac{B_j(t)}{\|B_j(t)\|}, \tag{22}$$

efficiently samples the prior distribution for search. Here,

$$B_j(t) = \sum_k \frac{\Lambda_k S_k(t) \nabla f_k(x_j(t))}{\langle f_k, f_k \rangle}, \tag{23}$$

where f_k are the Fourier basis functions that satisfy the Neumann boundary conditions on the domain to be searched and k is the

[4] We assume that the consensus step is fast compared to the contaminant spreading so that no compensation of delay is required at the nodes while running the Kalman filter. It is clear that for estimation, shortening the consensus step is crucial in order to have a consistent estimate.

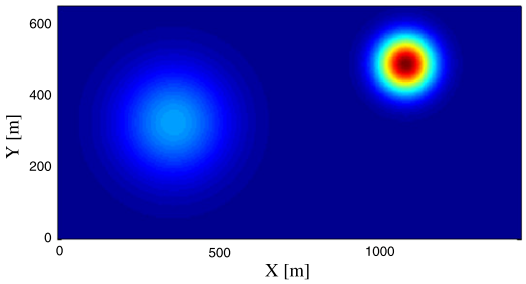


Fig. 13. Prior belief map (distribution) for targets.

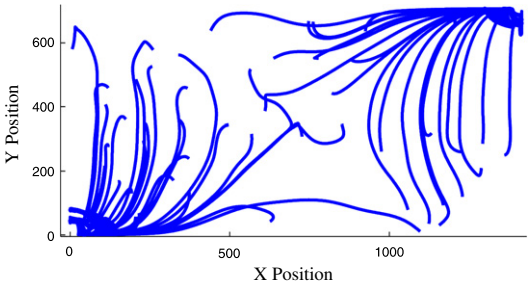


Fig. 14. Trajectories generated using distributing spectral search algorithm that uses wave equation based clustering.

corresponding basis vector number. The quantities $S_k(t)$ are governed by the following differential equation,

$$\frac{dS_k(t)}{dt} = \frac{\sum_{j=1}^{N} f_k(x_j(t))}{\langle f_k, f_k \rangle} - N\mu_k, \tag{24}$$

where N is the number of vehicles.

In Mathew and Mezic (2009) the trajectories are computed a-priori for a given distribution (belief map), using Eqs. (21)–(24). Here we perform online computations for trajectories generation in a distributed setting. The sum $\sum_{j=1}^{N} f_k(x_j(t))$ over all vehicles in Eq. (24) is the centralized quantity that needs to be computed in a distributed manner. At every time instant (every time step of the Runge Kutta scheme), the vehicles are partitioned into groups using the wave equation based clustering algorithm and the sum in Eq. (24) is computed over the clusters and the solutions added. All the vehicles then compute a piece of their trajectory for a predetermined horizon of time (for a single Runge Kutta time step). These pieces of trajectories for each agent are merged together to give Fig. 14. In this way, the mobile sensors group themselves into clusters and compute their trajectories in a distributed manner.

8. Conclusions

In this work, we have constructed a wave equation based algorithm for computing the clusters in a graph. The algorithm is completely distributed and at every node one can compute cluster assignments based solely on local information. In addition, this algorithm is orders of magnitude faster than state-of-the-art distributed eigenvector computation algorithms. Starting from a random initial condition at every node, one evolves the wave equation and updates the state based solely on the scalar states of neighbors. One then performs an FFT at each node and computes the sign of the components of the eigenvectors of the graph Laplacian. Complete eigenvector information can be transmitted to each node using multi-hop communication. This process is equivalent to spectral clustering.

The algorithm is also attractive from a distributed computing point of view, where parallel simulations of large dynamical systems (Klus, Sahai, Liu, & Dellnitz, 2011) can be coupled to the distributed clustering approach presented here, to provide scalable solutions for problems that are computationally and theoretically intractable. This application is the subject of current research.

Wave equation based clustering is demonstrated on community detection examples. Applications to multi-scale distributed estimation and distributed search are also demonstrated.

Current work includes the extension of the wave equation based algorithm for dynamic networks. This is clearly very important in situations where the weights on the edges of the graph are time varying. Examples of systems where dynamic graphs arise include UAV swarms, nonlinear dynamical systems and evolving social graphs.

Acknowledgments

The authors thank José Miguel Pasini, Alessandro Pinto and Pablo Parrilo for valuable discussions and suggestions. The authors also thank Andrea Lancichinetti for providing the Fortunato benchmark graph. The authors would also like to thank the anonymous reviewers for the constructive comments that helped to improve the paper.

References

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications Magazine*, *40*(8), 102–114.
Alrikson, P., & Rantzer, A. (2007). Experimental evaluation of a distributed Kalman filter algorithm. In *IEEE conference on decision and control*.
Belkin, M., & Niyogi, P. (2008). Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, *74*, 1289–1308.
Boyd, S., Diaconis, P., & Xiao, L. (2004). Fastest mixing Markov chain on a graph. *SIAM Review*, *46*, 667–689.
Carli, R., Chiuso, A., Schenato, L., & Zampieri, S. (2008). Distributed Kalman filtering based on consensus strategies. *IEEE Journal on Selected Areas in Communications*, *26*, 622–633.
Chung, F. R. K. (1997). *Spectral graph theory* (1st ed.) American Mathematical Society.
Coifman, R. R., Shkolnisky, Y., Sigworth, F. J., & Singer, A. (2008). Graph Laplacian tomography from unknown random projections. *IEEE Transactions on Image Processing*, *17*(10), 1891–1899.
Evans, L. C. (1998). *Partial differential equations*. American Mathematical Society.
Fiedler, M. (1973). Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, *23*, 289–305.
Fiedler, M. (1975). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, *25*(4), 619–633.
Franceschelli, M., Gasparri, A., & Seatzu, A. G. C. (2009). Decentralized Laplacian eigenvalues estimation for networked multi-agent systems. In *Proceedings of the conference on decision and control*.
Friedman, J., & Tillich, J. P. (2004). Wave equations for graphs and edge-based Laplacians. *Pacific Journal of Mathematics*, *216*(2), 229–266.
Speer, N., Fröhlich, H., Spieth, C., & Zell, A. (2005). Functional grouping of genes using spectral clustering and gene ontology. In *IEEE international joint conference on neural networks. Vol. 1* (pp. 298–303).
Ghiasi, S., Srivastava, A., Yang, X., & Sarrafzadeh, M. (2002). Optimal energy aware clustering in sensor networks. *Sensors*, *2*, 258–269.
Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826.
Golub, G., & Loan, C. V. (1996). *Matrix computations*. John Hopkins University Press.
Hein, M. (2006). Uniform convergence of adaptive graph-based regularization. In *Proceedings of the 19th annual conference on learning theory* (pp. 50–64). Springer.
Hein, M., Audibert, J.-Y., & von Luxburg, U. (2005). From graphs to manifolds—weak and strong pointwise consistency of graph Laplacians. In *Proceedings of the 18th annual conference on learning theory* (pp. 470–485). Springer.
Herman, I., Melançon, G., & Marshall, M. S. (2000). Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics*, *6*(1), 24–43.
Kac, M. (1966). Can one hear the shape of a drum? *The American Mathematical Monthly*, *73*(4), 1–23.
Kempe, D., & McSherry, F. (2008). A decentralized algorithm for spectral analysis. *Journal of Computer and System Sciences*, *74*(1), 70–83.
Kernighan, B. W., & Lin, S. (1970). A efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, *49*, 291–307.
Kim, J. -H., West, M., Lall, S., Scholte, E., & Banaszuk, A. (2008b). Stochastic multiscale approaches to consensus problems. In *Conference on decision and control*.
Kim, J. -H., West, M., Scholte, E., & Narayanan, S. (2008a). Multiscale consensus for decentralized estimation and its application to building systems. In *American control conference*.

Klus, S., Sahai, T., Liu, C., & Dellnitz, M. (2011). An efficient algorithm for the parallel solution of high-dimensional differential equations. *Journal of Computational and Applied Mathematics*, *235*(9), 3053–3062.
Kottak, C. P. (1991). *Cultural anthropology* (5th ed.) McGraw-Hill.
Lancichinetti, A., Fortunato, S., & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review E*, *78*, 046110.
Luxburg, U. v., Bousquet, O., & Belkin, M. (2004). On the convergence of spectral clustering on random samples: the normalized case. In *Proceedings of the 17th annual conference on learning theory. COLT* (pp. 457–471).
Mathew, G., & Mezic, I. (2009). Spectral multiscale coverage: a uniform coverage algorithm for mobile sensor networks. In *Proceedings of the conference on decision and control* (pp. 7872–7877).
Mohar, B. (1991). The Laplacian spectrum of graphs. Department of mathematics, Simon Fraser university. *Tech. rep*. Canada.
Muhammad, A., & Jadbabaie, A. (2007). Distributed computation of homology groups by gossip. In *Proceedings of the american control conference* (pp. 3438–3443).
Nadler, B., Lafon, S., Coifman, R. R., & Kevrekidis, I. G. (2006). Diffusion maps, spectral clustering and eigenfunctions of Fokker–Planck operators. *Advances in Neural Information Processing Systems*, *18*.
Nadler, B., Lafon, S., Coifman, R. R., & Kevrekidis, I. G. (2006). Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, *21*, 113–127. (special issue on) Diffusion maps and wavelets.
Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*, 8577–8582.
Olfati-Saber, R. (2007). Distributed Kalman filtering for sensor networks. In *IEEE conference on decision and control*.
Paccanaro, A., Casbon, J. A., & Saqi, M. A. S. (2006). Spectral clustering of protein sequences. *Nucleic Acids Research*, *34*, 1571–1580.
Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, *435*, 814–818.
Porter, M. A., Onnela, J.-P., & Mucha, P. J. (2009). Communities in networks. *Notices of the American Mathematical Society*,.
Reichardt, J., & Burnholdt, S. (2004). Detecting fuzzy community structures with a Potts model. *Physical Review Letters*, *93*, 218701.
Rosvall, M., & Bergstrom, C. T. (2007). An information—theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, *104*(18), 7327–7331.
Schmidt, R. O. (1986). Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation*, *AP-34*(3), 276–280.
Selle, C., & West, M. (2009). Multiscale networks for distributed consensus algorithms. In *IEEE conference on decision and control and Chinese control conference*.
Shah, D. (2009). *Gossip algorithms*. Now Publisher Inc..
Speranzon, A., Fischione, C., Johansson, K. H., & Sangiovanni-Vincentelli, A. (2008). A distributed minimum variance estimator for sensor networks. *IEEE Journal on Selected Areas in Communications*, *26*, 609–621.
Spielman, D. A., & Teng, S. -H. (2004). Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on theory of computing* (pp. 81–90).
Stoer, M., & Wagner, F. (1997). A simple min-cut algorithm. *Journal of the ACM*, *44*(4), 585–591.
Varigonda, S., Kalmar-Nagy, T., Labarre, B., & Mezic, I. (2004). Graph decomposition methods for uncertainty propagation in complex, nonlinear interconnected dynamical systems. In *IEEE conference on decision and control. Vol. 2* (pp. 1794–1798).
von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, *17*, 395–416.
Wagner, D., & Wagner, F. (1993). Between min cut and graph bisection. In *Proceedings of the 18th international symposium on mathematical foundations of computer science. MFCS* (pp. 744–750).
Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, *33*, 452–473.

Tuhin Sahai is Research Scientist at the United Technologies Research Center. His research interests lie in the broad areas of design, analysis and verification of continuous and discrete dynamical systems. He is particularly interested in high dimensional systems that tend to be analytically and computationally intractable. Tuhin obtained his Ph.D. from the Department of Theoretical and Applied Mechanics at Cornell University in January 2008.

Alberto Speranzon received the "Laurea" degree in computer engineering from University of Padova, Italy in 2000, the Tech. Lic. and Ph.D. in automatic control from the School of Electrical Engineering, Royal Institute of Technology, Stockholm, Sweden in 2004 and 2006, respectively. He is currently a Research Scientist at the United Technologies Research Center, in East Harford, CT, USA. Between October 2006 and September 2008 he was a Marie Curie Research Fellow at Unilever R and D, Port Sunlight, UK. From February to June 2005, he was visiting research scholar at Department of Information Engineering, University of Padova, Italy. During the period June to October 2006, he was postdoctoral researcher at the Automatic Control Lab, Royal Institute of Technology. His research interests are mainly in the area of distributed control, estimation and optimization, with particular focus on multi-vehicle systems and wireless sensor networks as well as hybrid and discrete-event systems.

Andrzej Banaszuk is a Fellow and a Program Leader of Sikorsky Program Office at United Technologies Research Center. Since joining UTRC in 1997, he has conducted research in analysis, design, and control of dynamical systems applied to jet engines, rotorcraft, unmanned systems, electric power networks, and buildings. Since 2000 he has led collaborative multi-university research teams in the area of flow control, control of combustion instabilities, and robust design of large uncertain dynamic networks. He is an author of 41 journal papers, 70 conference papers, and 6 patents. From 1999 to 2002, he was an Associate Editor of IEEE Transactions of Controls Systems Technology. He was appointed to serve on the Board of Governors of IEEE Control Systems Society in 2004. For his work on active and passive control of flow instabilities in jet engines he received IEEE Controls Systems Technology Award in 2007. He became an IEEE Fellow in 2011. He holds Ph.D. in EE from Warsaw University of Technology and in Mathematics from Georgia Institute of Technology.

Please cite this article in press as: Sahai, T., et al. Hearing the clusters of a graph: A distributed algorithm. Automatica (2011), doi:10.1016/j.automatica.2011.09.019