Random Node-Asynchronous Updates on Graphs

Oguzhan Teke, *Student Member, IEEE*, and Palghat P. Vaidyanathan, *Fellow, IEEE*

***Abstract*—This paper introduces a node-asynchronous communication protocol in which an agent in a network wakes up randomly and independently, collects states of its neighbors, updates its own state, and then broadcasts back to its neighbors. This protocol differs from consensus algorithms and it allows distributed computation of an arbitrary eigenvector of the network, in which communication between agents is allowed to be directed. (The graph operator is still required to be a normal matrix). To analyze the scheme, this paper studies a random asynchronous variant of the power iteration. Under this random asynchronous model, an initial signal is proven to converge to an eigenvector of eigenvalue 1 (a fixed point) even in the case of operator having spectral radius larger than unity. The rate of convergence is shown to depend not only on the eigenvalue gap but also on the eigenspace geometry of the operator as well as the amount of asynchronicity of the updates. In particular, the convergence region for the eigenvalues gets larger as the updates get less synchronous. Random asynchronous updates are also interpreted from the graph signal perspective, and it is shown that a non-smooth signal converges to the smoothest signal under the random model. When the eigenvalues are real, second order polynomials are used to achieve convergence to an arbitrary eigenvector of the operator. Using second order polynomials the paper formalizes the node-asynchronous communication model. As an application, the protocol is used to compute the Fiedler vector of a network to achieve autonomous clustering.**

***Index Terms*—Autonomous networks, node asynchronous updates, fixed point iteration, randomized iterations, smooth graph signals, polynomial filters.**

I. Introduction

RECENT years have witnessed an elevated interest in network structured data where the underlying graph is assumed to model the relation between the data sources. This is a very broad model that can be found in a variety of different contexts such as social, economic, and biological networks, among others [1], [2]. Analysis of such networked models is not new, and can be traced back to the consensus problem studied more than four decades ago [3]. The recent advancements in [4]–[7] studied the networked data (or, signals defined over graphs) from the signal processing point of view, in which the analysis is based on the "graph operator" whose eigenvectors serve as the graph Fourier basis (GFB). Examples of graph operators include the Laplacian matrices [4], the adjacency matrix [5], or a signal covariance matrix [8]. With the help of GFB, a number of topics such as sampling and reconstruction, multirate filter banks, and uncertainty principles have been extended to the case of graphs in [9]–[16]. Studies in [17]–[19] also use the graph operator for regression and learning over graphs.

Although eigenvectors of the graph operator provide a useful basis to analyze signals defined over the underlying network, graph Fourier transform appears to be an off-line procedure used only for analysis purposes. The main reason for this is the fact that the computation of eigenvectors (hence, the graph Fourier transform) requires the knowledge of the graph operator in the first place. This poses a significant limitation for an autonomous network, in which there is no centralized control mechanism (e.g. a base station). Examples include wireless ad-hoc networks [20], brain networks [21], and networked robots [22]. In these scenarios, agents (nodes) are assumed to know only their immediate neighbors, and the global structure of the network (graph) is not known. Thus, a computation that depends on eigenvectors directly (graph Fourier transform, spectral clustering, spectral coloring etc.) cannot be accomplished by the agents themselves.

The main goal of this study is to construct a communication protocol under which an arbitrary eigenvector of the underlying graph operator can be found by the agents. In this protocol, agents communicate with each other in an asynchronous manner. More precisely, we consider the following collect-compute-broadcast scenario: states of the agents as a whole will be considered as a graph signal. At a random time instance, an agent wakes up independently and collects states of its neighbors. Then, the agent updates its own state as a linear combination of the received data, which is assumed to be described precisely by the graph operator. Then, the agent broadcasts the amount of change in its state to its neighbors. It is important to emphasize three points regarding this scenario: 1) The signal over the network is driven only by the initial conditions (and the graph operator). Agents do not take measurements, they only exchange data (their states) between each other. 2) This is an iterative scheme, and the graph signal converges to the desired eigenvector through repeated communications. 3) Unlike an edge-asynchronous protocol, in which only a connected pair of agents communicate with each other, the scenario we consider here is node-asynchronous where an agent wakes up randomly and communicates with all of its neighbors.

A. Assumptions on the Graph Operator

In this study we will not require the graph operator to be a symmetric matrix, that is, edges in the network are allowed to be directed, possibly with unequal edge weights. Self-loops are also allowed. However, we do require the operator to be a normal matrix (equivalently, a unitary-diagonalizable matrix). We note that

Manuscript received April 14, 2018; revised October 7, 2018 and March 30, 2019; accepted April 1, 2019. Date of publication April 11, 2019; date of current version April 26, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mats Bengtsson. This work was supported in part by the Office of Naval Research under Grant N00014-18-1-2390, in part by the National Science Foundation under Grant CCF-1712633, and in part by the Electrical Engineering Carver Mead Research Seed Fund of the California Institute of Technology. *(Corresponding author: Oguzhan Teke.)*

The authors are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: oteke@caltech.edu; ppvnath@systems.caltech.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TSP.2019.2910485

symmetric (Hermitian) matrices are necessarily normal. Thus, results here are applicable to any graph with undirected edges. On the contrary, an arbitrary directed graph may not have a normal graph operator; it may even be non-diagonalizable (like the cases studied in [23]).

We assume that the graph operator (hence, the underlying network) is not time dependent. Only the node behaviors are time-varying (in a random fashion), but not their connectivity structure. We also assume that each node knows its neighbors.

B. Connections With Asynchronous Fixed Point Iterations

In this paper we will study the described node-asynchronous network model from a fixed point iteration view-point, in which the notion of graph shift (introduced in graph signal processing [7]) will be considered as the update step. In this approach, successively graph shifted signals can be examined as a sequence generated by a fixed point algorithm similar to the well-known power method. However, the notion of graph shift is not directly applicable to the node-asynchronous model considered here since a graph shift requires all the nodes to communicate at the same time instance, which contradicts with the asynchronicity assumption. In order to tackle this problem, we will focus on an asynchronous variant, in which only a subset of indices are updated in each iteration.

We note that asynchronous fixed point iterations are well-studied problems [24]. The papers in [25], [26] presented important convergence results for general non-linear update models. In fact, the first analysis of the problem can be traced back to the study in [27] (and references therein) under the name "chaotic relaxation," in which a linear model with an input was investigated. In these studies the convergence is guaranteed mainly under two assumptions: 1) The update step is a contraction. 2) The indices are updated frequently enough. (See [25]–[27] for precise descriptions of these assumptions.)

In our study, we will consider linear updates without an input (power method on the graph operator) in a probabilistic setting in which a *random* subset of indices is updated in each iteration. In this setting, we prove that iterations converge to a fixed point of the graph operator in the mean-squared sense even in the case of the graph operator not being a contraction. More precise conditions will be spelled out later.

It is also important to note that the random asynchronous variant of the power method studied here can be thought of as a special case of coordinate descent algorithms [28]. In particular, coordinate-wise (or, asynchronous as we refer here) power iteration was studied recently in [29], [30]. Both techniques are demonstrated to perform well on data sets when computing the dominant eigenvectors, however they are not directly applicable to the autonomous network model we consider here. Furthermore, our results show that iterations do not necessarily converge to the dominant eigenvector, but they converge to an eigenvector of the unit eigenvalue (a fixed point) *even if there are other eigenvalues with magnitudes greater than unity.*

C. Connections With Gossip Algorithms and Consensus

The most common form of gossip protocols assumes that an adjacent pair of nodes share their current states with each other at random time instances and update their states by an averaging [31]. Due to their versatility, gossip-like algorithms have been studied extensively for distributed parameter estimation and optimization problems [31]–[39]. Further examples include push-sum [40], [41] and subgradient-push [42] (where the updates are synchronous, but the network is time-varying.) We refer the reader to [43]–[46] for more general distributed/asynchronous optimization problems.

Although gossip protocols allow asynchronous communications, the canonical examples have edge-asynchronous (or, random link) behavior in which an edge gets activated randomly and a pair of nodes, linked by the edge, communicate with each other. See [47] for a treatment in random filtering in the context of graph signal processing. On the contrary, the model we consider here is node-asynchronous, that is, a node wakes up randomly and communicates with all of its neighbors. Thus, our analysis here deviates from the known results on gossip protocols.

More importantly, gossip protocols are designed in such a way that nodes reach a consensus. Depending on the problem formulation, the value of the consensus may be the estimated parameter(s), or the optimal solution of the objective function in general. For the consensus, graph operators, e.g., average connectivity matrix, the averaging matrix etc., have the property that the constant vector is the unique fixed point of the operator (assuming the graph is connected [31]). A signal over a network operating under a gossip protocol converges to the constant vector, which means that nodes reach a consensus. For example, the study in [48] searches for the optimal graph operator (for the fastest distributed averaging) under the constraint that the operator has constant vector as an eigenvector of eigenvalue 1. Differently in this study, we assume that the given graph operator has a fixed point, i.e., eigenvalue 1 exists, and show that randomized node-asynchronous updates converge to an eigenvector of eigenvalue 1 (a fixed point). *However, the eigenvector need not be a constant vector.* Thus, nodes may not reach a consensus. We will also show how to use polynomial filters in order to obtain convergence to an *arbitrary eigenvector* of the given graph operator. From this perspective, the problem we consider is more general than the consensus. In fact, the consensus can be considered as a particular instance of the problem studied here, in which the graph operator has the constant vector as an eigenvector.

D. Outline and Contributions

In Section II we first define the deterministic asynchronous update. This scheme is like the synchronous power iteration but values of only a subset of indices are updated. Afterwards, we consider the case where the update sets are selected at random. We impose a statistical model on the asynchronous updates, and derive, for later use, the expected value of the random update mechanism (Lemma 1). In Section III we consider cascades of random updates, and first analyze the expected signal in terms of the eigenvectors of the operator (Theorem 1). In order to prove the convergence of the updates, we consider the residual signal and bound its expected squared ℓ_2-norm (Theorem 2). Using this result we provide a sufficient condition (Corollary 2) and a necessary condition (Corollary 4) on the operator such that the signal is guaranteed to converge to an eigenvector of the unit eigenvalue through random updates in the mean-squared sense. We also show that asynchronous updates are better than the synchronous ones in terms of the convergence region of the eigenvalues (Corollary 3). In Section IV we demonstrate how the eigenspace geometry of the operator plays a role in the

convergence of the random asynchronous updates. In Section V we consider the problem from the graph signal processing point of view (Theorem 3). Since an arbitrary nonzero signal is proven to converge to the steady state, the signal in this state is said to be a "typical graph signal." We can interpret the typical signal as the smoothest signal on the graph (with respect to the graph operator). In Section VI we consider polynomials of the graph operator in order to make the iteration converge to other eigenvectors (corresponding to non unit eigenvalues) (Theorem 4). By an explicit construction, we prove that second order polynomials are sufficient to achieve this purpose in the case of real eigenvalues (Theorem 5). Then, in Section VI-D, we formally present the node-asynchronous communication protocol (Algorithm 1) that implements a second order polynomial of the graph operator. In Section VII we use the proposed algorithm to compute the Fiedler vector of a network in order to achieve an autonomous clustering via localized communication.

E. Preliminaries and Notation

We will assume that $\mathbf{A} \in \mathbb{C}^{N\times N}$ is an operator on the graph with N nodes. We consider $\mathbf{A}$ to be a *local* operator, that is, $A_{i,j} = 0$ when the nodes i and j are not neighbors. In particular, $A_{i,j}$ denotes the weight of the edge from the j^{th} node to the i^{th} node. We allow $A_{i,i}$ to be non-zero. Hence, the operator $\mathbf{A}$ can be the adjacency matrix, the Laplacian, the normalized Laplacian, and so on. We will use $\mathcal{N}_{\text{in}}(i)$ and $\mathcal{N}_{\text{out}}(i)$ to denote the incoming and outgoing neighbors of the i^{th} node. More precisely we have:

$$\mathcal{N}_{\text{in}}(i) = \{j \mid A_{i,j} \neq 0\}, \qquad \mathcal{N}_{\text{out}}(i) = \{j \mid A_{j,i} \neq 0\}. \quad (1)$$

We always assume that $\mathbf{A}$ is a normal matrix, i.e., $\mathbf{A}\mathbf{A}^{\text{H}} = \mathbf{A}^{\text{H}}\mathbf{A}$, where $\mathbf{A}^{\text{H}}$ denotes the conjugate transpose of $\mathbf{A}$. Eigenvalue decomposition of $\mathbf{A}$ will be denoted as:

$$\mathbf{A} = \mathbf{V}\,\mathbf{\Lambda}\,\mathbf{V}^{\text{H}}, \quad (2)$$

where $\mathbf{V}$ is a unitary matrix consisting of eigenvectors of $\mathbf{A}$, and $\mathbf{\Lambda}$ is the diagonal matrix with the eigenvalues, which may be complex in general. Given a signal $\mathbf{x}$, its graph Fourier transform (GFT), $\widehat{\mathbf{x}}$, on the operator $\mathbf{A}$ is defined as:

$$\widehat{\mathbf{x}} = \mathbf{V}^{\text{H}}\,\mathbf{x}, \quad \text{or,} \quad \mathbf{x} = \sum_{i=1}^{N} \widehat{x}_i\,\mathbf{v}_i, \quad (3)$$

where $\mathbf{v}_i$'s are the eigenvectors of $\mathbf{A}$.

We will use $\mathbb{P}[\cdot]$ and $\mathbb{E}[\cdot]$ to denote the probability and expectation, respectively. For a vector $\mathbf{x}$ we will use $\|\mathbf{x}\|_\infty$ to denote its largest element in absolute sense. Positive semi-definite ordering will be denoted by $\preccurlyeq$. For a matrix $\mathbf{X}$ we will use $\|\mathbf{X}\|_\infty$ to denote the largest absolute row-sum, $\|\mathbf{X}\|_{\max}$ to denote its largest element magnitude-wise, and $\text{tr}(\mathbf{X})$ to denote its trace. We will use $\text{diag}(\mathbf{X})$ to denote the diagonal masking of $\mathbf{X}$, that is, $(\text{diag}(\mathbf{X}))_{i,i} = X_{i,i}$ and $(\text{diag}(\mathbf{X}))_{i,j} = 0$ for $i \neq j$. We will use $\mathcal{T}$ to denote a subset of $\{1,\ldots,N\}$, and its size is denoted as $t = |\mathcal{T}|$. We will use the notation $\sum_{\mathcal{T}}$ to denote the summation over all subsets of $\{1,\ldots,N\}$ of a fixed size t where the value of t should follow from the context. We will use $\mathbf{D}_{\mathcal{T}} \in \mathbb{R}^{N\times N}$ to denote a diagonal matrix that has value 1 only at the indices specified by the set $\mathcal{T}$. That is,

$$\mathbf{D}_{\mathcal{T}} = \sum_{i\in\mathcal{T}} \mathbf{e}_i\,\mathbf{e}_i^{\text{H}}, \quad (4)$$

where $\mathbf{e}_i \in \mathbb{R}^N$ is the i^{th} standard vector that has 1 at the i^{th} index and 0 elsewhere. We will use $\mathbb{1}$ to denote a vector of all 1's. Further notice that $\mathbf{D}_{\mathcal{T}}$ satisfies the following identities for a given size t:

$$\sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}} = \binom{N-1}{t-1}\,\mathbf{I}, \quad (5)$$

$$\frac{1}{\binom{N}{t}} \sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}}\,\mathbf{A}\,\mathbf{D}_{\mathcal{T}} = \frac{t\,(N-t)\,\text{diag}(\mathbf{A}) + t\,(t-1)\,\mathbf{A}}{N(N-1)}, \quad (6)$$

which will be used in the subsequent proofs.

II. Asynchronous Power Iteration

Given a matrix of interest $\mathbf{A}$ and an initial signal $\mathbf{x}_0$ the conventional power iteration has the following form:

$$\mathbf{x}_k = \mathbf{A}\,\mathbf{x}_{k\text{-}1}, \quad \text{so that} \quad \mathbf{x}_k = \mathbf{A}^k\,\mathbf{x}_0, \quad (7)$$

where the updates here are considered without normalizing the signal at each iteration. Normalization is avoided here intentionally to preserve the local nature of the updates as will be elaborated next.

In the context of graph signal processing, the matrix $\mathbf{A}$ is assumed to be a local graph operator (shift matrix) and the signal $\mathbf{A}\mathbf{x}$ is referred to as the shifted version of $\mathbf{x}$ on the graph [7]. From this perspective $\mathbf{x}_k$ in (7) is the graph shifted version of $\mathbf{x}_{k\text{-}1}$. Since $\mathbf{A}$ is assumed to be a local operator a single shift can be implemented on the graph as a data exchange between the neighboring nodes. That is,

$$(\mathbf{x}_k)_i = \sum_{j\in\mathcal{N}_{\text{in}}(i)} A_{i,j}\;(\mathbf{x}_{k\text{-}1})_j \qquad \forall\, i. \quad (8)$$

Notice that a norm of the signal depends on values of all of the nodes in the graph. Therefore, a norm cannot be known locally in the graph setting, which is why we have avoided normalization in (7).

Although a "graph shift" can be performed locally, the model in (7) forces all the nodes to send and receive data at the same time. Therefore, the graph shift does not have an autonomous implementation since it requires a centralized timing mechanism (synchronization) over the underlying graph.

In this study we will consider a variation of the power iteration, in which not all but a subset of indices, denoted by $\mathcal{T}$, are updated simultaneously and the remaining ones stay unchanged. More precisely, given an update set $\mathcal{T}$ we consider the following *asynchronous* (coordinate-wise) power iteration:

$$y_i = \begin{cases} (\mathbf{A}\mathbf{x})_i, & i \in \mathcal{T}, \\ x_i, & i \notin \mathcal{T}, \end{cases} \quad (9)$$

where $\mathbf{x}$ is the vector before update, and $\mathbf{y}$ is the vector after the update. In words, this update computes the multiplication $\mathbf{A}\mathbf{x}$, but it only updates the values of the elements indexed by the set $\mathcal{T}$, and keeps the remaining elements the same. In short, (7) is a "synchronous" update, whereas (9) is asynchronous. Both (7) and (9) are also referred to as *state recursions*, where the graph signal $\mathbf{x}$ is regarded as the state of a system. The model in (9) was also studied in [25]–[27] with slight differences. Furthermore, (9) is reminiscent of the Hopfield neural network [49], with the difference that there is no nonlinearity in (9). (Studies in [25], [26] have non-linearity.)

The asynchronous update defined in (9) can be written as a matrix-vector multiplication as follows:

$$\mathbf{y} = \sum_{i \notin \mathcal{T}} \mathbf{e}_i \, \mathbf{e}_i^{\mathrm{H}} \, \mathbf{x} + \sum_{i \in \mathcal{T}} \mathbf{e}_i \, \mathbf{e}_i^{\mathrm{H}} \, \mathbf{A} \, \mathbf{x} = \mathbf{Q}(\mathcal{T}) \, \mathbf{x}, \qquad (10)$$

where $\mathbf{Q}(\mathcal{T})$ is the matrix representing the asynchronous update on a set $\mathcal{T}$, and it can be written as follows:

$$\mathbf{Q}(\mathcal{T}) = \mathbf{I} + \mathbf{D}_{\mathcal{T}} \, (\mathbf{A} - \mathbf{I}). \qquad (11)$$

In the next few sections, where we perform a convergence analysis, $\mathbf{A}$ can be treated as a generic matrix without considering specific relations to graphs. The relation to graph signals will be considered in Section V. When the model in (9) is implemented on a graph (i.e., $\mathbf{A}$ is a graph operator), only the nodes in the update set $\mathcal{T}$ need to be synchronized. If the update set is selected as $\mathcal{T} = \{1, \ldots, N\}$, then $\mathbf{Q}(\mathcal{T}) = \mathbf{A}$. That is, the asynchronous update in (9) reduces to the classical synchronous update (graph shift) in (7). On the other extreme, if a single node is updated, $|\mathcal{T}| = 1$, then *no synchronization is required at all* and the nodes are allowed to behave autonomously. We would like to note that the relation in (8) appears as if a node collects states of its neighbors, update its own state, and sits still. However, as we shall describe later in Section VI, we will consider the updates on a polynomial of the given graph operator, which will require the nodes to follow a collect-compute-broadcast scheme. These details will be elaborated in Section VI-D.

A. Randomized Asynchronous Updates

In this paper we will study the behavior of a cascade of asynchronous updates where the update set $\mathcal{T}$ is assumed to be selected at random in each iteration. More precisely, we assume that the k^{th} iteration has the following form:

$$\mathbf{x}_k = \mathbf{Q} \, \mathbf{x}_{k\text{-}1}, \qquad (12)$$

where $\mathbf{x}_k$ denotes the signal at the k^{th} iteration, and $\mathbf{Q}$ is a random matrix due to the fact that the underlying update set is selected at random.

It should be noted that $\mathbf{Q}$ and $\mathbf{Q}(\mathcal{T})$ are different from each other. The matrix $\mathbf{Q}(\mathcal{T})$ in (11) is a *deterministic* matrix. Given an update set $\mathcal{T}$, $\mathbf{Q}(\mathcal{T})$ represents the asynchronous update of (9). On the other hand, $\mathbf{Q}$ in (12) is a random variable *whose outcomes are in the form of* $\mathbf{Q}(\mathcal{T})$. More precisely, we consider the following probabilistic model:

$$\mathbb{P}[\, \mathbf{Q} = \mathbf{Q}(\mathcal{T}) \,] = p_t \, \binom{N}{t}^{-1}, \quad \text{where} \quad t = |\mathcal{T}|, \qquad (13)$$

where p_t denotes the probability of $\mathcal{T}$ having size t, that is,

$$p_t = \mathbb{P}\big[\, |\mathcal{T}| = t \,\big], \qquad (14)$$

and $\sum_{t=1}^{N} p_t = 1$.

According to the model in (13), subsets of equal size are selected with equal probabilities. Therefore, the update scheme does not have any bias toward any node(s). To put differently, all the nodes are treated equally in the network. When $p_N = 1$, the model in (13) reduces to the regular power iteration in (7). When $p_1 = 1$, only one node is selected uniformly at random, which corresponds to the autonomous network model of interest.

The number of nodes to be updated, $T = |\mathcal{T}|$, is a discrete random variable whose distribution will be shown to determine the behavior of the asynchronous updates. We will see later in Section III-B that the following definition is very useful in our quantitative analysis:

$$\delta_T = \frac{\mathbb{E}[\, T\,(N-T)\,]}{\mathbb{E}[\, T\,(N-1)\,]} = \frac{N - \mu_T - \sigma_T^2/\mu_T}{N-1}, \qquad (15)$$

where μ_T and σ_T^2 denote the mean and the variance of the random quantity T, respectively. It can be verified that $0 \leqslant \delta_T \leqslant 1$ with $\delta_T = 0$ if and only if all the nodes are updated in each iteration (synchronous power iteration), and $\delta_T = 1$ if and only if exactly one node is updated in each iteration. As a result of this, δ_T will be referred to as *the amount of asynchronicity* of iterations in the rest of the paper.

We now prove:

Lemma 1: Expectation of the random matrix $\mathbf{Q}$ in (13) is:

$$\mathbb{E}[\mathbf{Q}] = \frac{\mu_T}{N} \, \mathbf{A} + \left(1 - \frac{\mu_T}{N}\right) \mathbf{I}. \qquad (16)$$

Proof: The expectation of $\mathbf{Q}$ can be written as

$$\mathbb{E}[\mathbf{Q}] = \mathbb{E}\big[\, \mathbb{E}[\mathbf{Q}|T] \,\big], \qquad (17)$$

where the outer expectation is with respect to T (size of the sets), and the inner expectation is with respect to the content of the subsets of size T. Using (13) we have that

$$\mathbb{E}[\mathbf{Q} \,|\, T] = \sum_{\mathcal{T}} \mathbb{P}[\, \mathbf{Q} = \mathbf{Q}(\mathcal{T}) \,|\, T \,] \;\; \mathbf{Q}(\mathcal{T}), \qquad (18)$$

$$= \sum_{\mathcal{T}} \; \frac{1}{\binom{N}{T}} \, \Big(\mathbf{I} + \mathbf{D}_{\mathcal{T}} \, (\mathbf{A} - \mathbf{I}) \Big), \qquad (19)$$

where $\sum_{\mathcal{T}}$ denotes a summation over subsets of size T. Then,

$$\mathbb{E}[\mathbf{Q} \,|\, T] = \frac{1}{\binom{N}{T}} \sum_{\mathcal{T}} \mathbf{I} + \frac{1}{\binom{N}{T}} \sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}} \, (\mathbf{A} - \mathbf{I}), \qquad (20)$$

$$= \mathbf{I} + \frac{1}{\binom{N}{T}} \binom{N-1}{T-1} \, \mathbf{I} \, (\mathbf{A} - \mathbf{I}), \qquad (21)$$

$$= T/N \; \mathbf{A} + (1 - T/N) \; \mathbf{I}, \qquad (22)$$

where (21) follows from (5). Due to (17), we have

$$\mathbb{E}[\mathbf{Q}] = \mathbb{E}[\; T/N \; \mathbf{A} + (1 - T/N) \; \mathbf{I} \,], \qquad (23)$$

$$= \mu_T/N \; \mathbf{A} + (1 - \mu_T/N) \; \mathbf{I}, \qquad (24)$$

which gives the result in (16). ■

Notice that $\mathbb{E}[\mathbf{Q}]$ is a convex combination of the operator $\mathbf{A}$ and the identity matrix. The quantity μ_T/N is the average fraction of the nodes that are updated simultaneously per iteration, and it appears as the weight of the operator $\mathbf{A}$ in $\mathbb{E}[\mathbf{Q}]$. The case of $\mu_T = N$ results in $\mathbb{E}[\mathbf{Q}] = \mathbf{A}$, which corresponds to the case of synchronous power iteration.

III. Cascade of Asynchronous Updates

For the most practical scenarios we are interested in studying a sequence of random updates and the effect of the underlying matrix $\mathbf{A}$ on the convergence of the iterations. In the case of the synchronous (unnormalized) power iteration in (7), it is well known that an arbitrary nonzero initial signal $\mathbf{x}_0$ converges to a nonzero $\mathbf{x}$ if and only if $\mathbf{x}$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{x}$ (i.e., 1 is an eigenvalue of $\mathbf{A}$), and the remaining eigenvalues of $\mathbf{A}$ satisfy $|\lambda| < 1$. If there is another eigenvalue satisfying $|\lambda| = 1$, then the signal

$\mathbf{x}_k$ may fall into limit cycles, and if $|\lambda| > 1$, the signal grows in an unbounded manner through the iterations.

The random asynchronous update in (12) has very different convergence properties as we shall see. Iteratively using (12), the signal at the k^{th} iteration can be written as:

$$\mathbf{x}_k = \mathbf{Q}_k\ \mathbf{Q}_{k\text{-}1} \cdots\ \mathbf{Q}_2\ \mathbf{Q}_1\ \mathbf{x}_0, \tag{25}$$

where $\mathbf{x}_0$ denotes the initial signal, and $\mathbf{Q}_i$'s are *independent* and *identically* distributed copies of the random matrix $\mathbf{Q}$ in (13). It should be noted that $\mathbf{x}_k$ is a random vector due to the fact that $\mathbf{Q}_i$'s are random variables.

A. Expected Amount of Projection onto the Eigenvectors

In order to characterize the behavior of $\mathbf{x}_k$, we first define the following quantity:

$$\widehat{x}_{k,\,j} = \mathbf{v}_j^{\mathrm{H}}\ \mathbf{x}_k, \tag{26}$$

which is the amount of projection on the j^{th} eigenvector (or, the j^{th} graph Fourier coefficient) of $\mathbf{x}_k$ at the k^{th} iteration. Due to the randomness of the updates, $\widehat{x}_{k,j}$ are random quantities as well. The following theorem gives the expected value of these coefficients:

Theorem 1: Let $\mathbf{v}_j$ and λ_j be an eigenpair of $\mathbf{A}$. Then,

$$\mathbb{E}[\widehat{x}_{k,\,j}] = \left(1 + \frac{\mu_T}{N}\,(\lambda_j - 1)\right)^k\ \widehat{x}_{0,\,j}. \tag{27}$$

Proof: From (25) and (26) we have

$$\mathbb{E}[\widehat{x}_{k,\,j}] = \mathbf{v}_j^{\mathrm{H}}\ \mathbb{E}[\mathbf{Q}_k\ \mathbf{x}_{k\text{-}1}] = \mathbf{v}_j^{\mathrm{H}}\ \mathbb{E}[\mathbf{Q}]\ \mathbb{E}[\mathbf{x}_{k\text{-}1}], \tag{28}$$

where the last equality follows from the fact that $\mathbf{Q}_k$'s are independent and identically distributed random variables. Using the spectral decomposition of $\mathbf{A}$ in (2), $\mathbb{E}[\mathbf{Q}]$ in (16) can be written as:

$$\mathbb{E}[\mathbf{Q}] = \mathbf{V}\left(\frac{\mu_T}{N}\,\mathbf{\Lambda} + \left(1 - \frac{\mu_T}{N}\right)\mathbf{I}\right)\mathbf{V}^{\mathrm{H}}. \tag{29}$$

Therefore, (28) results in the following:

$$\mathbb{E}[\widehat{x}_{k,\,j}] = \mathbf{v}_j^{\mathrm{H}}\ \mathbf{V}\left(\frac{\mu_T}{N}\mathbf{\Lambda} + \left(1 - \frac{\mu_T}{N}\right)\mathbf{I}\right)\mathbf{V}^{\mathrm{H}}\ \mathbb{E}[\mathbf{x}_{k\text{-}1}], \tag{30}$$

$$= (1 + \mu_T/N\,(\lambda_j - 1))\ \ \mathbb{E}[\widehat{x}_{k\text{-}1,\,j}]. \tag{31}$$

Iterative use of (31) gives the result in (27). ■

Theorem 1 shows that expected value of the graph Fourier coefficients depends not only on the corresponding eigenvalues but on the average number of nodes updated in each iteration as well. An immediate corollary is as follows:

Corollary 1: Let $\mathbf{v}_j$ and λ_j be an eigenpair of $\mathbf{A}$. If

$$\left|1 + \frac{\mu_T}{N}\,(\lambda_j - 1)\right| < 1. \tag{32}$$

Then,

$$\lim_{k\to\infty} \mathbb{E}[\widehat{x}_{k,\,j}] = 0. \tag{33}$$

Proof: This follows from Theorem 1 and (26). ■

In the case of synchronous updates $\mu_T = N$, hence the relation in (27) reduces to $\mathbb{E}[\widehat{x}_{k,\,j}] = \widehat{x}_{k,\,j} = \lambda_j^k\ \widehat{x}_{0,\,j}$ as expected. Furthermore, (32) reduces to $|\lambda| < 1$, which is the well-known condition for the convergence of the power iteration of (7).

When the updates are asynchronous we have $\mu_T < N$, and the region of convergence (32) in the complex eigenvalue plane is larger. In particular, one can readily verify the following:

$$\begin{matrix} |\lambda_j| \leqslant 1 \\ \lambda_j \neq 1 \end{matrix} \implies \left|1 + \frac{\mu_T}{N}\,(\lambda_j - 1)\right| < 1, \tag{34}$$

which implies that $\mathbb{E}[\widehat{x}_{k,\,j}]$ converges to zero even if λ_j is *on* the unit circle, except when $\lambda_j = 1$. This is very much unlike to the synchronous case where the coefficients corresponding to unit magnitude eigenvalues do not die out through the iterations.

The unit eigenvalue $\lambda_j = 1$ deserves a specific attention since it has $\mathbb{E}[\widehat{x}_{k,\,j}] = \widehat{x}_{0,\,j}$ irrespective of the value of μ_T. If the initial signal $\mathbf{x}_0$ has a nonzero projection onto the eigenspace of the unit eigenvalue, then $\mathbb{E}[\mathbf{x}_k]$ always has a nonzero projection onto the eigenspace of the unit eigenvalue. Furthermore, the following lemma shows that a signal is invariant to the random asynchronous updates if and only if the signal lies in the eigenspace of the unit eigenvalue.

Lemma 2: A signal $\mathbf{x}$ is invariant under all asynchronous updates if and only if it is invariant under the synchronous update. That is,

$$\mathbf{A}\mathbf{x} = \mathbf{x} \iff \mathbf{Q}(\mathcal{T})\,\mathbf{x} = \mathbf{x}\ \ \forall \mathcal{T}. \tag{35}$$

Proof: ($\Leftarrow$) Assume that $\mathbf{Q}(\mathcal{T})\,\mathbf{x} - \mathbf{x} = \mathbf{0}$ for all $\mathcal{T}$. Since it is true for *any* subset $\mathcal{T}$, it should hold true for all subsets of a fixed size t as well. Then,

$$\mathbf{0} = \sum_{\mathcal{T}} \mathbf{Q}(\mathcal{T})\,\mathbf{x} - \mathbf{x} = \sum_{\mathcal{T}} \mathbf{D}_{\mathcal{T}}\,(\mathbf{A} - \mathbf{I})\,\mathbf{x}, \tag{36}$$

$$= \binom{N-1}{t-1}\,(\mathbf{A} - \mathbf{I})\,\mathbf{x}, \tag{37}$$

which proves that $\mathbf{A}\mathbf{x} = \mathbf{x}$.

The converse ($\Rightarrow$) simply follows from (11). ■

Lemma 2 shows that if the random signal $\mathbf{x}_k$ ever reaches a steady-state point $\mathbf{x}$ through iterations, then $\mathbf{x}$ should be in the eigenspace of the unit eigenvalue. The result of Corollary 1 supports this claim as well. This is not a surprising result as the eigenspace of eigenvalue 1 consists of non-zero fixed points of the operator. However, neither Lemma 2 nor Corollary 1 says anything about the convergence of the random asynchronous iteration as k increases. In the following section, we will prove that $\mathbf{x}_k$ indeed converges to an eigenvector of the unit eigenvalue (a fixed point) as k goes to infinity.

B. Convergence in Mean-Squared Sense

In the following we will assume that $\mathbf{A}$ has a unit eigenvalue with multiplicity $M \geqslant 1$. This assumption ensures that the asynchronous update equation has a fixed point (Lemma 2). Without loss of generality we will order the eigenvalues of $\mathbf{A}$ such that $\lambda_j \neq 1$ for $1 \leqslant j \leqslant N - M$. Notice that non-unit eigenvalues are allowed to be complex in general, and complex eigenvalues on or outside the unit circle are not ruled out. Then, the eigenvalue decomposition of $\mathbf{A}$ can be written as:

$$\mathbf{A} = [\mathbf{U}\ \ \mathbf{V}_1]\mathrm{diag}\big([\lambda_1\ \cdots\ \lambda_{N\text{-}M}\ 1\ \cdots\ 1]\big)[\mathbf{U}\ \ \mathbf{V}_1]^{\mathrm{H}}, \tag{38}$$

where $\mathbf{V}_1 \in \mathbb{C}^{N\times M}$ is an orthonormal basis for the eigenspace of the unit eigenvalue, and $\mathbf{U} \in \mathbb{C}^{N\times(N-M)}$ corresponds to the eigenvectors of the non-unit eigenvalues. Since $\mathbf{A}$ is assumed to be a normal matrix, we have $\mathbf{U}^{\mathrm{H}}\,\mathbf{V}_1 = \mathbf{0}$, and $\mathbf{U}^{\mathrm{H}}\,\mathbf{U} = \mathbf{I}$.

We now define the following quantities:

$$\rho^+ = \lambda_{\max}\left(\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\right), \tag{39}$$

$$\rho^- = \lambda_{\min}\left(\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\,\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\right), \tag{40}$$

which will play a crucial role in the analysis of convergence. Notice that ρ^+ and ρ^- do not depend on the particular selection of the basis matrix $\mathbf{U}$. Just the column space of $\mathbf{U}$ determines their values. Notice that $\mathbf{U}\mathbf{U}^{\mathrm{H}} \preccurlyeq \mathbf{I}$, which implies that $\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}}) \preccurlyeq \mathbf{I}$, hence $\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U} \preccurlyeq \mathbf{I}$. Based on these we have,

$$0 < \rho^- \leqslant \rho^+ \leqslant 1, \tag{41}$$

where $0 < \rho^-$ is readily shown to be valid.

For an arbitrary $\mathbf{x}_k$, let $\mathbf{r}_k$ denote the residual from the projection of $\mathbf{x}_k$ onto the column space of $\mathbf{V}_1$. That is,

$$\mathbf{r}_k = \mathbf{x}_k - \mathbf{V}_1\,\mathbf{V}_1^{\mathrm{H}}\,\mathbf{x}_k = \mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_k. \tag{42}$$

Then, the convergence of $\mathbf{x}_k$ to an eigenvector of the unit eigenvalue is equivalent to the convergence of $\mathbf{r}_k$ to zero. The following theorem, whose proof is presented in Appendix B, provides bounds for $\mathbf{r}_k$ as follows:

Theorem 2: The expected squared ℓ_2-norm of the residual at the k^{th} iteration is bounded as follows:

$$\psi^k\,\|\mathbf{r}_0\|_2^2 \;\leqslant\; \mathbb{E}\big[\,\|\mathbf{r}_k\|_2^2\,\big] \;\leqslant\; \Psi^k\,\|\mathbf{r}_0\|_2^2, \tag{43}$$

where

$$\Psi = \max_{1\leqslant j\leqslant N\text{-}M}\; 1 + \frac{\mu_T}{N}\left(|\lambda_j|^2 - 1 + \delta_T\,(\rho^+ - 1)\,|\lambda_j - 1|^2\right). \tag{44}$$

$$\psi = \min_{1\leqslant j\leqslant N\text{-}M}\; 1 + \frac{\mu_T}{N}\left(|\lambda_j|^2 - 1 + \delta_T\,(\rho^- - 1)\,|\lambda_j - 1|^2\right). \tag{45}$$

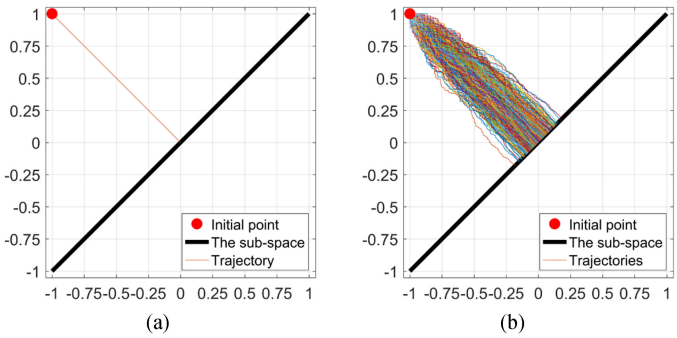
♢

The importance of Theorem 2 is twofold: First, it reveals the effect of the eigenvalues (λ_j), the eigenspace geometry (ρ^+, ρ^-), and the amount of asynchronicity of the updates (δ_T) on the rate of convergence. In the synchronous case $\delta_T = 0$ and $\mu_T = N$, hence we have $\Psi = \max_{1\leqslant j\leqslant N\text{-}M}|\lambda_j|^2$. This result is consistent with the well-known fact that the rate of convergence of the power iteration is determined by the second largest eigenvalue. However, in the asynchronous case ($\delta_T > 0$), not just the eigenvalues but the eigenspace geometry of $\mathbf{A}$ has an effect. As a result, similar matrices may have different convergence rates due to their different eigenspaces. This point will be elaborated in Section IV. Furthermore, in order to guarantee that $\mathbb{E}[\|\mathbf{r}_k\|_2^2] \leqslant \varepsilon\,\|\mathbf{r}_0\|_2^2$ for a given error threshold ε, inequalities in (43) show that it is necessary to have at least $\lfloor \log(\varepsilon)/\log(\psi)\rfloor$ iterations, and sufficient to have $\lceil \log(\varepsilon)/\log(\Psi)\rceil$ iterations.

Secondly, Theorem 2 reveals a region for the eigenvalues such that the residual error through asynchronous updates is guaranteed to convergence to zero in the mean-squared sense. The following corollary presents this result formally.

Corollary 2: Assume that all non-unit eigenvalues of $\mathbf{A}$ satisfy the following condition:

$$\left|\lambda - \frac{\alpha}{\alpha+1}\right| < \frac{1}{\alpha+1}, \tag{46}$$



(a) (b)

Fig. 1. Some realizations of the trajectories of the signal through updates for (a) the non-random synchronous case, (a) the random asynchronous case.

where

$$\alpha = \delta_T\,(\rho^+ - 1). \tag{47}$$

Then,

$$\lim_{k\to\infty}\;\; \mathbb{E}\big[\,\|\mathbf{r}_k\|_2^2\,\big] = 0. \tag{48}$$

Proof: From (44) it is clear that $\Psi < 1$ if and only if

$$|\lambda|^2 - 1 + \alpha\,|\lambda - 1|^2 < 0, \tag{49}$$

for all non-unit eigenvalues λ. The inequality in (49) can be equivalently written as in (46). Since it implies that $\Psi < 1$, Theorem 2 guarantees the convergence of $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ to zero as the number of updates, k, goes to infinity. ■

An important remark is as follows: Corollary 2 provides a condition under which $\mathbf{r}_k$ is guaranteed to converge to a point (zero) as k goes to infinity. On the other hand, $\mathbf{x}_k$ itself only converges to a random variable defined over the eigenspace of the unit eigenvalue. This is illustrated in Figure 1 where the eigenspace of the unit eigenvalue is spanned by the vector $[1\;\;1]^{\mathrm{H}}$, and $\mathbf{x}_0 = [-1\;\;1]^{\mathrm{H}}$. In the synchronous case the signal converges to a point through a deterministic trajectory as shown in Figure 1(a). For the random asynchronous case, Figure 1(b) illustrates the trajectories of the signals for different realizations. Convergence of $\mathbf{r}_k$ to zero implies that the limit of $\mathbf{x}_k$ always lie in the eigenspace of the unit eigenvalue (with a random orientation). Since any point in the eigenspace is an eigenvector, we can safely say that $\mathbf{x}_k$ converges to an eigenvector of the unit eigenvalue.

Notice that the convergence region for the eigenvalues defined in (46) is parametrized by α, and it is a disk on the complex plane with radius $1/(\alpha+1)$ centered at $\alpha/(\alpha+1)$. This region is visualized in Figure 2. Notice that $0 \leqslant \delta_T \leqslant 1$ and $0 < \rho^+ \leqslant 1$ always hold true. As a result α satisfies $-1 < \alpha \leqslant 0$. The key observation is that the region in (46) grows as α approaches -1, and it is the smallest (and corresponds to the unit disk) when $\alpha = 0$. The quantity β and the large circle in Figure 2 will be explained after Corollary 4.

Corollary 2 reveals the combined effect of the eigenspace geometry of $\mathbf{A}$ (quantified with ρ^+) and the amount of asynchronicity (quantified with δ_T) on the convergence of the iterations. In the case of $\delta_T = 0$ the region reduces to the unit disk, which is the well-known condition on the eigenvalues for the synchronous updates to converge. This is an expected result since the case of $\delta_T = 0$ corresponds to the synchronous update itself. More importantly, the synchronous updates imply $\alpha = 0$

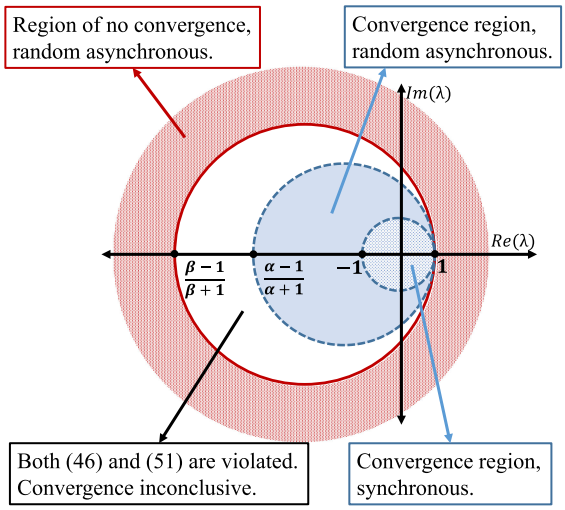


Fig. 2. Regions (given in (46) and (51)) for the eigenvalues such that random asynchronous updates are guaranteed to converge and diverge, respectively.

independent of the eigenspace geometry of $\mathbf{A}$. Therefore, the convergence is determined entirely by the eigenvalues of $\mathbf{A}$ in the synchronous case.

On the other hand, the case of asynchronous updates results in a *larger* convergence region for the eigenvalues. First of all, it should be noted that asynchronous updates increase the convergence region if the eigenspace geometry of $\mathbf{A}$ permits. If $\rho^+ = 1$ then $\alpha = 0$, and the region of convergence is not improved by asynchronous iterations. However, if $\rho^+ < 1$ (which is the case in most practical applications), then it is possible to enlarge the region of convergence using asynchronous iterations. As δ_T gets larger (less number of nodes are updated concurrently), α gets smaller, hence the convergence region gets larger. Even if one index is left unchanged in *some* iterations, we have $\delta_T > 0$, and the residual $\mathbf{r}_k$ can converge to zero, *even when non-unit eigenvalues outside the unit circle might exist.* This is a remarkable property of the asynchronous updates since the residual (hence the signal itself) would blow up in the case of synchronous updates. Notice that in the extreme case of $\delta_T = 1$, the region of convergence is the largest possible. That is to say, *updating exactly one node in each iteration maximizes the region of convergence of the eigenvalues.* On the other extreme, the synchronous update is the most restrictive case, which is formally stated in the following corollary:

Corollary 3: If the synchronous updates on $\mathbf{A}$ converge, then

$$\lim_{k\to\infty} \quad \mathbb{E}\big[\, \|\mathbf{r}_k\|_2^2 \,\big] = 0, \tag{50}$$

for random updates on $\mathbf{A}$ with any amount of asynchronicity.

Proof: If the synchronous updates converge, then all non-unit eigenvalues of $\mathbf{A}$ satisfy $|\lambda| < 1$. Hence, they also satisfy (46) for any value of α. Therefore, Corollary 2 ensures the convergence of the updates irrespective of the value of δ_T. ■

It should be clear that converse of Corollary 3 is not true. Thus consider a scenario in which a signal over a network of nodes with autonomous (asynchronous) behavior stays in the steady-state. If the nodes start to operate synchronously, then it is possible for the signal to blow up. This happens if some of the eigenvalues fall outside of the reduced convergence region due to the reduction in the amount of asynchronicity. In fact, the study in [21] claims that large-scale synchronization of neurons is an underlying mechanism of epileptic seizures. Similarly, the study in [50] presents the relation between increased neural synchrony and epilepsy as well as Parkinson's disease. It should be noted that neural networks follow nonlinear models whereas the model we consider here is linear. Thus, results presented here do not apply to brain networks. Nevertheless, these neurobiological observations are consistent with the implications of Corollary 2 and Corollary 3 from a conceptual point of view.

Apart from the convergence of the iterations, Theorem 2 is also useful to characterize the case of non-converging iterations. In this regard, the following corollary presents a region for the eigenvalues such that asynchronous updates are guaranteed not to converge.

Corollary 4: Assume that all non-unit eigenvalues of $\mathbf{A}$ satisfy the following:

$$\left|\lambda - \frac{\beta}{\beta+1}\right| \geqslant \frac{1}{\beta+1}, \tag{51}$$

where

$$\beta = \delta_T\ (\rho^- - 1). \tag{52}$$

Then,

$$\mathbb{E}\big[\, \|\mathbf{r}_k\|_2^2 \,\big] \geqslant \|\mathbf{r}_0\|_2^2. \tag{53}$$

Furthermore, if (51) is satisfied with strict inequality, then $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ grows unboundedly as k goes to infinity.

Proof: From (45) it is clear that $\psi \geqslant 1$ if and only if

$$|\lambda|^2 - 1 + \beta\,|\lambda - 1|^2 \geqslant 0, \tag{54}$$

for all non-unit eigenvalues λ. The inequality in (54) can be equivalently written as in (51). Since (51) implies that $\psi \geqslant 1$, Theorem 2 indicates that $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ is lower bounded by $\|\mathbf{r}_0\|_2^2$. If (51) is satisfied strictly, then $\psi > 1$. As a result, $\mathbb{E}[\|\mathbf{r}_k\|_2^2]$ grows unboundedly as k goes to infinity. ■

From the definitions in (47) and (52) note that $\alpha \geqslant \beta$ is always true due to the fact that $\rho^+ \geqslant \rho^-$. Therefore, the conditions in (46) and (51) describe disjoint regions on the complex plane. See Figure 2. Corollary 4 also shows that the condition $|\lambda - \beta/(\beta+1)| < 1/(\beta+1)$ is *necessary* for the iterations to converge, whereas the condition in (46) is *sufficient* for the convergence (both in the mean square sense). If there exists an eigenvalue that violates both (46) and (51), then convergence is inconclusive. This region is also indicated in Figure 2.

At this point it is important to compare the implications of Corollary 2 with the classical result presented in [26], [27]. Under the mild assumption that all the indices are selected sufficiently often (see [27] for precise definition), the study [27] showed that the linear asynchronous model in (9) converges for any index sequence if and only if the spectral radius of $|\mathbf{A}|$ is strictly less than unity, where $|\mathbf{A}|$ denotes a matrix with element-wise absolute values of $\mathbf{A}$. On the other hand, our Corollary 2 allows eigenvalues with magnitudes grater than unity. Although these two results appear to be contradictory (when $\mathbf{A}$ consists of non-negative elements), the key difference is the notion of convergence. As an example, consider the matrix $\mathbf{A}_2$ defined in (55). Its spectral radius is exactly 1, and [27] proved that *there exists* a sequence of indices under which iterations on $\mathbf{A}_2$ do not converge. For example, assuming N is odd, consider the

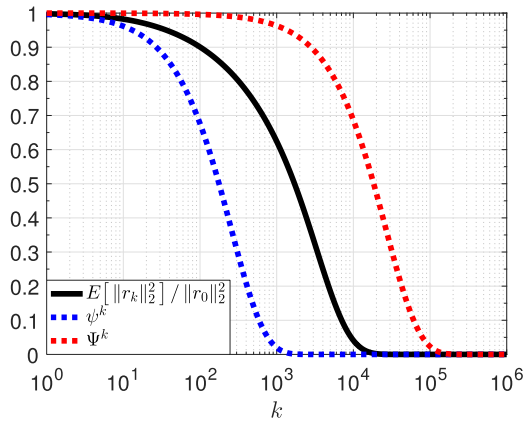


Fig. 3. Simulated convergence of random asynchronous updates on the cyclic shift matrix of size $N = 32$ together with the bounds provided by Theorem 2.

index sequence generated as $i = (2k-1)(\mathrm{mod}\ N)+1$. However, Corollary 2 proves the convergence in a statistical *mean-square averaged* sense. (See Figure 3.) In short, when compared with [27], Corollary 2 requires a weaker condition on $\mathbf{A}$ and guarantees a convergence in a weaker (and probabilistic) sense.

IV. The Importance of the Eigenspace Geometry

Corollary 2 presented in the previous section showed that the signal converges to an eigenvector of the unit eigenvalue via random asynchronous updates even in the case of $\mathbf{A}$ having other eigenvalues with magnitudes larger than one. Thus, asynchronous updates can enlarge the convergence region of the eigenvalues if the eigenspace geometry of $\mathbf{A}$ permits. In this section we will consider a simple example to demonstrate the effect of the eigenspace geometry on the convergence of the asynchronous updates. For this purpose we will consider the following matrices of size N:

$$\mathbf{A}_1 = \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_N \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} & & 1 \\ 1 & & \\ & \ddots & \\ & 1 & \end{bmatrix}, \tag{55}$$

where the diagonal entries of $\mathbf{A}_1$ are $A_n = e^{j2\pi n/N}$, and $\mathbf{A}_2$ is the cyclic permutation matrix. Note that $\mathbf{A}_1$ and $\mathbf{A}_2$ are related by a similarity, so they have identical eigenvalues. In particular they have eigenvalue 1 with multiplicity $M = 1$. However, they have different eigenspace geometries that affect the behavior of asynchronous iterations as demonstrated next.

Notice that an update with $\mathbf{A}_1$ corresponds to element-wise multiplication with complex exponentials that does not change the magnitude of the entries. As a result, iterations with $\mathbf{A}_1$ do not converge or blow up, whether the updates are synchronous or asynchronous. Indeed in this case $\rho^+ = \rho^- = 1$ and $\beta = 0$, which lead to the same conclusion by Corollary 4.

Next, $\mathbf{A}_2$ is circulant matrix, and the normalized DFT matrix diagonalizes it. In this case $\rho^+ = \rho^- = 1 - 1/N$. In the synchronous case $\delta_T = 0$, hence $\beta = 0$, and Corollary 4 shows that the residual error is bounded below. In fact, the power iteration on $\mathbf{A}_2$ neither converges nor blows up since it corresponds to the cyclic shift of the vector, and the initial signal repeats itself after every N iterations. In the asynchronous case, $\mathbf{A}_2$ has $\alpha = -\delta_T/N$, hence any nonzero amount of asynchronicity implies $\alpha < 0$, in which case the convergence region in (46) contains the unit closed disk. Thus, the initial signal converges to an eigenvector of the unit eigenvalue. As an example consider the case of updating exactly one index in each iteration. This is equivalent to randomly selecting an index and assigning its value to the next one. As random updates are applied repeatedly the signal will have more and more duplicate elements until all the elements are the same. The final vector is $\eta\,\mathbb{1}$ where η is a random variable. Thus, the initial signal converges to a constant vector, which is the eigenvector of $\mathbf{A}_2$ with the unit eigenvalue. (See Figure 1(b).) Theorem 2 verifies this convergence in the mean-squared sense with the following bounds on the rate:

$$\Psi = \max_{1 \leqslant n \leqslant N\text{-}1} 1 - \frac{1}{N}\frac{1}{N}\left|e^{j2\pi n/N} - 1\right|^2 \approx 1 - 4\pi^2/N^4, \tag{56}$$

and

$$\psi = \min_{1 \leqslant n \leqslant N\text{-}1} 1 - \frac{1}{N}\frac{1}{N}\left|e^{j2\pi n/N} - 1\right|^2 \approx 1 - 4/N^2. \tag{57}$$

In order to compare the bounds in (56) and (57) with the actual rate of convergence, asynchronous iterations with $\delta_T = 1$ are simulated on $\mathbf{A}_2$ of size $N = 32$. Figure 3 visualizes the expected squared ℓ_2-norm of the residual with respect to the iteration index as well as the bounds given by Theorem 2. The result is obtained by averaging over 10^7 independent runs.

Simulations show that the bounds suggested by Theorem 2 are off by an order of magnitude in this specific example. Since the random updates converge faster than the upper bound given by Theorem 2, i.e., (56), tighter bounds on the rate of convergence may be possible to obtain.

V. Interpretation of The Asynchronous Updates in the Context of Graph Signal Processing

In Sections II and III we defined and studied the random asynchronous updates from a linear algebraic point of view where the results (Theorems 1, 2, and their corollaries) are general enough to apply them to an arbitrary normal matrix. When $\mathbf{A}$ is assumed to be a graph operator, the asynchronous iterations offer an insightful way to interpret a "typical" graph signal. Consider a network of fully autonomous nodes (agents) and assume that a node updates its value via retrieving information from its neighbors. This update scheme can be modeled with (9) where the update set $\mathcal{T}$ has only one element ($\delta_T = 1$). In this model, the local graph operator $\mathbf{A}$ models the way nodes update their values. If a node computes the sum of its neighbors, then $\mathbf{A}$ is the adjacency matrix; if a node computes the sum of differences with its neighbors, then $\mathbf{A}$ is the graph Laplacian; if a node computes a weighted average of its neighbors, then $\mathbf{A}$ is a weighted adjacency matrix. Therefore, the matrix $\mathbf{A}$ describes what the nodes compute, the update scheme in (9) describes the time dynamics of the nodes, and the values held by the nodes are considered as a signal on the graph (with respect to the operator $\mathbf{A}$). In the following, we will interpret random asynchronous updates in this context and argue that a "typical signal" on a graph is necessarily a "smooth signal" with respect to the graph operator. For this purpose first notice that the GFT of $\mathbf{A}\mathbf{x} - \mathbf{x}$

is given by $(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}$. Then, we define a notion of smoothness accordingly as follows:

Definition 1 (Smoothness Set): A graph signal $\mathbf{x}$ belongs to the set $\mathcal{S}_\epsilon$ if its graph Fourier transform $\widehat{\mathbf{x}}=\mathbf{V}^{\mathrm{H}}\mathbf{x}$ satisfies

$$|\widehat{x}_i|\,|\lambda_i-1|\leqslant\epsilon\qquad\forall\, i, \tag{58}$$

for the given graph operator $\mathbf{A}$. ♢

A signal $\mathbf{x}$ belongs to $\mathcal{S}_\epsilon$ if the differences between the graph Fourier coefficients of $\mathbf{x}$ and $\mathbf{A}\mathbf{x}$ are not larger than ϵ in absolute sense. That is, small values of ϵ implies that $\mathbf{x}$ and $\mathbf{A}\mathbf{x}$ are similar to each other. Hence, we can interpret ϵ as a scale of the smoothness of the signal $\mathbf{x}$ with respect to the operator $\mathbf{A}$. Here, the smoothness is quantified according to the total variation (TV) of the eigenvectors, $\mathrm{TV}(\mathbf{v}_i)=|\lambda_i-1|$, as introduced in [6]. For a given value of ϵ, the set $\mathcal{S}_\epsilon$ describes the signals with $|\widehat{x}_i|\leqslant\epsilon/|\lambda_i-1|$. So, for a smooth signal, the Fourier coefficient $\widehat{x}_i$ should be smaller for those λ_i whose $\mathbf{v}_i$ has large total variation. The condition in (58) is equivalent to bounding a weighted max-norm of the GFT of $\mathbf{x}$, that is,

$$\mathbf{x}\in\mathcal{S}_\epsilon\quad\Longleftrightarrow\quad\|\,(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}\,\|_\infty\leqslant\epsilon, \tag{59}$$

where the weight matrix is selected as $|\mathbf{\Lambda}-\mathbf{I}|$. The set $\mathcal{S}_\epsilon$ depends on the underlying graph operator. *A signal that is smooth on one graph may not be smooth on another graph.*

In the following we will consider the effect of a single asynchronous update on the smoothness of the signal. For this purpose let $\mathbf{x}$ be the initial signal and $\mathbf{y}$ be the signal after an update. According to (11) they are related as $\mathbf{y}=\mathbf{Q}(\mathcal{T})\,\mathbf{x}$, which can be equivalently written in the graph Fourier domain as follows:

$$\widehat{\mathbf{y}}=\mathbf{V}^{\mathrm{H}}\,\mathbf{Q}(\mathcal{T})\,\mathbf{V}\,\widehat{\mathbf{x}}=\widehat{\mathbf{x}}+\sum_{i\in\mathcal{T}}\mathbf{V}^{\mathrm{H}}\mathbf{e}_i\,\mathbf{e}_i^{\mathrm{H}}\,\mathbf{V}\,(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}. \tag{60}$$

The following theorem reveals a relation between smooth graph signals and a single asynchronous update of (9).

Theorem 3: Assume that the signal $\mathbf{x}$ belongs to $\mathcal{S}_\epsilon$ of a graph with operator $\mathbf{A}$. Then, the signal $\mathbf{y}$ computed as in (9) satisfies the following:

$$\|\widehat{\mathbf{y}}-\widehat{\mathbf{x}}\|_\infty\;\leqslant\;\epsilon\,|\mathcal{T}|\,\|\mathbf{V}\|_{\max}\,\|\mathbf{V}\|_\infty. \tag{61}$$

Proof: Assume that $\mathbf{x}\in\mathcal{S}_\epsilon$. Then, we can write the following set of inequalities for a fixed index j:

$$|\widehat{y}_j-\widehat{x}_j|=\left|\mathbf{e}_j^{\mathrm{H}}\sum_{i\in\mathcal{T}}\mathbf{V}^{\mathrm{H}}\mathbf{e}_i\,\mathbf{e}_i^{\mathrm{H}}\,\mathbf{V}\,(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}\right|, \tag{62}$$

$$\leqslant\sum_{i\in\mathcal{T}}\left|\mathbf{e}_j^{\mathrm{H}}\mathbf{V}^{\mathrm{H}}\mathbf{e}_i\right|\,\left|\mathbf{e}_i^{\mathrm{H}}\,\mathbf{V}\,(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}\right|, \tag{63}$$

$$\leqslant\|\mathbf{V}\|_{\max}\,\sum_{i\in\mathcal{T}}\left|\mathbf{e}_i^{\mathrm{H}}\,\mathbf{V}\,(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}\right|, \tag{64}$$

$$\leqslant\|\mathbf{V}\|_{\max}\,\sum_{i\in\mathcal{T}}\left\|\mathbf{V}^{\mathrm{H}}\,\mathbf{e}_i\right\|_1\,\|(\mathbf{\Lambda}-\mathbf{I})\,\widehat{\mathbf{x}}\|_\infty, \tag{65}$$

$$\leqslant\|\mathbf{V}\|_{\max}\,|\mathcal{T}|\,\|\mathbf{V}\|_\infty\,\epsilon, \tag{66}$$

where (62) follows from (60), (63) follows from the triangle inequality, (64) follows from the definition of $\|\mathbf{V}\|_{\max}$, (65) follows from the Hölder inequality, and (66) follows from the fact that $\|\mathbf{V}\|_\infty$ is the largest ℓ_1-norm of the rows of $\mathbf{V}$. Then, we have the following:

$$\|\widehat{\mathbf{y}}-\widehat{\mathbf{x}}\|_\infty=\max_j|\widehat{y}_j-\widehat{x}_j|\leqslant\epsilon\,|\mathcal{T}|\,\|\mathbf{V}\|_{\max}\,\|\mathbf{V}\|_\infty, \tag{67}$$

where the inequality follows from the fact that the bound in (66) is valid for any index j. ■

If the underlying graph is circulant, then $\mathbf{V}$ is the normalized DFT matrix, and Theorem 3 reduces to the following simple form as a corollary: $\|\widehat{\mathbf{y}}-\widehat{\mathbf{x}}\|_\infty\leqslant\epsilon\,|\mathcal{T}|$ [51].

The bound given by Theorem 3 is not tight in general. Nevertheless, it provides a useful interpretation: if the initial signal is smooth on the graph (belongs to $\mathcal{S}_\epsilon$ with ϵ being small), then the amount of change in each GFT coefficient is also small, hence a smooth signal remains to be (relatively) smooth on the graph after a single asynchronous update. If the initial signal is not smooth (ϵ is large), then the right-hand-side of (61) is a large quantity, thus we cannot reach a conclusion regarding the effect of an asynchronous update.

We will say that *a signal* $\mathbf{x}$ *is typical to the graph (with respect to the operator* $\mathbf{A}$*) if it satisfies* $\mathbf{A}\mathbf{x}=\mathbf{x}$. This definition is motivated by the following three observations. First of all Lemma 2 shows that a typical signal is invariant (stationary) under any asynchronous update. Secondly, a signal is typical if and only if it is the smoothest signal (w.r.t. Definition 1):

$$\mathbf{x}\in\mathcal{S}_0\quad\Longleftrightarrow\quad\mathbf{A}\,\mathbf{x}=\mathbf{x}. \tag{68}$$

This is consistent with the studies in [4], [7], [9] that consider typical signals to be smooth on the graph. Thirdly, and most importantly, Corollary 2 proves that when the nodes communicate autonomously for a sufficiently long time, the signal becomes typical to the graph irrespective of the starting point. Once the signal gets typical it stays the same over the network. Therefore, a steady signal over an autonomous network is necessary typical with respect to the operator $\mathbf{A}$.

The equivalence given in (68) also shows the smoothing effect of the asynchronous updates. An arbitrary initial signal is not expected to be smooth over a graph, but Corollary 2 proves that the signal over an autonomous network eventually becomes typical, hence the smoothest. However, it should be noted that the smoothing (convergence) does not happen monotonically unlike the power iteration of (7). Some updates might be adversarial that increase the variation of the signal over the graph. See Figure 1 of [51] for an illustrative example. Nevertheless, some other updates cancel them out in the long run as proven by Theorem 2.

VI. Asynchronous Polynomial Filters on Graphs

Lemma 2 of Section III showed that there exists a signal invariant under an asynchronous update if and only if $\mathbf{A}$ has an eigenvalue 1. In this case, according to Corollary 2, an arbitrary nonzero initial signal converges to an eigenvector with eigenvalue 1. These results show that asynchronous updates on $\mathbf{A}$ can only converge to the eigenspace of $\mathbf{A}$ with the unit eigenvalue. In this section we will challenge this limitation: what if we want asynchronous updates to converge to an eigenspace of $\mathbf{A}$ with eigenvalue other than 1? In order to approach this problem we will start with an L^{th} order polynomial of the given operator,

that is,

$$h(\mathbf{A}) = \sum_{n=0}^{L} h_n \, \mathbf{A}^n, \tag{69}$$

for some set of coefficients h_n's, and consider asynchronous updates on $h(\mathbf{A})$ that are defined as follows:

$$(\, \mathbf{x}_{k+1} \,)_i = \begin{cases} \big(\, h(\mathbf{A}) \, \mathbf{x}_k \,\big)_i, & i \in \mathcal{T}, \\ (\, \mathbf{x}_k \,)_i, & i \notin \mathcal{T}. \end{cases} \tag{70}$$

Polynomials of a graph operator are useful to consider because of the following two reasons. Firstly, they are localized. Computation of $(h(\mathbf{A})\,\mathbf{x})_i$ requires the i^{th} node to retrieve information only from its L-hop neighbors. If the polynomial is of low order (L has a small value), then $h(\mathbf{A})\,\mathbf{x}$ can be computed locally, which is crucial to the autonomous model we consider in this study. Secondly, $\mathbf{A}$ and $h(\mathbf{A})$ have the same eigenvectors, that is,

$$h(\mathbf{A}) = \mathbf{V} \; h(\mathbf{\Lambda}) \; \mathbf{V}^{\mathrm{H}}. \tag{71}$$

Therefore, a carefully constructed polynomial can manipulate the eigenvalues of $\mathbf{A}$ in such a way that asynchronous iterations on $h(\mathbf{A})$ can be guaranteed to converge to a desired eigenspace of $\mathbf{A}$ even though iterations on $\mathbf{A}$ itself fail to do so. This idea is formally presented in the following theorem.

Theorem 4: Let λ_i denote the eigenvalues of a given graph operator $\mathbf{A}$. For a specific target eigenvalue λ_j, assume that a polynomial $h(\cdot)$ satisfies the following conditions:

$$h(\lambda_j) = 1 \qquad \text{and} \qquad |h(\lambda_i)| < 1 \quad \forall\, \lambda_i \neq \lambda_j. \tag{72}$$

Then, random updates on $h(\mathbf{A})$ as in (70) converge to an eigenvector of $\mathbf{A}$ with eigenvalue λ_j for any amount of asynchronicity $0 \leqslant \delta_T \leqslant 1$.

Proof: Since $h(\cdot)$ is assumed to satisfy (72), $h(\mathbf{A})$ has a unit eigenvalue, and the non-unit eigenvalues are strictly less than one in magnitude. Hence, Corollary 2 ensures that the updates converge to a point in the eigenspace of $h(\mathbf{A})$ with the eigenvalue 1, which is equivalent to the eigenspace of $\mathbf{A}$ with the eigenvalue λ_j due to the property in (72). ■

Polynomial filters play an important role in the area of graph signal processing. Starting from the early works [4]–[6], polynomial filters are designed to achieve a desired frequency response on the graph. On the contrary, the condition (72) disregards the overall response of the filter since its objective is to isolate a single eigenvector. Thus, the design procedure and the properties of the polynomials (to be presented in the subsequent sections) differ from the polynomial approximation ideas considered in [4]–[6].

Theorem 4 tells that an *arbitrary* eigenvector of the graph can be computed in a decentralized manner if a low order polynomial satisfying (72) is constructed. In this regard the updates on polynomial filters resemble the beam steering in antenna arrays [52]: assume that the order of the polynomial, L, is fixed. Then, the communication pattern between the nodes is completely determined by the operator $\mathbf{A}$ and the order L (See Algorithm 1). Once the nodes start to update their values randomly and asynchronously, the behavior of the signal is controlled by the polynomial coefficients. By changing the coefficients in the light of (72), one can *steer* the signal over the graph to desired "directions," which happen to be the eigenvectors of the operator. Here, L corresponds to the number of elements (sensors) in the array, and the filter coefficients serve the purpose of steering in both cases.

Note that the condition in (72) is not necessary in general for the convergence of random updates with a fixed amount of asynchronicity. (See Section IV.) However, (72) is necessary to guarantee the convergence for all levels of asynchronicity including the most restrictive case of power iteration.

Notice that if the operator $\mathbf{A}$ is the graph Laplacian, and the target eigenvalue is $\lambda_j = 0$, then the corresponding eigenvector is the constant vector. In this case the problem reduces to the consensus problem, and the condition in (72) becomes a relaxed version of the polynomial considered in [53].

In the following sections we will assume that eigenvalues of $\mathbf{A}$ are real valued, which is the case for undirected graphs, and assume that $h_n \in \mathbb{R}$. Although the complex case can also be considered in this framework, as we shall see, some of the results do not extend to the complex case.

A. The Optimal Polynomial

In this section we consider the construction of the polynomial that has the largest gap between the unit eigenvalue and the rest. In order to represent the condition (72) in the matrix-vector form we will use a vector of length $L+1$ to denote the polynomial in (69), that is, $\mathbf{h} = [h_0 \; \cdots \; h_L]^T$. In addition, let $\mathbf{\Phi}$ be a Vandermonde matrix constructed with the eigenvalues of $\mathbf{A}$ in the following form:

$$\mathbf{\Phi} = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^L \\ 1 & \lambda_2 & \lambda_2^2 & \cdots & \lambda_2^L \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \lambda_N & \lambda_N^2 & \cdots & \lambda_N^L \end{bmatrix} \in \mathbb{R}^{N \times (L+1)}. \tag{73}$$

In the case of repeated eigenvalues, we assume that the repeated rows of the matrix $\mathbf{\Phi}$ are removed. Let $\boldsymbol{\phi}_j$ denote the row of $\mathbf{\Phi}$ corresponding to the target eigenvalue λ_j, and let $\bar{\mathbf{\Phi}}_j$ denote the remaining rows of $\mathbf{\Phi}$.

In order to find the optimal L^{th} order polynomial satisfying (72), we consider the following optimization problem:

$$\begin{array}{ll} \max\limits_{c,\; \mathbf{h}} & c \\ \text{s.t.} & \begin{array}{c} \boldsymbol{\phi}_j \; \mathbf{h} = 1, \\ \big|\bar{\mathbf{\Phi}}_j \; \mathbf{h}\big| \leqslant (1-c) \; \mathbb{1}. \end{array} \end{array} \tag{74}$$

First of all notice that the constraints in (74) are linear due to the fact that $\mathbf{\Phi}$ and $\mathbf{h}$ are real valued. The objective function is linear as well. Hence, (74) is a linear programming that can be solved efficiently given the eigenvalue matrix $\mathbf{\Phi}$.

The constraints of (74) enforce the polynomial to satisfy the desired condition in (72) while the objective function maximizes the distance between the unit and non-unit eigenvalues of $h(\mathbf{A})$. Therefore, the formulation in (74) searches for the polynomial that yields the fastest rate of convergence on $h(\mathbf{A})$ among all polynomials of order L satisfying (72). Hence, we will refer to the solution of (74) as *the optimal polynomial*.

It should be noted that the solution of (74) is optimal with respect to the worst case scenario of the synchronous updates. In general, the polynomial generated via (74) may not give the fastest rate of convergence for an arbitrary amount of asynchronicity.

B. Sufficiency of Second Order Polynomials

In order to make use of the construction in (74), the order of the polynomial, L, should be selected appropriately so that the problem is feasible and admits a solution. One way to guarantee the feasibility is to select $L = N - 1$, in which case a solution always exists due to the invertibility of the Vandermonde matrix $\mathbf{\Phi}$ (by disregarding the multiplicity of eigenvalues). In this case, however, updates are no longer localized, which prevents the asynchronous model of (9) from being useful. At the other extreme, the case of $L = 1$ is insufficient to ensure the condition (72) in general. Therefore, nonlocal updates are required for the sake of flexibility in the eigenvectors. Nevertheless, the locality of the updates needs only to be compromised marginally, as the following theorem shows that $L = 2$ is in fact *sufficient* to satisfy (72).

Theorem 5: Assume that the operator $\mathbf{A}$ has real eigenvalues $\lambda_i \in \mathbb{R}$. For a given target eigenvalue λ_j, the condition in (72) is satisfied by the following second order polynomial:

$$h(\lambda) = 1 - 2\,\epsilon\,(\lambda - \lambda_j)^2 / s_j^2, \tag{75}$$

for any ϵ in $0 < \epsilon < 1$ and s_j satisfying the following:

$$s_j \geqslant \max_{1 \leqslant i \leqslant N} |\lambda_i - \lambda_j|. \tag{76}$$

Proof: It is clear that $h(\lambda_j) = 1$. In the following we will show that $-1 < h(\lambda_i) < 1$ for all $\lambda_i \neq \lambda_j$. For the upper bound note that $(\lambda_i - \lambda_j)^2 > 0$ for all $\lambda_i \neq \lambda_j$. Therefore,

$$1 - h(\lambda_i) = 2\,\epsilon\,(\lambda_i - \lambda_j)^2 / s_j^2 > 0, \tag{77}$$

which proves that $h(\lambda_i) < 1$ for all $\lambda_i \neq \lambda_j$. For the lower bound notice that we have $s_j^2 \geqslant (\lambda_i - \lambda_j)^2$ for all λ_i by the condition in (76). Therefore we have

$$h(\lambda_i) = 1 - 2\,\epsilon\,(\lambda_i - \lambda_j)^2 / s_j^2 \geqslant 1 - 2\epsilon > -1, \tag{78}$$

for all λ_i. ■

Notice that ϵ in (75) is a free parameter which can be tuned to increase the gap between the eigenvalues. Thus, the polynomial given in (75) is *not* guaranteed to be optimal in general. It merely shows that a second order polynomial satisfying (72) always exists, which also implies the feasibility of (74) in the case of $L = 2$, or larger.

An important remark is as follows: the sufficiency of second order polynomials does not extend to the complex case in general. To see this consider the following set of N complex numbers: $\lambda_n = e^{j2\pi n/(N\text{-}1)}$ for $1 \leqslant n \leqslant N$-1 and $\lambda_N = 0$. As shown in the supplementary document, *no* polynomial of order $L \leqslant N$-2 (possibly with complex coefficients) can satisfy $|h(\lambda_n)| < 1$ for $1 \leqslant n \leqslant N$-1 and $h(\lambda_N) = 1$. This adversarial example shows not only that second order polynomials are insufficient, but also that a polynomial of order N-1 is in fact necessary in the complex case in general. Although no guarantee can be provided, low order polynomials might still exist in the complex case depending on the values of the eigenvalues of a given operator $\mathbf{A}$.

C. Spectrum-Blind Construction of Suboptimal Polynomials

Although the solution of (74) provides the optimal polynomial, it requires the knowledge of all the eigenvalues of $\mathbf{A}$. Such information is not available and difficult to obtain in general. By compromising the optimality, we will discuss a way of constructing second order polynomials satisfying (72) *without* using the knowledge of all eigenvalues of $\mathbf{A}$, except the target eigenvalue λ_j.

First of all notice that a value for the coefficient s_j used in (75) can be found using only the minimum and the maximum eigenvalues of the operator. That is, the following selection

$$s_j = \max\{\lambda_{\max} - \lambda_j,\ \ \lambda_j - \lambda_{\min}\}, \tag{79}$$

satisfies (76). Therefore, the minimum, the maximum and the target eigenvalues suffice to construct the polynomial (75).

In fact (76) can be satisfied by using an appropriate upper bound for $\lambda_{\max}$ and lower bound for $\lambda_{\min}$. For example if $\mathbf{A}$ is the adjacency matrix or Laplacian, we can use the largest degree $d_{\max}$ of the graph to select s_j as follows:

1) The Laplacian: In this case the eigenvalues are bounded as $0 \leqslant \lambda_i \leqslant 2\,d_{\max}$. Hence,

$$s_j = d_{\max} + |\lambda_j - d_{\max}|. \tag{80}$$

2) The Adjacency: In this case the eigenvalues are bounded as $-d_{\max} \leqslant \lambda_i \leqslant d_{\max}$. Hence,

$$s_j = d_{\max} + |\lambda_j|. \tag{81}$$

3) The Normalized Laplacian: In this case the eigenvalues are bounded as $0 \leqslant \lambda_i \leqslant 2$. Hence,

$$s_j = 1 + |\lambda_j - 1|. \tag{82}$$

Since the selections in (80), (81), and (82) do not use the eigenvalues of the (corresponding) operator $\mathbf{A}$, the polynomial in (75) can be constructed using only the target eigenvalue λ_j.

Aforementioned constructions also provide a trade-off between the available information and the rate of convergence. This point will be elaborated in Section VII.

D. Implementation of Second Order Polynomials

In Section VI-B graph signals are shown to converge to an arbitrary eigenvector of the underlying graph operator $\mathbf{A}$ through random asynchronous updates running on an appropriate second order polynomial filter. In this section, we will show that asynchronous updates on a second order polynomial can be implemented as a node-asynchronous communication protocol in which nodes follow a collect-compute-broadcast scheme independently from each other. For this purpose, we first write a second polynomial of $\mathbf{A}$ explicitly as follows:

$$h(\mathbf{A}) = h_0\,\mathbf{I} + h_1\,\mathbf{A} + h_2\,\mathbf{A}^2, \tag{83}$$

where the filter coefficients h_0, h_1, h_2 are assumed to be predetermined such that (72) is satisfied for the eigenvalue of the target eigenvector. Then, we define an auxiliary variable $\mathbf{y}$ as:

$$\mathbf{y} = \mathbf{A}\,\mathbf{x}, \tag{84}$$

where $\mathbf{x}$ denotes the signal on the graph, and $\mathbf{y}$ is the "graph shifted" signal. We will assume that the i^{th} node stores x_i and y_i simultaneously. Thus, (x_i, y_i) can be considered as the state of the i^{th} node. Then, we can write the following:

$$(h(\mathbf{A})\,\mathbf{x})_i = h_0\,x_i + h_1\,y_i + h_2\,(\mathbf{A}\mathbf{y})_i. \tag{85}$$

Using (85), asynchronous updates with $\delta_T = 1$ (only one node is updated per iteration) running on $h(\mathbf{A})$ can be equivalently

Algorithm 1: Node-Asynchronous Communication Protocol.

procedure INITIALIZATION(i)
  Initialize x_i randomly.
  Collect x_j from nodes $j \in \mathcal{N}_{\text{in}}(i)$.
  $y_i \leftarrow \sum_{j \in \mathcal{N}_{\text{in}}(i)} A_{i,j}\, x_j$.
procedure PASSIVE STAGE(i)
  if a broadcast u is received from the node j **then**
    $y_i \leftarrow y_i + A_{i,j}\, u$.
  if the node j sends a request **then**
    Send y_i to node j.
procedure ACTIVE STAGE(i)
  Collect y_j from nodes $j \in \mathcal{N}_{\text{in}}(i)$.
  $u \leftarrow (h_0 - 1)\, x_i + h_1\, y_i + h_2 \sum_{j \in \mathcal{N}_{\text{in}}(i)} A_{i,j}\, y_j$.
  $x_i \leftarrow x_i + u$.
  Broadcast u to all nodes $j \in \mathcal{N}_{\text{out}}(i)$.

written in the following three steps:

$$u \leftarrow (h_0 - 1)\, x_i + h_1\, y_i + h_2\, \mathbf{A}_{[i,:]}\, \mathbf{y}, \tag{86}$$

$$x_i \leftarrow x_i + u, \tag{87}$$

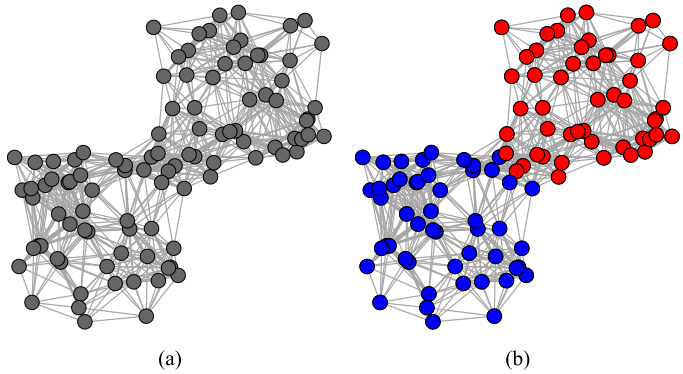
$$\mathbf{y} \leftarrow \mathbf{y} + \mathbf{A}_{[:,i]}\, u, \tag{88}$$

where $\mathbf{A}_{[i,:]}$ and $\mathbf{A}_{[:,i]}$ denote the i^{th} row and column of $\mathbf{A}$, respectively.

It is important to note that equations in (86)–(88) are in the form of a collect-compute-broadcast scheme. In (86), the term $\mathbf{A}_{[i,:]}\, \mathbf{y}$ requires the i^{th} node to collect y_j's from all $j \in \mathcal{N}_{\text{in}}(i)$. In (87), the node simply updates its own signal. In (88), the term $\mathbf{A}_{[:,i]}\, u$ requires the i^{th} node to broadcast u to all $j \in \mathcal{N}_{\text{out}}(i)$. These three steps can be converted into a node-asynchronous communication protocol as in Algorithm 1.

Algorithm 1 consists of three procedures: initialization, active stage, and passive stage. In the initialization, the i^{th} node assigns a random value to its own signal x_i, then it constructs the auxiliary variable y_i by collecting x_j from its neighbors. Once the initialization is completed, the i^{th} node waits in the passive stage, in which the graph signal x_i is not updated. However, its neighbors can request y_i, or send a broadcast. When a broadcast is received, the i^{th} node updates only its auxiliary variable y_i. When the i^{th} node wakes up randomly, it gets into the active stage, in which it collects the auxiliary variable y_j from its neighbors, updates its signal x_i, and then broadcasts the amount of update to its neighbors. Then, the node goes back to the passive stage.

Five comments are in order: 1) The random update model in (13) implies that all nodes have the same probability of going into the active stage. 2) As the signal $\mathbf{x}$ converges to the target eigenvector of $\mathbf{A}$, the ratio y_i/x_i converges to the corresponding eigenvalue of $\mathbf{A}$ (assuming x_i is non-zero), thus $h(y_i/x_i)$ converges to 1 due to (72). 3) In the active stage, the broadcast (the step in (88)) is essential to ensure that $\mathbf{x}$ and $\mathbf{y}$ satisfy (84). 4) The amount of update for x_i is computed by the i^{th} node itself. The amount of update for y_i is dictated by the neighbors of the i^{th} node. Thus, y_i can also be considered as a buffer. 5) Since edges are allowed to be directed, a node may collect data from the j^{th} node in the active stage, but may not send data back to the j^{th} node.



(a) (b)

Fig. 4. (a) A graph on $N = 100$ nodes with 2 clusters. The graph has undirected edges with binary weights. (b) The result of the spectral clustering based on (89) with colors representing the clusters.

As a final remark we note that Algorithm 1 assumes reliable communication between the nodes, i.e., no link failures. In this case Algorithm 1 is exactly equivalent to the model in (9) running on $h(\mathbf{A})$. As long as the polynomial coefficients are selected properly (see Theorem 4), the signal $\mathbf{x}$ in Algorithm 1 is guaranteed to converge to the eigenvector targeted by the polynomial. In the case of link failures, Algorithm 1 deviates from the model in (9), thus the convergence guarantees presented here are not applicable. Nevertheless, we have numerically observed that *Algorithm 1 converges even in the case of random link failures*. This case will be studied in future.

VII. An Application: Autonomous Clustering

In this section we will consider the problem of clustering in autonomous (ad-hoc) networks [22], [54], [55]. For this purpose we will combine the well-known spectral clustering [56] with the polynomial filtering proposed in the preceding section.

Given a network, the second smallest eigenvalue of its graph Laplacian, λ_2, is known as the *algebraic connectivity* of the graph [57]. Roughly speaking graphs with larger λ_2 tend to be more "connected" than the others. Furthermore, the corresponding eigenvector $\mathbf{v}_2$, also known as the *Fiedler vector*, can be utilized to cluster the graph into two partitions. The signal $\mathbf{x}$ computed as

$$\mathbf{x} = \text{sign}(\mathbf{v}_2), \tag{89}$$

indicates the corresponding cluster of the nodes. Similar spectral ideas are used in [58] to obtain approximate graph coloring and in [59] to identify the hidden M-Block cyclic structure from noisy measurements under random permutations.

In the following we will consider the idea of asynchronous polynomial filtering in order to compute the eigenvector $\mathbf{v}_2$ of the Laplacian. For this purpose λ_2 will be selected as the target eigenvalue. As a result, nodes will be able to identify the cluster they belong to in an autonomous manner. Such a behavior can be considered as swarm intelligence as well: independent simple computation by individual agents (nodes) can obtain a global information regarding the whole community (graph) [60].

For the graph visualized in Figure 4(a), the result of the spectral clustering based on (89) is demonstrated in Figure 4(b) where the clusters are represented with different colors. In the remaining, labels found by (89) will be referred to as the correct labels.

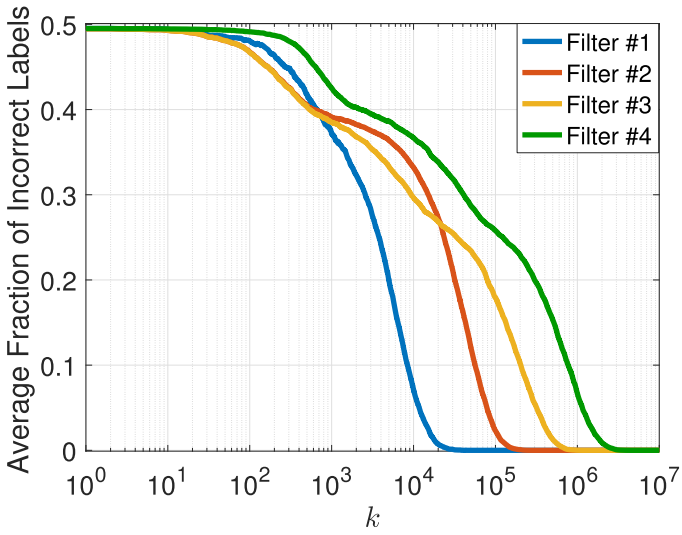


Fig. 5. Results of the autonomous clustering experiment, which are obtained by averaging over 10^4 independent experiments.

The autonomous clustering on this network is simulated using the following four different polynomial filters:

1) The optimal third order filter via (74).
2) The optimal second order filter via (74).
3) A second order filter of (75) by selecting s_2 as in (79) using λ_2, $\lambda_{\max}$, and setting $\lambda_{\min} = 0$.
4) A second order filter of (75) by selecting s_2 as in (80) using λ_2 and $d_{\max}$.

In the simulations all the nodes are initialized randomly, and the random asynchronous iterations run on the constructed polynomials of the Laplacian. We use $\delta_T = 1$, i.e., one node is randomly chosen and updated at every iteration. The label of a node is the sign of its most recent value as in (89). The average fraction of incorrect labels versus number of iterations is presented in Figure 5 for four filters mentioned above.

Figure 5 shows that the number of incorrect labels go down to zero as iterations progress, which is proven to be the case by Corollary 2 due to the construction of the filters in Section VI. The figure also illustrates the trade-off between the complexity, the amount of spectral information used and the rate of convergence. Although filters #1 and #2 are constructed using all the eigenvalues of the Laplacian, filter #1 yields a faster convergence due to its higher order (complexity). Filters #2, #3, and #4 have the same order, but their constructions use lesser and lesser amounts of spectral information. As a result, they yield lower and lower rates of convergence.

VIII. Conclusions

In this paper, we proposed a node-asynchronous communication protocol in which nodes follow a collect-compute-broadcast scheme randomly and independently from each other. Different than the consensus, this protocol can converge to an arbitrary eigenvector of the graph operator of interest. In order to analyze the convergence behavior, we introduced a randomized asynchronous variant of the power iteration, which performs the regular power iteration (or, the graph shift) but only a random subset of the indices are updated. Assuming that the underlying operator has eigenvalue 1, we proved that repetition of such randomized updates converges to an eigenvector of the eigenvalue 1 (a fixed point) even in the case of operator having other eigenvalues with magnitudes larger than one. We also showed that not only the eigenvalue gap but also the eigenspace geometry of the operator affects the behavior of the convergence. Moreover, we showed that as the updates get more asynchronous the convergence region for the eigenvalues gets larger. In order to make the updates converge to an arbitrary eigenvector, we considered polynomials of the operator. In particular, we showed that second order polynomials are sufficient to achieve such a convergence. By combining the asynchronous iterations and second order polynomials, we formally presented the node-asynchronous communication protocol. As an application, we used the proposed algorithm to compute the Fiedler vector of a network in order to achieve autonomous clustering. Simulations verified that the algorithm indeed clusters the network successfully.

Although we presented an upper and a lower bound on the rate of convergence of the randomized updates, simulations showed that neither of them is tight. We plan to obtain tighter bounds in future studies. We also plan to study the relation between the eigenspace geometry of the operator and the underlying graph structure. Such a result would relate the convergence behavior directly to the underlying graph.

Appendix A
A Useful Inequality

Lemma 3: Let $\mathbf{U} \in \mathbb{C}^{N\times M}$ and $\mathbf{X} \succeq \mathbf{0}$. Then,

$$\frac{\mathrm{tr}\big(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{X}\mathbf{U}^{\mathrm{H}})\mathbf{U}\big)}{\mathrm{tr}(\mathbf{X})} \leqslant \lambda_{\max}\big(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\big), \quad (90)$$

and

$$\frac{\mathrm{tr}\big(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{X}\mathbf{U}^{\mathrm{H}})\mathbf{U}\big)}{\mathrm{tr}(\mathbf{X})} \geqslant \lambda_{\min}\big(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\big), \quad (91)$$

♢

Proof: Consider the following problem

$$\max_{\mathbf{X}\succeq\mathbf{0}} \ \mathrm{tr}\Big(\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\mathbf{X}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\Big) \quad \text{s.t.} \quad \mathrm{tr}(\mathbf{X}) = 1. \quad (92)$$

Using the eigenvalue decomposition $\mathbf{X} = \mathbf{V}\mathrm{diag}(\boldsymbol{\lambda})\mathbf{V}^{\mathrm{H}}$, and the fact that $\mathbf{X}$ has unit trace, we can write the problem as

$$\max_{\mathbf{V},\,\boldsymbol{\lambda}} \ \mathrm{tr}\Big(\mathbf{Q}^{\mathrm{H}}\mathrm{diag}(\mathbf{Q}\mathrm{diag}(\boldsymbol{\lambda})\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\Big) \ \text{s.t.} \ \begin{cases} \mathbf{Q} = \mathbf{U}\mathbf{V} \\ \mathbf{V}^{\mathrm{H}}\mathbf{V} = \mathbf{I} \\ \mathbb{1}^{\mathrm{H}}\boldsymbol{\lambda} = 1 \\ \boldsymbol{\lambda} \geqslant \mathbf{0} \end{cases} \quad (93)$$

Notice that the objective function can be written as

$$\mathrm{tr}\Big(\mathbf{Q}^{\mathrm{H}}\mathrm{diag}(\mathbf{Q}\mathrm{diag}(\boldsymbol{\lambda})\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\Big) = \mathbb{1}^{\mathrm{H}}\,\mathbf{Z}^{\mathrm{H}}\,\mathbf{Z}\,\boldsymbol{\lambda}, \quad (94)$$

where the matrix $\mathbf{Z} \in \mathbb{R}^{N\times M}$ is defined as $Z_{i,j} = |Q_{i,j}|^2$.

Now, consider the problem of maximization over $\boldsymbol{\lambda}$

$$\max_{\boldsymbol{\lambda}} \quad \mathbb{1}^{\mathrm{H}}\,\mathbf{Z}^{\mathrm{H}}\,\mathbf{Z}\,\boldsymbol{\lambda} \quad \text{s.t.} \quad \mathbb{1}^{\mathrm{H}}\boldsymbol{\lambda} = 1, \quad \boldsymbol{\lambda} \geqslant \mathbf{0}. \quad (95)$$

Since the vector $\boldsymbol{\lambda}$ is constrained to be nonnegative and sum to 1, the objective function of (95) is the convex combination of the elements of the vector $\mathbf{Z}^{\mathrm{H}}\,\mathbf{Z}\,\mathbb{1}$, whose elements are nonnegative

as well. Since a convex combination is maximized when the largest element is selected, the solution of (95) is $\|\mathbf{Z}^{\mathrm{H}}\mathbf{Z}\,\mathbb{1}\|_\infty$. This can also be seen from the fact that the problem in (95) is in the form of the dual-norm formulation of the ℓ_1 norm.

Notice that

$$\|\mathbf{Z}^{\mathrm{H}}\mathbf{Z}\,\mathbb{1}\|_\infty = \max_{1\leqslant i\leqslant M} \left(\mathbf{Q}^{\mathrm{H}}\mathrm{diag}(\mathbf{Q}\mathbf{Q}^{\mathrm{H}})\,\mathbf{Q}\right)_{i,i}, \tag{96}$$

$$= \max_{1\leqslant i\leqslant M} \left(\mathbf{V}^{\mathrm{H}}\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\,\mathbf{V}\right)_{i,i}, \tag{97}$$

$$= \max_{1\leqslant i\leqslant M} \mathbf{v}_i^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v}_i, \tag{98}$$

where we use $\mathbf{Q}=\mathbf{U}\mathbf{V}$ from the conditions in (93), and the fact that $\mathbf{V}$ is unitary. Hence, the maximization over $\mathbf{V}$ can then be written as

$$\max_{\mathbf{v}_i} \quad \mathbf{v}_i^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v}_i \quad \text{s.t.} \quad \mathbf{v}_i^{\mathrm{H}}\,\mathbf{v}_j=\delta_{i,j}, \tag{99}$$

which can be simplified to

$$\max_{\mathbf{v}} \quad \mathbf{v}^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}\,\mathbf{v} \quad \text{s.t.} \quad \|\mathbf{v}\|_2^2=1, \tag{100}$$

whose solution is simply the largest eigenvalue of the matrix $\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\,\mathbf{U}$. Therefore for any $\mathbf{X}\succeq\mathbf{0}$ we have

$$\frac{\mathrm{tr}\left(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{X}\mathbf{U}^{\mathrm{H}})\mathbf{U}\right)}{\mathrm{tr}(\mathbf{X})} \leqslant \lambda_{\max}\left(\mathbf{U}^{\mathrm{H}}\mathrm{diag}(\mathbf{U}\mathbf{U}^{\mathrm{H}})\mathbf{U}\right) \tag{101}$$

For the inequality in (91), the maximization in (95) is replaced with the minimization. Since the minimum of the objective function is achieved when the minimum element of $\mathbf{Z}^{\mathrm{H}}\mathbf{Z}\,\mathbb{1}$ is selected, maximization in (96) (hence the one in (100)) is replaced with minimization, which yields the minimum eigenvalue problem in (100). This validates the inequality in (91). ■

Appendix B
Proof of Theorem 2

Consider the following covariance matrix:

$$\mathbf{C}_k = \mathbb{E}[\mathbf{U}^{\mathrm{H}}\mathbf{x}_k\,\mathbf{x}_k^{\mathrm{H}}\,\mathbf{U}], \tag{102}$$

which gives $\mathrm{tr}(\mathbf{C}_k)=\mathbb{E}\left[\,\|\mathbf{U}^{\mathrm{H}}\mathbf{x}_k\|_2^2\,\right]=\mathbb{E}[\,\|\mathbf{r}_k\|_2^2\,]$. We can write $\mathbf{C}_k$ as follows

$$\mathbf{C}_k = \mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{I}\,\mathbf{x}_{k\text{-}1}\,\mathbf{x}_{k\text{-}1}^{\mathrm{H}}\,\mathbf{I}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}]. \tag{103}$$

Notice that we have $\mathbf{I}=\mathbf{V}_1\mathbf{V}_1^{\mathrm{H}}+\mathbf{U}\mathbf{U}^{\mathrm{H}}$. Furthermore, $\mathbf{Q}\,\mathbf{V}_1=\mathbf{V}_1$ is valid for *any* outcome of the random matrix $\mathbf{Q}$ in (11). We note that $\mathbf{V}_1^{\mathrm{H}}\,\mathbf{Q}\neq\mathbf{V}_1^{\mathrm{H}}$ in general. Further using the fact that $\mathbf{V}_1^{\mathrm{H}}\,\mathbf{U}=\mathbf{0}$, we can write (103) as follows:

$$\mathbf{C}_k = \mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_{k\text{-}1}\,\mathbf{x}_{k\text{-}1}^{\mathrm{H}}\,\mathbf{U}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}], \tag{104}$$

$$= \mathbb{E}\left[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbb{E}[\mathbf{U}^{\mathrm{H}}\,\mathbf{x}_{k\text{-}1}\,\mathbf{x}_{k\text{-}1}^{\mathrm{H}}\,\mathbf{U}]\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}\right], \tag{105}$$

$$= \mathbb{E}\left[\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k\,\mathbf{U}\,\mathbf{C}_{k\text{-}1}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}_k^{\mathrm{H}}\,\mathbf{U}\right], \tag{106}$$

where (105) follows from the fact that each update is independent of the previous ones.

In order to compute the expectation in (106), in the following we will first find the conditional expectation (conditioned on T) as

$$\bar{\mathbf{C}}_k = \mathbb{E}[\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}\,\mathbf{U}\,\mathbf{C}_{k\text{-}1}\,\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}^{\mathrm{H}}\,\mathbf{U}\;|\;T\,]. \tag{107}$$

Then, $\mathbf{C}_k$ will be found as $\mathbf{C}_k=\mathbb{E}[\,\bar{\mathbf{C}}_k\,]$ where the expectation is with respect to T.

Since $\mathbf{A}$ is assumed to be normal matrix, its eigenspaces are orthonormal to each other. Therefore, the matrix $\mathbf{U}$ corresponds to the eigenvectors of $\mathbf{A}$ with non-unit eigenvalues. Furthermore, the normality implies also that the left and the right eigenvectors of $\mathbf{A}$ are conjugate transpose of each other. That is, $\mathbf{V}_1^{\mathrm{H}}\,(\mathbf{A}-\mathbf{I})=\mathbf{0}$, and $\mathbf{U}^{\mathrm{H}}\,(\mathbf{A}-\mathbf{I})=\mathbf{\Sigma}\,\mathbf{U}^{\mathrm{H}}$, where $\mathbf{\Sigma}\in\mathbb{C}^{(N\text{-}M)\times(N\text{-}M)}$ is a diagonal matrix with diagonal entries $\{\sigma_1,\,\ldots,\,\sigma_{N\text{-}M}\}$ where $\sigma_i=\lambda_i-1$. Thus, the normality of $\mathbf{A}$ implies the following equality:

$$\mathbf{U}^{\mathrm{H}}\,\mathbf{Q}(\mathcal{T})\,\mathbf{U}=\mathbf{I}+\mathbf{U}^{\mathrm{H}}\,\mathbf{D}_{\mathcal{T}}\,\mathbf{U}\,\mathbf{\Sigma}. \tag{108}$$

Using (108) in (107), the expectation can be written as:

$$\bar{\mathbf{C}}_k = \frac{1}{\binom{N}{T}}\sum_{\mathcal{T}}\Big(\mathbf{I}+\mathbf{U}^{\mathrm{H}}\mathbf{D}_{\mathcal{T}}\,\mathbf{U}\,\mathbf{\Sigma}\Big)\mathbf{C}_{k\text{-}1}\Big(\mathbf{I}+\mathbf{\Sigma}^{\mathrm{H}}\mathbf{U}^{\mathrm{H}}\mathbf{D}_{\mathcal{T}}\,\mathbf{U}\Big),$$

$$= \mathbf{C}_{k\text{-}1}+\frac{T}{N}\Big(\mathbf{\Sigma}\,\mathbf{C}_{k\text{-}1}+\mathbf{C}_{k\text{-}1}\mathbf{\Sigma}^{\mathrm{H}}\Big)+\frac{T(T-1)}{N(N-1)}\mathbf{\Sigma}\,\mathbf{C}_{k\text{-}1}\mathbf{\Sigma}^{\mathrm{H}}$$

$$+\frac{T(N-T)}{N(N-1)}\,\mathbf{U}^{\mathrm{H}}\,\mathrm{diag}(\mathbf{U}\,\mathbf{\Sigma}\,\mathbf{C}_{\mathrm{k}\text{-}1}\,\mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{U}^{\mathrm{H}})\,\mathbf{U}, \tag{109}$$

where the last step follows from (5) and (6).

By taking the trace of both sides of (109) and using (90) from Lemma 3 we obtain the following:

$$\mathrm{tr}(\bar{\mathbf{C}}_k)\leqslant\mathrm{tr}\Bigg(\mathbf{C}_{k\text{-}1}\,\Bigg[\mathbf{I}+\frac{T}{N}\big(\mathbf{\Sigma}+\mathbf{\Sigma}^{\mathrm{H}}+\mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{\Sigma}\big)+\frac{T(N-T)}{N(N-1)}\,(\rho^{+}-1)\,\mathbf{\Sigma}^{\mathrm{H}}\,\mathbf{\Sigma}\Bigg]\Bigg), \tag{110}$$

where ρ^{+} is defined as in (39). By taking the expectation of both sides in (110) with respect to T, we get the following:

$$\mathrm{tr}(\mathbf{C}_k)\leqslant\mathrm{tr}\Bigg(\mathbf{C}_{k\text{-}1}\,\Bigg[\mathbf{I}+\frac{\mu_T}{N}\big(\mathbf{\Sigma}+\mathbf{\Sigma}^{\mathrm{H}}+\mathbf{\Sigma}^{\mathrm{H}}\mathbf{\Sigma}\big)+\frac{\mu_T}{N}\,\delta_T\,(\rho^{+}-1)\,\mathbf{\Sigma}^{\mathrm{H}}\mathbf{\Sigma}\Bigg]\Bigg), \tag{111}$$

$$= \sum_{i=1}^{N-M}(\mathbf{C}_{k\text{-}1})_{i,i}\;\;\Psi_i\;\leqslant\;\Psi\,\mathrm{tr}(\mathbf{C}_{\mathrm{k}\text{-}1}), \tag{112}$$

where Ψ_i's are defined as

$$\Psi_i=1+\frac{\mu_T}{N}\big(|\lambda_i|^2-1\big)+\frac{\mu_T}{N}\,\delta_T\,(\rho^{+}-1)\,|\lambda_i-1|^2, \tag{113}$$

and Ψ is defined as $\Psi=\max_i\Psi_i$. Iterative use of the inequality in (112) gives the upper bound in (43).

For the lower bound in (43), we consider the trace of both sides of (109), use (91) from Lemma 3, and take the expectation with respect to T. This will provide a lower bound for $\mathrm{tr}(\mathbf{C}_k)$ similar to (111) where ρ^{+} is replaced with ρ^{-} from (40). Hence, we get

$$\mathrm{tr}(\mathbf{C}_k)\geqslant\sum_{i=1}^{N-M}(\mathbf{C}_{k\text{-}1})_{i,i}\;\;\psi_i\;\geqslant\;\psi\,\mathrm{tr}(\mathbf{C}_{k\text{-}1}), \tag{114}$$

where ψ_i's are defined as

$$\psi_i = 1 + \frac{\mu_T}{N}\left(|\lambda_i|^2 - 1\right) + \frac{\mu_T}{N}\,\delta_T\,(\rho^- - 1)\,|\lambda_i - 1|^2, \quad (115)$$

and ψ is defied as $\psi = \min_i \psi_i$. Iterative use of the inequality in (114) gives the lower bound in (43).

APPENDIX C
A COUNTER-EXAMPLE TO THE INSUFFICIENCY OF SECOND ORDER POLYNOMIALS IN THE COMPLEX CASE

Consider the following set of N complex numbers: $\lambda_n = e^{j2\pi n/(N\text{-}1)}$ for $1 \leqslant n \leqslant N$-1 and $\lambda_N = 0$. Assume that there exists a polynomial of order $L \leqslant N$-2 such that it satisfies the following mapping:

$$\left|h(\lambda_n)\right| < 1 \quad \text{for} \quad 1 \leqslant n \leqslant N\text{-}1, \quad (116)$$

$$h(\lambda_N) = 1. \quad (117)$$

When the polynomial is written explicitly as $h(\lambda) = \sum_{k=0}^{L} h_k\,\lambda^k$, the condition in (116) can be written as

$$\left|\sum_{k=0}^{L} h_k\, e^{j\,2\,\pi\,n\,k/(N\text{-}1)}\right|^2 = \sum_{k,s=0}^{L} h_k\, h_s^*\, e^{j\,2\,\pi\,n\,(k-s)/(N\text{-}1)} < 1. \quad (118)$$

Notice that the inequality in (118) holds true for all n in $1 \leqslant n \leqslant N$-1. Therefore, the following also holds true:

$$N - 1 > \sum_{n=1}^{N\text{-}1} \sum_{k,s=0}^{L} h_k\, h_s^*\, e^{j\,2\,\pi\,n\,(k-s)/(N\text{-}1)}, \quad (119)$$

$$= \sum_{k,s=0}^{L} h_k\, h_s^* \sum_{n=1}^{N\text{-}1} e^{j\,2\,\pi\,n\,(k-s)/(N\text{-}1)}. \quad (120)$$

Note that $0 \leqslant k, s \leqslant L \leqslant N$-2. As a result, the inner summation in (120) is nonzero if and only if $k = s$. That is,

$$\sum_{n=1}^{N\text{-}1} e^{j\,2\,\pi\,n\,(k-s)/(N\text{-}1)} = (N\text{-}1)\;\;\delta_{k,s}, \quad (121)$$

where $\delta_{k,s}$ stands for the Dirac delta function. Then, the inequality in (120) becomes:

$$1 > \sum_{k,s=0}^{L} h_k\, h_s^*\, \delta_{k,s} = \sum_{k=0}^{L} |h_k|^2. \quad (122)$$

Notice that the condition in (117) implies that $h_0 = 1$. Therefore, (122) can be written as:

$$1 > \sum_{k=0}^{L} |h_k|^2 = 1 + \sum_{k=1}^{L} |h_k|^2, \quad (123)$$

which implies $0 > \sum_{k=1}^{L} |h_k|^2$, which is a contradiction. Hence, no polynomial (possibly with complex coefficients) of order $L \leqslant N$-2 can satisfy the conditions in (116) and (117). Therefore, a polynomial of order N-1 is in fact necessary in the complex case in general.

ACKNOWLEDGMENT

The authors are grateful to the reviewers for their very helpful suggestions and for drawing our attention to a wealth of literature on relevant work.

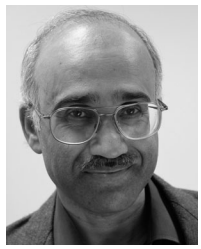
REFERENCES

[1] M. E. J. Newman, *Networks: An Introduction*. Oxfrod, U.K.: Oxford Univ. Press, 2010.
[2] M. Jackson, *Social and Economic Networks*. Princeton, NJ, USA: Princeton Univ. Press, 2008.
[3] M. H. Degroot, "Reaching a consensus," *J. Amer. Statist. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.
[4] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.
[5] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, Sep. 2014.
[6] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, Jun. 2014.
[7] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.
[8] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Trans. Signal Process.*, vol. 65, no. 22, pp. 5911–5926, Nov. 2017.
[9] A. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Apr. 2016.
[10] S. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2786–2799, Jun. 2012.
[11] S. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4673–4685, Oct. 2013.
[12] O. Teke and P. P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs—Part I: Fundamentals," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 409–422, Jan. 2017.
[13] O. Teke and P. P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs—Part II: M-Channel filter banks," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 423–437, Jan. 2017.
[14] X. Wang, P. Liu, and Y. Gu, "Local-set-based graph signal reconstruction," *IEEE Trans. Signal Process.*, vol. 63, no. 9, pp. 2432–2444, May 2015.
[15] A. Agaskar and Y. M. Lu, "A spectral graph uncertainty principle," *IEEE Trans. Inf. Theory*, vol. 59, no. 7, pp. 4338–4356, Jul. 2013.
[16] O. Teke and P. P. Vaidyanathan, "Uncertainty principles and sparse eigenvectors of graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5406–5420, Oct. 2017.
[17] D. Romero, M. Ma, and G. B. Giannakis, "Kernel-based reconstruction of graph signals," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 764–778, Feb. 2017.
[18] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 856–869, Sep. 2017.
[19] A. Venkitaraman, S. Chatterjee, and P. Händel, "Kernel regression for signals over graphs," Nov. 2018, arXiv:1706.02191v5.
[20] Z. J. Haas, J. Deng, B. Liang, P. Papadimitratos, and S. Sajama, *Wireless Ad Hoc Networks*. Hoboken, NJ, USA: Wiley, 2003.
[21] B. Percha, R. Dzakpasu, M. Zochowski, and J. Parent, "Transition from local to global phase synchrony in small world neural network and its possible implications for epilepsy," *Phys. Rev. E*, vol. 72, Sep. 2005, Art. no. 031909.
[22] V. Kumar, D. Rus, and G. S. Sukhatme, *Networked Robots*. Berlin, Germany: Springer, 2008, pp. 943–958.
[23] J. A. Deri and J. M. F. Moura, "Spectral projector-based graph Fourier transforms," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 785–795, Sep. 2017.
[24] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Nashua, NH, USA: Athena Scientific, 1997.

[25] D. P. Bertsekas, "Distributed asynchronous computation of fixed points," *Math. Program.*, vol. 27, no. 1, pp. 107–120, Sep. 1983.
[26] G. M. Baudet, "Asynchronous iterative methods for multiprocessors," *J. ACM*, vol. 25, no. 2, pp. 226–244, Apr. 1978.
[27] D. Chazan and W. Miranker, "Chaotic relaxation," *Linear Algebra Appl.*, vol. 2, pp. 199–222, Apr. 1969.
[28] S. J. Wright, "Coordinate descent algorithms," *Math. Program.*, vol. 151, no. 1, pp. 3–34, Jun. 2015.
[29] Q. Lei, K. Zhong, and I. S. Dhillon, "Coordinate-wise power method," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2064–2072.
[30] J. Wang, W. Wang, D. Garber, and N. Srebro, "Efficient coordinate-wise leading eigenvector computation," 2017, arXiv:1702.07834.
[31] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.
[32] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. 44th IEEE Conf. Decision Control*, Dec. 2005, pp. 2996–3000.
[33] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.
[34] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.
[35] A. Tahbaz-Salehi and A. Jadbabaie, "A necessary and sufficient condition for consensus over random networks," *IEEE Trans. Autom. Control*, vol. 53, no. 3, pp. 791–795, Apr. 2008.
[36] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.
[37] S. Kar and J. Moura, "Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 355–369, Jan. 2009.
[38] S. Kar and J. M. F. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3315–3326, Jul. 2008.
[39] S. Kar, J. M. F. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Trans. Inf. Theory*, vol. 58, no. 6, pp. 3575–3605, Jun. 2012.
[40] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization," in *Proc. IEEE Conf. Decis. Control*, Dec. 2012, pp. 5453–5458.
[41] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proc. IEEE Symp. Found. Comp. Sci.*, Oct. 2003, pp. 482–491.
[42] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015.
[43] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, Sep. 1986.
[44] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.
[45] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," in *Proc. IEEE Conf. Decis. Conrol*, Dec. 2013, pp. 3671–3676.
[46] P. Bianchi and J. Jakubowicz, "Convergence of a multi-agent projected stochastic gradient algorithm for non-convex optimization," *IEEE Trans. Autom. Control*, vol. 58, no. 2, pp. 391–405, Feb. 2013.
[47] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Filtering random graph processes over random time-varying graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4406–4421, Aug. 2017.
[48] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, no. 1, pp. 65–78, 2004.
[49] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci.*, vol. 79, no. 8, pp. 2554–2558, 1982.
[50] P. J. Uhlhaas and W. Singer, "Neural synchrony in brain disorders: Relevance for cognitive dysfunctions and pathophysiology," *Neuron*, vol. 52, no. 1, pp. 155–16, 2006.
[51] O. Teke and P. P. Vaidyanathan, "The asynchronous power iteration: A graph signal perspective," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2018, pp. 4059–4063.
[52] H. L. Van Trees, *Optimum Array Processing*. Hoboken, NJ, USA: Wiley, 2002.
[53] A. Sandryhaila, S. Kar, and J. M. F. Moura, "Finite-time distributed consensus through graph filters," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, May 2014, pp. 1080–1084.
[54] O. Younis and S. Fahmy, "HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mobile Comput.*, vol. 3, no. 4, pp. 366–379, Oct. 2004.
[55] S. Basagni, "Distributed clustering for ad hoc networks," in *Proc. Parallel Architectures, Algorithms, Netw.*, 1999, pp. 310–315.
[56] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Neural Inf. Process. Syst.*, 2002, pp. 849–856.
[57] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Math. J.*, vol. 23, no. 2, pp. 298–305, 1973.
[58] B. Aspvall and J. R. Gilbert, "Graph coloring using eigenvalue decomposition," *SIAM J. Algebr. Discrete Methods*, vol. 5, no. 4, pp. 526–538, 1984.
[59] O. Teke and P. P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs," *Proc. SPIE*, vol. 10394, Aug. 2017, Art. no. 103941R.
[60] E. Bonabeau, G. Theraulaz, and M. Dorigo, *Swarm Intelligence: From Natural to Artificial Systems*. Hyderabad, India: Univ. Press, 1999.

Oguzhan Teke (S'12) received the B.S. and the M.S. degrees in electrical and electronics engineering from Bilkent University, Ankara, Turkey, in 2012, and 2014, respectively. He is currently working toward the Ph.D. degree in electrical engineering at the California Institute of Technology, Pasadena, CA, USA.

His research interests include signal processing, graph signal processing, and convex optimization.



Palghat P. Vaidyanathan (S'80–M'83–SM'88–F'91) was born in Calcutta, India, on October 16, 1954. He received the B.Sc. (Hons.) degree in physics and the B.Tech. and M.Tech. degrees in radiophysics and electronics, all from the University of Calcutta, Kolkata, India, in 1974, 1977, and 1979, respectively, and the Ph.D degree in electrical and computer engineering from the University of California at Santa Barbara, Santa Barbara, CA, USA, in 1982. He was a Postdoctoral Fellow with the University of California at Santa Barbara from September 1982 to March 1983. In March 1983, he joined the Electrical Engineering Department, California Institute of Technology, Pasadena, CA, USA, as an Assistant Professor, and is currently the Kio and Eiko Tomiyasu Professor of Electrical Engineering there. His main research interests are in digital signal processing, multirate systems, wavelet transforms, signal processing for digital communications, genomic signal processing, and sparse array signal processing.

Prof. Vaidyanathan has authored more than 550 papers in journals and conferences, and is the author/coauthor of the four books: *Multirate Systems and Filter Banks* (Prentice-Hall, 1993), *Linear Prediction Theory* (Morgan and Claypool, 2008), and (with Phoong and Lin) *Signal Processing and Optimization for Transceiver Systems*, (Cambridge University Press, 2010), and *Filter Bank Transceivers for OFDM and DMT Systems*, (Cambridge University Press, 2010). He was a recipient of the Award for excellence in teaching at the California Institute of Technology multiple times. In 2016, he received the Northrup Grumman Prize for excellence in Teaching at Caltech. He also received the NSF's Presidential Young Investigator award in 1986. About 15 of his papers have received prizes from IEEE and other organizations. He received the 1995 F. E. Terman Award of the American Society for Engineering Education, sponsored by Hewlett Packard Co., for his contributions to engineering education. He has given several plenary talks at several conferences and has been chosen a distinguished lecturer for the IEEE Signal Processing Society for the year 1996–97. In 1999, he was chosen to receive the IEEE CAS Society's Golden Jubilee Medal. He is a recipient of the IEEE Gustav Kirchhoff Award (an IEEE Technical Field Award) in 2016, the IEEE Signal Processing Society's Technical Achievement Award in 2002, Education Award in 2012, and the Society Award in 2016. In 2019, he was elected to the U.S. National Academy of Engineering.