NODE-ASYNCHRONOUS SPECTRAL CLUSTERING ON DIRECTED GRAPHS

Oguzhan Teke and P. P. Vaidyanathan

Department of Electrical Engineering, MC 136-93,
California Institute of Technology, Pasadena, CA 91125, USA
E-mail: oteke@caltech.edu, ppvnath@systems.caltech.edu

ABSTRACT

In recent years the convergence behavior of random node asynchronous graph communications have been studied for the case of undirected graphs. This paper extends these results to the case of graphs having arbitrary directed edges possibly with a non-diagonalizable adjacency matrix. Assuming that the graph operator has eigenvalue 1 and the input signal satisfies a certain condition (which ensures the existence of fixed points), this study presents the necessary and sufficient condition for the mean-squared convergence of the graph signal. The presented condition depends on the graph operator as well as the update probabilities, and the convergence of the randomized asynchronous updates may be achieved even when the underlying operator is not stable in the synchronous setting. As an application, the node-asynchronous updates are combined with polynomial filtering in order to obtain a spectral clustering for directed networks. The convergence is also verified with numerical simulations.

Index Terms— Graph signal processing, node-asynchronous iterations, autonomous clustering, directed graphs.

1. INTRODUCTION

Graph signal processing provides versatile tools for the analysis of data defined over irregular domains, which are typically modeled as networks with the underlying graph representing the dependency structure [1–5]. This broad model is applicable to the analysis of data in various different fields such as social, economic, and biological networks, among others [6, 7].

The analysis in graph signal processing is based on the "graph operator," whose eigenvectors serve as the graph Fourier basis. With the use of this basis, sampling, reconstruction, multirate processing of graph signals and some uncertainty results have been extended to the case of graphs [8–16]. Another important notion in this area is the "graph shift," which is also based on the graph operator. Since the operator is assumed to follow the connectivity structure of the underlying graph, the graph shift has a localized and distributed implementation. That is, nodes on the graph can execute a graph shift on their own simply by exchanging data with their neighbors and computing a weighted average. With the use of successive communications (successive graph shifts), the notion of polynomial and rational filtering is successfully extended to the case of graphs [17–26].

Although the graph shift can be implemented in a distributed fashion, it requires simultaneous communication. That is, all the nodes should send and receive data at the same time instance, or nodes should wait until all the communications are terminated before proceeding to the next graph shift. Although a synchronization mechanism can be integrated into the communication protocol, it becomes an important limitation when the size of the network is large, or the network has autonomous behavior.

In recent years, the studies [27–32] showed that synchronicity is not necessary for the stability of graph communications in general. That is, a node can "wake-up" at a random time instance independent of the others, do the necessary computations on its own, and then broadcast information to its neighbors. It is clear that such a randomized behavior introduces stochasticity into the signal on the graph. It is proven under mild conditions that such signals converge (in stochastic mean-squared sense) even when the nodes communicate with each other randomly and asynchronously.

It is also important to point out that the convergence behavior depends on the precise assumptions made regarding the stochastic behavior of the nodes and the underlying graph. In particular, the studies [27, 28] consider the case of zero input (i.e., power iteration) and prove the convergence of randomized updates assuming that the graph operator is a normal matrix and the indices are equally likely to be updated in every iteration. Since graphs with undirected edges have symmetric (hence, normal) graph operators, convergence results of [27, 28] are applicable mainly to undirected graphs. In fact, the autonomous clustering example studied in [27] assumes undirected graphs exclusively. When the underlying graph has directed edges, or nodes are updated with non-equal probabilities the results of [27, 28] do not hold true in general.

In this study, we will extend the convergence results of [27, 28] to the case of arbitrary (possibly non-diagonalizable) matrices with arbitrary update probabilities. We will also allow the input signal to be a nonzero constant with an additive zero mean noise. With the removal of the normality assumption on the graph operator, the results presented here can guarantee the convergence of randomized node-asynchronous updates even for graphs with arbitrary directed edges. More importantly, this study will present the necessary and sufficient condition for the mean-squared convergence of the randomized updates, whereas [27, 28] present a sufficiency condition only. Furthermore, this study shows the effect of the input noise precisely. As an application, we will re-visit the clustering example in [27] and show that the spectral clustering (based on the graph Laplacian) can be obtained with randomized asynchronous updates even in the case of directed graphs.

In Section 2, we consider synchronous state recursions and discuss the conditions under which the recursions have a fixed point. In Section 2.2, we present the necessary and sufficient condition for the mean-squared convergence (Theorem 1). Then, in Section 3 we use a combination of polynomial graph filters and random asynchronous updates in order to achieve spectral clustering for directed graphs. In Section 3.3 we experimentally verify the convergence.

1.1. Preliminaries and Notation

We will use $\otimes$ to denote the Kronecker product. For a matrix $\mathbf{X}$ we will use $\mathrm{tr}(\mathbf{X})$ to denote its trace, and $\rho(\mathbf{X})$ to denote its spectral radius (the largest eigenvalue in absolute sense). We will use $\mathbf{X}^*$, $\mathbf{X}^{\mathrm{H}}$ and $\mathbf{X}^{\dagger}$ to denote its conjugate, conjugate transpose, and pseudo-inverse, respectively. We will use $\mathrm{vec}(\mathbf{X})$ to denote a vector obtained by cascading the columns of $\mathbf{X}$, and $\mathrm{diag}(\mathbf{X})$ to denote the diagonal masking of $\mathbf{X}$. We define the matrix $\mathbf{J}$ as follows:

$$\mathbf{J} = \sum_{i=1}^{N} (\mathbf{e}_i\, \mathbf{e}_i^{\mathrm{H}}) \otimes (\mathbf{e}_i\, \mathbf{e}_i^{\mathrm{H}}) \in \mathbb{R}^{N^2 \times N^2}, \tag{1}$$

where $\mathbf{e}_i$ denotes the i^{th} vector of the standard basis of size N.

This work was supported in parts by the ONR grant N00014-18-1-2390, the NSF grant CCF-1712633, and the Electrical Engineering Carver Mead Research Seed Fund of the California Institute of Technology.

Authorized licensed use limited to: CALIFORNIA INSTITUTE OF TECHNOLOGY. Downloaded on July 02,2021 at 15:57:10 UTC from IEEE Xplore. Restrictions apply.

2. ON ASYNCHRONOUS STATE RECURSIONS

Given an *arbitrary* matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$ and an input signal $\mathbf{u} \in \mathbb{C}^N$, we will consider the following type of recursion on a state vector $\mathbf{x}_k$:

$$\mathbf{x}_{k+1} = \mathbf{A}\,\mathbf{x}_k + \mathbf{u} + \mathbf{w}_k, \tag{2}$$

where $\mathbf{x}_0$ is the initial state vector, and $\mathbf{w}_k$ denotes noise with the following statistics:

$$\mathbb{E}[\mathbf{w}_k] = \mathbf{0}, \qquad \mathbb{E}[\,\mathbf{w}_k\,\mathbf{w}_s^{\mathrm{H}}\,] = \delta(k-s)\,\boldsymbol{\Gamma}, \tag{3}$$

where $\delta(\cdot)$ denotes the discrete Dirac delta function.

When considered from the viewpoint of graph signal processing, an iteration in the form of (2) can be implemented on a graph (modeled by the matrix $\mathbf{A}$) in a distributed fashion as a data exchange between the neighborhood nodes [17–27]. The vector $\mathbf{u}$ becomes the input signal defined on the graph, where the nodes are the "domain" analogous to time. The index k denotes the round of communication, so the graph signal $\mathbf{u}$ does not have any dependency on the iteration index k in the model (2), but the value of $\mathbf{w}_k$ is different in each update cycle due to it being random.

The mathematical model in (2) requires all the nodes to communicate simultaneously, or wait for each other, before starting the next round of communication. Thus, an implementation of (2) on a graph needs a synchronization over the network, which becomes an important limitation when the size of the network is large, or the network has an autonomous behavior. In order to eliminate the need for synchronization, the recent works [27–32] consider a randomized asynchronous variant of the state recursions, in which only a random subset of indices are updated simultaneously and the remaining ones stay unchanged in each iteration of (2). More precisely,

$$(\mathbf{x}_{k+1})_i = \begin{cases} (\mathbf{A}\,\mathbf{x}_k)_i + u_i + (\mathbf{w}_k)_i, & i \in \mathcal{T}_k, \\ (\mathbf{x}_k)_i, & i \notin \mathcal{T}_k, \end{cases} \tag{4}$$

where u_i denotes the i^{th} index of the input signal, and $\mathcal{T}_k$ denotes the set of randomly selected indices updated at the iteration k.

In the asynchronous model (4), it is assumed that the i^{th} index of $\mathbf{x}_k$ is updated randomly and independently with probability p_i in every iteration of (4). We will use $\mathbf{P}$ to denote the diagonal matrix consisting of the probability p_i's. More precisely,

$$\mathbf{P} = \operatorname{diag}\left(\begin{bmatrix} p_1 & p_2 & \cdots & p_N \end{bmatrix}\right) \in \mathbb{R}^{N\times N}. \tag{5}$$

The studies in [27–32] consider the convergence properties of the recursions in (4) under different assumptions regarding the matrix $\mathbf{A}$, the input $\mathbf{u}$, and the update probabilities of the indices. More precisely, the studies [27, 28] work under the following three assumptions: the input is identically zero, $\mathbf{A}$ is a normal matrix and has eigenvalue(s) equal to 1, and all the indices are updated with equal probabilities. Due to the normality assumption on $\mathbf{A}$, these results are applicable mainly to undirected graphs, and they are inconclusive regarding arbitrary directed graphs. Furthermore, the equal update probabilities may not be practical to obtain in some applications.

The second set of studies in [29,30] allow the graph to have arbitrary directed edges and nodes to be updated with arbitrary probabilities. However, their convergence results are based on the assumption that the matrix $\mathbf{A}$ does *not* have an eigenvalue 1. Although these results are useful for asynchronous implementation of rational graph filters, in the absence of eigenvalue 1 of the matrix $\mathbf{A}$ the random asynchronous updates do not converge to eigenvectors of the graph operator. Thus, applications that require the computation of these eigenvectors (e.g., autonomous spectral clustering) cannot make use of the randomized asynchronous updates.

In this study, we will focus on the recursions in (4) as well. However, we will consider a more general case that extends the scenarios studied both in [27, 28] and [29, 30]. So, the setting here reduces to those in [27–30] when additional assumptions are made. That is, *we will allow the graph to have arbitrary directed edges, the nodes to be updated with arbitrary probabilities. Moreover the graph operator is allowed to have unit eigenvalue(s), and the input can be nonzero.* More importantly, this study will present *the necessary and sufficient* condition for the mean-squared convergence, whereas both [27, 28] and [29, 30] have sufficiency conditions only.

2.1. Fixed Points of the Recursion

We first point out that fixed points of the state recursions (whether implemented synchronously or asynchronously) are determined by the matrix $\mathbf{A}$ and input vector $\mathbf{u}$ only. More precisely, a fixed point of the update schemes in (2) and (4) satisfies the following equation:

$$(\mathbf{I} - \mathbf{A})\,\mathbf{x} = \mathbf{u}, \tag{6}$$

which *may* or *may not* have a solution depending on the input vector $\mathbf{u}$ and the eigenvalues of the matrix $\mathbf{A}$. Throughout this study we assume that a fixed point satisfying (6) exists. More precisely, *we always assume that the following holds true regarding the input:*

$$\mathbf{u} \in \operatorname{col}(\mathbf{I} - \mathbf{A}). \tag{7}$$

Let m denote the geometric multiplicity of the eigenvalue 1 of the matrix $\mathbf{A}$, so the null space of the matrix $\mathbf{I} - \mathbf{A}$ has dimension m. We note that the matrix $\mathbf{A}$ need not be symmetric (even diagonalizable) in general. We also allow the possibility of $m = 0$, which corresponds to the case of $\mathbf{A}$ not having eigenvalue 1, in which case (7) is satisfied for any input signal $\mathbf{u}$.

When $m \geqslant 1$, there are infinitely many fixed points corresponding to the solutions of (6). In such case we consider a specific fixed point, denoted as $\mathbf{x}^\star$, that corresponds to the fixed point with the minimum ℓ_2-norm. More precisely, we define $\mathbf{x}^\star$ as follows:

$$\mathbf{x}^\star \triangleq (\mathbf{I} - \mathbf{A})^\dagger\,\mathbf{u} = \left\{ \arg\min_{\mathbf{x}} \|\mathbf{x}\|_2 \quad \text{s.t.} \quad (\mathbf{I} - \mathbf{A})\,\mathbf{x} = \mathbf{u} \right\}. \tag{8}$$

Then, a fixed point $\mathbf{x}$ satisfying (6) can be decomposed as follows:

$$\mathbf{x} = \mathbf{x}^\star + \mathbf{c} \qquad \text{where} \qquad \mathbf{c} \in \operatorname{null}(\mathbf{I} - \mathbf{A}). \tag{9}$$

In the rest of the paper will use $\mathbf{V}_1 \in \mathbb{C}^{N\times m}$ to denote an orthonormal basis for the null-space of $\mathbf{I} - \mathbf{A}$, and we will use $\mathbf{Q} \in \mathbb{C}^{N\times N}$ to denote the projection matrix onto the orthogonal complement of the null-space of $\mathbf{I} - \mathbf{A}$. More precisely:

$$\operatorname{col}(\mathbf{V}_1) = \operatorname{null}(\mathbf{I} - \mathbf{A}), \qquad \text{and} \qquad \mathbf{Q} = \mathbf{I} - \mathbf{V}_1\,\mathbf{V}_1^\dagger. \tag{10}$$

We note that when $\mathbf{A}$ does not have an eigenvalue equal to 1, i.e., for the case of $m = 0$, the matrix $\mathbf{I} - \mathbf{A}$ becomes full-rank, the projection matrix defined in (10) reduces to $\mathbf{Q} = \mathbf{I}$, and $\mathbf{x}^\star$ defined in (8) becomes the unique fixed point satisfying (6). Thus, the results (to be presented next) are valid even when the matrix $\mathbf{A}$ does not have an eigenvalue 1, and they extend the results of [29, 30].

The case of zero input, i.e., $\mathbf{u} = \mathbf{0}$, is an important special case, as the range space condition in (7) is readily satisfied irrespective of the matrix $\mathbf{A}$. Furthermore, the point defined in (8) reduces to $\mathbf{x}^\star = \mathbf{0}$, and a fixed point $\mathbf{x}$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{x}$. Thus, fixed points become the right-eigenvectors of $\mathbf{A}$ corresponding to eigenvalue 1 (if eigenvalue 1 exists). The special case of zero input is, in fact, considered in [27, 28] with the additional assumption of $\mathbf{A}$ being a normal matrix. In this study we allow $\mathbf{A}$ to be an arbitrary matrix. We will focus on this special case later in Section 3 when considering autonomous spectral clustering of directed graphs.

2.2. Convergence in the Mean-Squared Sense

When the matrix $\mathbf{A}$ has an eigenvalue equal to 1 the fixed points of the update scheme are not unique, and they form an affine subspace of dimension m as described in (9). As a result, the random vector $\mathbf{x}_k$ evolving according to the random asynchronous model (4) does not necessarily converge to the point $\mathbf{x}^\star$; rather, the limit of the random vector $\mathbf{x}_k$ (as k goes to infinity) is a random vector defined over the affine subspace in (9). We refer to [27, Figure 1] for a visual representation of this behavior.

In order to quantify the convergence of the random vector $\mathbf{x}_k$, we define the error vector as the residual from the orthogonal projection of $\mathbf{x}_k$ onto the affine subspace of fixed points. More precisely,

$$\mathbf{r}_k = \mathbf{Q}\,\left(\mathbf{x}_k - \mathbf{x}^\star\right), \tag{11}$$

which is a random vector due to the randomness of $\mathbf{x}_k$. Then, the convergence of $\mathbf{x}_k$ to the affine subspace of fixed points is equivalent

Authorized licensed use limited to: CALIFORNIA INSTITUTE OF TECHNOLOGY. Downloaded on July 02,2021 at 15:57:10 UTC from IEEE Xplore. Restrictions apply.

to the convergence of $\mathbf{r}_k$ to zero. It is important to note that the projection matrix $\mathbf{Q}$ has rank $N-m$, so the convergence of $\mathbf{r}_k$ to zero does not imply the convergence of $\mathbf{x}_k$ to the point $\mathbf{x}^\star$ in general.

Due to its randomness, convergence properties of $\mathbf{r}_k$ depend on how exactly the convergence is defined in the statistical setting. Similar to the previous works [27–32], this study will consider the convergence in the mean-squared sense. More precisely, we will focus on the correlation matrix of the error term:

$$\mathbf{R}_k = \mathbb{E}\big[\mathbf{r}_k\, \mathbf{r}_k^{\rm H}\big], \tag{12}$$

which is a deterministic matrix possibly with complex entries (the matrix $\mathbf{A}$ is never assumed to be real valued). Furthermore, the convergence of $\mathbf{R}_k$ to zero (as k goes to infinity) implies the convergence of the random vector $\mathbf{r}_k$ to zero in the mean-squared sense.

When there is noise in the system, i.e., $\mathbf{\Gamma} \neq \mathbf{0}$, the error correlation matrix does not converge to zero. Depending on the stability of the randomized asynchronous updates the error correlation matrix either increases unboundedly, or it reaches an error floor (the limit of $\mathbf{R}_k$) depending on the amount of noise. The following theorem (whose proof is provided in the supplementary document in [33]) presents the precise condition for the stability of the randomized asynchronous updates together with the exact value of the error floor:

Theorem 1. *The limit of the vectorized error correlation matrix is given as follows irrespective of the initial point* $\mathbf{x}_0$*:*

$$\lim_{k\to\infty} \operatorname{vec}(\mathbf{R}_k) = \big(\mathbf{I} - \mathbf{\Theta}\,\mathbf{S}\big)^{-1}\, \mathbf{\Theta}\, \boldsymbol{\gamma} \tag{13}$$

if and only if the following holds true:

$$\rho(\mathbf{\Theta}\,\mathbf{S}) < 1, \tag{14}$$

where the matrices $\mathbf{S} \in \mathbb{C}^{N^2\times N^2}$ *and* $\mathbf{\Theta} \in \mathbb{C}^{N^2\times N^2}$ *are as follows:*

$$\mathbf{S} = \bar{\mathbf{A}}^* \otimes \bar{\mathbf{A}} + \Big((\mathbf{I}-\mathbf{P})\otimes\mathbf{P}\Big)\,\mathbf{J}\,\Big((\mathbf{A}^*-\mathbf{I})\otimes(\mathbf{A}-\mathbf{I})\Big), \tag{15}$$

$$\bar{\mathbf{A}} = \mathbf{I} + \mathbf{P}\,(\mathbf{A}-\mathbf{I}), \qquad \mathbf{\Theta} = \mathbf{Q}^* \otimes \mathbf{Q}, \tag{16}$$

and the vector $\boldsymbol{\gamma} \in \mathbb{C}^{N^2}$ *is as follows:*

$$\boldsymbol{\gamma} = \operatorname{vec}\Big(\mathbf{P}\,\mathbf{\Gamma}\,\mathbf{P} + \operatorname{diag}\big(\mathbf{P}\,\mathbf{\Gamma}\,(\mathbf{I}-\mathbf{P})\big)\Big). \tag{17}$$

It is important to note that the stability condition given in (14) depends on both $\mathbf{A}$ and $\mathbf{P}$. So, a random asynchronous system may be stable for some set of probabilities, but it may not be stable for some other set of probabilities. We also note that the stability of the matrix $\mathbf{A}$ is *not necessary* to satisfy the condition (14) in general. That is, a system may be unstable in the synchronous world, but it may get stable when the state variables are updated randomly and asynchronously. This remarkable property has been observed (and proven) in earlier studies as well [27, 28, 31].

We also note that the main difference between the result of [27] and Theorem 1 of this study is the additional assumptions in [27] on the matrix $\mathbf{A}$ and the probabilities, and their corresponding consequences. More precisely, [27] assumes $\mathbf{A}$ to be a normal matrix (unitarily diagonalizable) and the update probabilities to be equal, i.e., $\mathbf{P} = p\,\mathbf{I}$ for some p, then it shows that the convergence region for the eigenvalues can get only larger with the randomized asynchronicity. See [27, Figure 2] for a visual representation. As a result, randomized asynchronous updates are guaranteed to converge in the mean-squared sense whenever the synchronous counter-part converges [27, Corollary 3]. However, if the indices are updated with non-equal probabilities, *or* the matrix $\mathbf{A}$ is not unitarily diagonalizable, then random asynchronous updates may not convergence while its synchronous counter-part does. Theorem 1 of this paper considers the case of *arbitrary* $\mathbf{A}$ *and arbitrary probabilities*, and it presents the necessary and sufficient condition for the convergence.

It is also worth noting that the matrix $\mathbf{S}$ given in (15) is first defined in [31], where it is shown that the stability of $\mathbf{S}$ determines the mean-squared stability of the randomized recursions when $\mathbf{A}$ does not have an eigenvalue 1. When the matrix $\mathbf{A}$ has an eigenvalue equal to 1, the following can be verified:

$$\mathbf{S}\,\left(\mathbf{V}_1^* \otimes \mathbf{V}_1\right) = \mathbf{V}_1^* \otimes \mathbf{V}_1. \tag{18}$$

So, the matrix $\mathbf{S}$ is not stable by construction. However, the stability of the recursions in the presence of eigenvalue 1 is determined by the matrix $\mathbf{\Theta}\,\mathbf{S}$, which can still be stable depending on the update probabilities. It is also worth noting that when eigenvalue 1 does not exist we get $\mathbf{\Theta} = \mathbf{I}$, and Theorem 1 reduces to the result in [31].

As a final remark, we observe that the error floor given in (13) depends on the matrices $\mathbf{A}$ and $\mathbf{P}$ as well as the noise covariance matrix $\mathbf{\Gamma}$. Therefore, the noise statistics (the matrix $\mathbf{\Gamma}$) affects the error floor, but it does not affect the stability of the randomized asynchronous system. Furthermore, the error floor depends linearly on the noise covariance matrix $\mathbf{\Gamma}$, whereas the dependence on $\mathbf{A}$ and $\mathbf{P}$ is non-linear. In the noise-free case, i.e., $\mathbf{\Gamma} = \mathbf{0}$, the error floor in (13) becomes zero. Thus, condition in (14) is both necessary and sufficient for the mean-squared convergence of the error vector $\mathbf{r}_k$.

3. SPECTRAL CLUSTERING FOR DIRECTED GRAPHS

As an application of the randomized asynchronous recursions in the context of graph signal processing, this section will consider spectral clustering for directed graphs. We note that this application is considered also in [27], where the underlying graph was assumed to be undirected. The purpose of this section is to show that an autonomous clustering can be achieved even in the case of directed graphs.

Spectral clustering is well-understood for undirected graphs [34], and there are various different ways to extend it to the case of directed graphs [35–38]. In this study, we will consider the following Laplacian matrix of a given *directed* graph:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{19}$$

where $\mathbf{D}$ is the diagonal matrix with i^{th} entry being $d_i = \sum_j A_{i,j}$, and $\mathbf{A}$ denotes the adjacency matrix of the underlying graph. The matrix $\mathbf{A}$ is not symmetric, it may even be non-diagonalizable.

When the underlying graph is directed, the Laplacian matrix $\mathbf{L}$ is no longer symmetric, and it has complex eigenvalues in general. Nevertheless, the constant vector is still an eigenvector of $\mathbf{L}$ corresponding to eigenvalue zero, and the eigenvector that corresponds to the smallest (in magnitude) non-zero eigenvalue can be used to cluster the graph into two partitions [39]. This is visualized in Figure 1.

We note that studying the second dominant eigenvector of the graph Laplacian in (19) is not necessarily the best way for obtaining spectral clustering. Some other graph operators considered in [35–38] may perform better in specific examples, and we are not favoring the use of $\mathbf{L}$ in general. The matrix $\mathbf{L}$ in (19) is considered as an example to demonstrate the use of randomized asynchronous updates for spectral clustering.

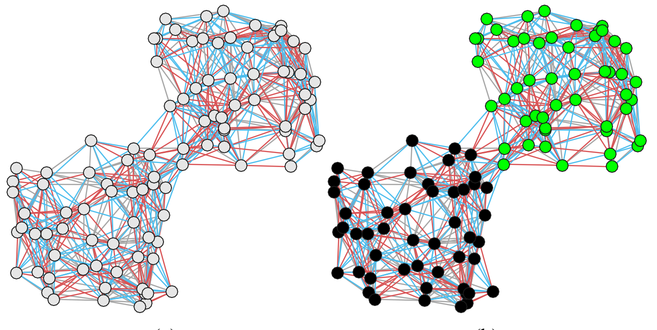

Fig. 1: (a) A directed graph (with binary edge weights) on $N = 100$ nodes with 2 clusters. The edges directed from left to right are colored in blue, and the edges directed from right to left are colored in red. Bidirectional edges are colored in gray. (b) The result of the spectral clustering based on the sign pattern of the eigenvector of $\mathbf{L}$.

Authorized licensed use limited to: CALIFORNIA INSTITUTE OF TECHNOLOGY. Downloaded on July 02,2021 at 15:57:10 UTC from IEEE Xplore. Restrictions apply.

3.1. Random Asynchronous Recursions on Polynomials

Without loss of generality, we can assume that the eigenvalues of the matrix $\mathbf{L}$ are ordered as follows: $0 = \lambda_1 \leqslant |\lambda_2| \leqslant \cdots \leqslant |\lambda_N|$. Then, the spectral clustering is based on the eigenvector $\mathbf{v}_2$ (corresponding to eigenvalue λ_2) of $\mathbf{L}$, which can be computed via randomized asynchronous recursions as follows. We first construct a polynomial $h(\cdot)$ such that $h(\lambda_2) = 1$ is satisfied, in which case the vector $\mathbf{v}_2$ becomes an eigenvector of the matrix polynomial $h(\mathbf{L})$ corresponding to eigenvalue 1. More precisely,

$$\mathbf{L}\,\mathbf{v}_2 = \lambda_2\,\mathbf{v}_2 \qquad \Longrightarrow \qquad h(\mathbf{L})\,\mathbf{v}_2 = \mathbf{v}_2. \tag{20}$$

Then, we use the randomized updates in (4) with the transition matrix being $h(\mathbf{L})$ and the input signal being $\mathbf{u} = \mathbf{0}$. As a result, the iterant $\mathbf{x}_k$ in (4) converges to an eigenvector of $h(\mathbf{L})$ corresponding to eigenvalue 1, which happens to be the vector $\mathbf{v}_2$. This approach is considered also in [27], but the graph there was assumed undirected.

The importance of polynomials follows from their distributed implementation in the graph setting [3,4,8], and polynomials are still practical when the communications are assumed to be asynchronous. However, it should be noted that the convergence of the vector $\mathbf{x}_k$ to $\mathbf{v}_2$ is achieved with random updates only when the polynomial $h(\cdot)$ is selected in such a way that the matrix $h(\mathbf{L})$ and the update probabilities $\mathbf{P}$ satisfy the stability condition (14) of Theorem 1. An approach for designing such polynomials will be considered next.

3.2. Design of the Polynomial Filters

Although the condition (14) is both necessary and sufficient for the convergence of the randomized updates, the condition itself is difficult to manipulate since $\mathbf{S}$ defined in (15) is a not a symmetric matrix even when the underlying graph is undirected. Instead of working with (14) directly, we will follow the approach considered in [27] and focus on the convergence of the synchronous case. More precisely, we select a polynomial $h(\cdot)$ that satisfies the following:

$$h(\lambda_2) = 1, \qquad\qquad |h(\lambda_i)| < 1 \quad \forall\, i \neq 2. \tag{21}$$

When the underlying graph is undirected and nodes are updated with equal probabilities, the specification in (21) is sufficient to ensure the convergence even with asynchronous updates [27]. However, when the underlying graph is directed, (21) does not ensure the convergence of the randomized updates in general. Nevertheless, we will keep using (21) because of the following two reasons: 1) In practice, polynomials designed according to (21) can provide convergence even in the random asynchronous case. (See Section 3.3) 2) The search for a polynomial satisfying (21) can be cast as the following optimization problem, which can be solved numerically:

$$\max_{c,\ \mathbf{h}} \quad c \qquad \text{s.t.} \qquad \begin{array}{c} \boldsymbol{\phi}_2\,\mathbf{h} = 1, \\ \left|\bar{\boldsymbol{\Phi}}_2\,\mathbf{h}\right| \leqslant (1-c)\,\mathbb{1}, \end{array} \tag{22}$$

where $\mathbf{h}$ is a row vector of length $L{+}1$ corresponding to the coefficients of the polynomial $h(\cdot)$ of order L. Here $\boldsymbol{\Phi}$ is constructed with the eigenvalues of $\mathbf{L}$ as follows:

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \cdots & \lambda_1^L \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \lambda_N & \lambda_N^2 & \cdots & \lambda_N^L \end{bmatrix} \in \mathbb{C}^{N\times(L+1)}, \tag{23}$$

and $\boldsymbol{\phi}_2$ denotes the second row of the Vandermonde matrix $\boldsymbol{\Phi}$, and $\bar{\boldsymbol{\Phi}}_2$ denotes the remaining rows (all except the second) of $\boldsymbol{\Phi}$.

When the optimal solution of (22) satisfies $c^\star > 0$, the corresponding polynomial $h(\cdot)$ satisfies the specification in (21). In the case of undirected graphs, [27] showed that second order polynomials, i.e., $L = 2$, are sufficient to satisfy (21). However, this result is no longer valid in the case of directed graphs. Nevertheless, a quadratic polynomial with the following coefficients can be shown to satisfy (21) for the graph example in Figure 1(a):

$$h_0 \approx 0.9972, \qquad h_1 \approx 0.0127, \qquad h_2 \approx -0.0063. \tag{24}$$

Then, we can use Algorithm 1 in [27], which implements the asynchronous updates running on a second order polynomial, in order to obtain the spectral clustering in Figure 1(b) in an autonomous way.

3.3. Simulation Results

In this section, we numerically demonstrate the convergence of the randomized asynchronous updates for the spectral clustering of the directed graph visualized in Figure 1(a). For this purpose, we consider the updates running on the matrix polynomial $h(\mathbf{L})$, where the Laplacian matrix $\mathbf{L}$ is constructed as in (19), and the polynomial coefficients are selected as in (24). For simplicity, we consider the case of nodes being updated with equal probabilities, i.e., $\mathbf{P} = p\,\mathbf{I}$ for the values of $p \in \{0.1,\ 0.5,\ 0.9,\ 1\}$, and set $\boldsymbol{\Gamma} = 10^{-8}\,\mathbf{I}$.

It is clear that $h(\mathbf{L})$ ensures the convergence in the synchronous case since its coefficients are designed according to (21). One can also verify that the stability condition in (14) is met in the asynchronous case with the tested probabilities. Figure 2(a) verifies this convergence numerically as well. We also note that the convergence is not exact due to the presence of the input noise, and the expected norm of the error vector reaches an error floor as given in (13).

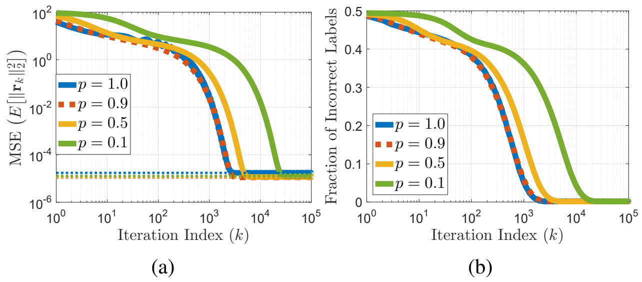

Fig. 2: (a) Mean squared ℓ_2-norm of the error, i.e., $\mathrm{tr}(\mathbf{R}_k)$. Dotted lines correspond to the theoretical error floor given by (13). (b) Averaged error in clustering based on the sign pattern of the iterant $\mathbf{x}_k$. Results are obtained by averaging over 10^5 independent realizations.

Since the spectral clustering in Figure 1(b) uses the sign pattern of the eigenvector, the sign pattern of the most recent value of $\mathbf{x}_k$ can be used to estimate the clustering at iteration k. That is, the node i can determine the cluster it belongs to on its own simply by the sign of the value of $(\mathbf{x}_k)_i$ it holds. As $\mathbf{x}_k$ converges to the eigenvector, estimations get better and converge to the result of the spectral clustering. This behavior is demonstrated in Figure 2(b), which shows that the exact convergence of $\mathbf{x}_k$ to $\mathbf{v}_2$ is not necessary to obtain the exact spectral clustering. Figure 2(b) shows that the sign pattern of $\mathbf{x}_k$ provides the correct clustering on average when the error vector has squared ℓ_2-norm less than approximately 10^{-4}.

4. CONCLUDING REMARKS & FUTURE DIRECTIONS

In this study we considered the convergence behavior of the node-asynchronous updates where the underlying graph is allowed to have directed edges possibly with a non-diagonalizable adjacency matrix. The graph signal has a stochastic behavior due to the random behavior of the nodes. We presented the necessary and sufficient condition that ensures the mean-squared convergence of the graph signal to a fixed point of the update scheme under the assumption that the graph operator has an eigenvalue equal to 1 and the input signal is selected such that fixed points exist. We also considered the use of random asynchronous updates in polynomial filtering in order to compute the eigenvectors of the underlying graph operator, which allowed us to obtain spectral clustering even in the case of directed networks.

In future we will analyze the relation between the presented mean-squared stability condition and the graph operator as well as the node update probabilities. We will consider the optimal selection of the node update probabilities that provide the highest rate of convergence. We will also consider different approaches for the design of polynomial filters that ensure the convergence of the updates.

Authorized licensed use limited to: CALIFORNIA INSTITUTE OF TECHNOLOGY. Downloaded on July 02,2021 at 15:57:10 UTC from IEEE Xplore. Restrictions apply.

5. REFERENCES

[1] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, April 2013.

[2] ——, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, June 2014.

[3] 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.

[4] 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, Sept. 2014.

[5] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[6] M. E. J. Newman, *Networks: An Introduction*. Oxford Uni. Press, 2010.

[7] M. Jackson, *Social and Economic Networks*. Princeton Uni. Press, 2008.

[8] 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.

[9] ——, "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.

[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, June 2012.

[11] ——, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4673–4685, Oct. 2013.

[12] A. Anis, A. Gadde, and A. Ortega, "Efficient sampling set selection for bandlimited graph signals using graph spectral proxies," *IEEE Trans. Signal Process.*, vol. 64, no. 14, pp. 3775–3789, July 2016.

[13] 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.

[14] A. Agaskar and Y. M. Lu, "A spectral graph uncertainty principle," *IEEE Trans. on Inf. Theory*, vol. 59, no. 7, pp. 4338–4356, July 2013.

[15] M. Tsitsvero, S. Barbarossa, and P. D. Lorenzo, "Signals on graphs: Uncertainty principle and sampling," *IEEE Trans. Signal Process.*, vol. 64, no. 18, pp. 4845–4860, Sept. 2016.

[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. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *International Conference on Distributed Computing in Sensor Systems and Workshops*, June 2011, pp. 1–8.

[18] D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard, "Distributed signal processing via chebyshev polynomial approximation," *IEEE Trans. on Sig. and Inf. Process. Net.*, vol. 4, no. 4, pp. 736–751, Dec. 2018.

[19] S. Safavi and U. A. Khan, "Revisiting finite-time distributed algorithms via successive nulling of eigenvalues," *IEEE Sig. Process. Letters*, vol. 22, no. 1, pp. 54–57, Jan. 2015.

[20] A. Sandryhaila, S. Kar, and J. M. F. Moura, "Finite-time distributed consensus through graph filters," in *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*, May 2014, pp. 1080–1084.

[21] S. Segarra, A. G. Marques, and A. Ribeiro, "Distributed implementation of linear network operators using graph filters," in *Allerton Conference on Communication, Control, and Computing*, Sept. 2015, pp. 1406–1413.

[22] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, Aug. 2017.

[23] X. Shi, H. Feng, M. Zhai, T. Yang, and B. Hu, "Infinite impulse response graph filters in wireless sensor networks," *IEEE Sig. Process. Letters*, vol. 22, no. 8, pp. 1113–1117, Aug. 2015.

[24] E. Isufi, A. Loukas, A. Simonetto, and G. Leus, "Autoregressive moving average graph filtering," *IEEE Trans. on Sig. Process.*, vol. 65, no. 2, pp. 274–288, Jan. 2017.

[25] ——, "Filtering random graph processes over random time-varying graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4406–4421, Aug. 2017.

[26] A. Loukas, A. Simonetto, and G. Leus, "Distributed autoregressive moving average graph filters," *IEEE Sig. Process. Letters*, vol. 22, no. 11, pp. 1931–1935, Nov. 2015.

[27] O. Teke and P. P. Vaidyanathan, "Random node-asynchronous updates on graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 11, pp. 2794–2809, June 2019.

[28] ——, "The random component-wise power method," in *Proc. SPIE, Wavelets and Sparsity XVIII*, vol. 11138, Sep. 2019.

[29] ——, "Node-asynchronous implementation of rational filters on graphs," in *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*, May 2019, pp. 7530–7534.

[30] ——, "IIR filtering on graphs with random node-asynchronous updates," *Submitted to IEEE TSP*, June 2019.

[31] ——, "Randomized asynchronous recursions with a sinusoidal input," in *Asilomar Conf. on Signals, Systems and Computers*, Oct. 2019, to appear.

[32] ——, "Asynchronous nonlinear updates on graphs," in *Asilomar Conf. on Signals, Systems and Computers*, Oct. 2018, pp. 998–1002.

[33] ——. (2019) Supplementary document for node-asynchronous spectral clustering on directed graphs. [Online]. Available: http://systems.caltech.edu/dsp/students/oteke/files/icassp2020.pdf

[34] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *NIPS 14*, 2002, pp. 849–856.

[35] D. Zhou, B. Schölkopf, and T. Hofmann, "Semi-supervised learning on directed graphs," in *Proc. Conf. Neural Information Processing Systems*, 2004, pp. 1633–1640.

[36] D. Zhou, J. Huang, and B. Schölkopf, "Learning from labeled and unlabeled data on a directed graph," in *Proceedings of the 22Nd International Conference on Machine Learning*, 2005, pp. 1036–1043.

[37] W. Pentney and M. Meila, "Spectral clustering of biological sequence data," in *Proceedings of the 20th National Conference on Artificial Intelligence*, 2005, pp. 845–850.

[38] M. Meila and W. Pentney, "Clustering by weighted cuts in directed graphs," in *Proceedings of the SIAM International Conference on Data Mining*, 2007, pp. 135–144.

[39] C. W. Wu, "Algebraic connectivity of directed graphs," *Linear and Multilinear Algebra*, vol. 53, no. 3, pp. 203–223, 2005.

Authorized licensed use limited to: CALIFORNIA INSTITUTE OF TECHNOLOGY. Downloaded on July 02,2021 at 15:57:10 UTC from IEEE Xplore. Restrictions apply.