Asynchronous Nonlinear Updates on Graphs

Oguzhan Teke and P. P. Vaidyanathan
Department of Electrical Engineering
California Institute of Technology
oteke@caltech.edu, ppvnath@systems.caltech.edu

***Abstract*—The notion of graph shift, introduced recently in graph signal processing, extends many classical signal processing techniques to graphs. Its practical importance follows from its localization: a single graph shift requires nodes to communicate only with their neighbors. However, communications should happen simultaneously, which requires a synchronization over the graph. In order to overcome this restriction, recent studies consider a random asynchronous variant of the graph shift, which is also suitable for autonomous networks. A graph signal under this randomized scheme is shown to converge (under mild conditions) to an eigenvector of the eigenvalue 1 of the operator even if the operator has other eigenvalues with magnitudes larger than unity. If the eigenvalue 1 does not exist, the operator can be easily normalized in theory. However, in practice, the normalization requires one to know the (dominant) eigenvalues, which may not be possible to obtain in large autonomous networks. To eliminate this limitation, this study considers the use of a nonlinearity in the updates making the scheme similar in spirit to the Hopfield neural network model. Our simulation results show that a graph signal still approaches the eigenvector of the dominant eigenvalue although the convergence is not exact. Nevertheless, approximation is sufficient to accomplish certain tasks including autonomous clustering.**

***Index Terms*—Autonomous networks, nonlinear filters, randomized iterations, autonomous clustering.**

I. Introduction

In recent information era the data in general is defined with respect to an irregular domain, in which case the irregularity of the domain is represented with a graph. Such data models are very broad and can be found in a variety of different contexts such as social, economic, and biological networks, among others [1], [2]. The recent advancements in [3]–[6] 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). 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 [7] (and references therein).

In this study we will consider signals over a network of independent agents in which there is no centralized control mechanism. Agents receive information from their neighbors at random time instances repeatedly and independently from each other. Then, they update their own value as a linear combination of the received data, which is assumed to be described precisely by the graph operator. The notion of the graph shift, introduced in [6], is not directly applicable to the autonomous network model considered here since a graph shift requires all the agents to communicate at the same time instance. Such a synchronization between the agents is not possible in an autonomous setting. In order to overcome this limitation we consider randomized asynchronous updates, which are similar to the well-known power iteration (or, the graph shift) except that only a random subset of indices are updated in each iteration.

Studies [8], [9] showed that the signal under such random asynchronous model converges to an eigenvector of the graph operator corresponding to eigenvalue 1 in the mean-squared sense. When the operator does not have the eigenvalue 1, the study in [9] considered the use of polynomials, where the polynomial is designed in such a way that it maps an arbitrary eigenvalue to 1. Despite its flexibility, main limitation of the use of polynomials is that the design of the polynomial requires to know the eigenvalues of the graph operator, which is not feasible in an autonomous network considered here.

In this study, we will consider a second order polynomial (which is shown to be sufficient for the convergence) that can be constructed using inexact values of the eigenvalues. However, such a construction causes the signal to diverge. In order to compensate for this, we introduce a saturation into the update model, under which the signal converges to a point that is approximately equivalent to the eigenvector (of the desired eigenvalue). The approximation is observed to be sufficient to accomplish certain tasks including autonomous clustering.

In the following, we first present the assumptions made for the graph operator. Then, Sections II and III provide an overview of the results presented in [9]. In particular, Section II presents the randomized asynchronous state recursions and shows its convergence behavior (Theorem 1 and Corollary 1). Section III describes the use of polynomial filters in order to obtain a convergence to an arbitrary eigenvector and shows the sufficiency of second order polynomials (Theorem 2). Main contributions of this study are presented in Sections IV and V. In Section IV we introduce a non-linearity into the randomized asynchronous update scheme, which allows us to design polynomial filters using inexact values of the eigenvalues of the graph. In Section V we implement the non-linear update model to obtain the spectral clustering of the underlying network.

A. Preliminaries and Assumptions

We will assume that $\boldsymbol{A} \in \mathcal{C}^{N\times N}$ is an operator on the graph with N nodes. We consider $\boldsymbol{A}$ 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 always assume that $\boldsymbol{A}$ is a normal matrix (equivalently, a unitary-diagonalizable matrix), i.e., $\boldsymbol{A}\boldsymbol{A}^* = \boldsymbol{A}^*\boldsymbol{A}$, where $\boldsymbol{A}^*$ denotes the transpose conjugate of $\boldsymbol{A}$. We note that Hermitian matrices are necessarily normal, but not vice-versa.

II. Asynchronous Power Iteration

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

$$\boldsymbol{x}_k = \boldsymbol{A}\,\boldsymbol{x}_{k\text{-}1}, \qquad \text{so that} \qquad \boldsymbol{x}_k = \boldsymbol{A}^k\,\boldsymbol{x}_0, \tag{1}$$

where the updates here are considered without normalizing the signal at each iteration. Normalization is avoided here

This work was supported in parts by the ONR grants N00014-17-1-2732 and N00014-18-1-2390, the NSF grant CCF-1712633, and the Electrical Engineering Carver Mead Research Seed Fund of the California Institute of Technology.

intentionally to preserve the local nature of the updates as will be elaborated next.

In the context of graph signal processing, the matrix $\boldsymbol{A}$ is assumed to be a local graph operator (shift matrix) and the signal $\boldsymbol{A}\,\boldsymbol{x}$ is referred to as the shifted version of $\boldsymbol{x}$ on the graph [6]. From this perspective $\boldsymbol{x}_k$ in (1) is the graph shifted version of $\boldsymbol{x}_{k\text{-}1}$. Since $\boldsymbol{A}$ 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,

$$(\boldsymbol{x}_k)_i = \sum_{j\in\mathcal{N}_{\text{in}}(i)} A_{i,j}\ (\boldsymbol{x}_{k\text{-}1})_j \qquad\qquad \forall\, i. \tag{2}$$

where $\mathcal{N}_{\text{in}}(i)$ denotes the incoming neighbors of the node i. 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 (1).

Although a "graph shift" can be performed locally, the model in (1) 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 $\mathcal{T}_k$, are updated simultaneously and the remaining ones stay unchanged. More precisely, given an update set $\mathcal{T}_k$ we consider the following *asynchronous* (coordinate-wise) power iteration:

$$\big(\boldsymbol{x}_k\big)_i = \begin{cases} \big(\boldsymbol{A}\,\boldsymbol{x}_{k\text{-}1}\big)_i, & i\in\mathcal{T}_k,\\ \big(\boldsymbol{x}_{k\text{-}1}\big)_i, & i\notin\mathcal{T}_k, \end{cases} \tag{3}$$

where $\boldsymbol{x}_{k\text{-}1}$ is the vector before update, and $\boldsymbol{x}_k$ is the vector after the update. In words, this update computes the multiplication $\boldsymbol{A}\,\boldsymbol{x}_{k\text{-}1}$, but it only updates the values of the elements indexed by the set $\mathcal{T}_k$, and keeps the remaining elements the same.

In this section $\boldsymbol{A}$ will be treated as a generic matrix without considering specific relations to graphs. When the model in (3) is implemented on a graph (i.e., $\boldsymbol{A}$ is a graph operator), only the nodes in the update set $\mathcal{T}_k$ need to be synchronized. If the update set is selected as $\mathcal{T}_k=\{1,\cdots,N\}$ in every iteration, the asynchronous update in (3) reduces to the classical synchronous update (graph shift) in (1). On the other extreme, if a single node is updated, $|\mathcal{T}_k|=1$, then *no synchronization is required at all* and the nodes are allowed to behave autonomously.

We will assume that the size of the subsets has a predetermined fixed value, that is, $|\mathcal{T}_k|=t$. However, the content of $\mathcal{T}_k$ is assumed to be selected uniformly random among all possible $\binom{N}{t}$ subsets of size t at every iteration of (3). This assumption ensures that there is no bias toward any node(s) in the network. More generally, the size of $\mathcal{T}_k$ need not be the same at every iteration. The case of updating different number of nodes in different iterations is considered in [9].

We will see later in Section II-A that the following definition is very useful in our quantitative analysis:

$$\delta_T = \frac{N-t}{N-1}, \qquad \text{and} \qquad 0\leqslant\delta_T\leqslant 1. \tag{4}$$

It can be seen that $\delta_T=0$ 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.

A. Convergence in Mean-Squared Sense

We would like to first note that the fixed points of the update scheme in (3) correspond to the eigenspace of eigenvalue 1 (unit eigenvalue) of the matrix $\boldsymbol{A}$. As discussed in [9], if the random signal $\boldsymbol{x}_k$ ever converges to a non-zero point $\boldsymbol{x}$ through iterations, then $\boldsymbol{x}$ should lie in the eigenspace of the unit eigenvalue. Thus, it is necessary for $\boldsymbol{A}$ to have eigenvalue 1 in order to establish a convergence to a non-zero point. For this reason, we will assume that $\boldsymbol{A}$ has a unit eigenvalue with multiplicity $M\geqslant 1$. Without loss of generality we will order the eigenvalues of $\boldsymbol{A}$ such that $\lambda_i\neq 1$ for $1\leqslant i\leqslant N\text{-}M$. Notice that non-unit eigenvalues are allowed to be complex.

Then, the eigenvalue decomposition of $\boldsymbol{A}$ can be written as:

$$\boldsymbol{A} = [\boldsymbol{U}\ \ \boldsymbol{V}_1]\,\mathrm{diag}\left([\lambda_1\ \cdots\ \lambda_{N\text{-}M}\ 1\ \cdots\ 1]\right)[\boldsymbol{U}\ \ \boldsymbol{V}_1]^*, \tag{5}$$

where $\boldsymbol{V}_1\in\mathcal{C}^{N\times M}$ is an orthonormal basis for the eigenspace of the unit eigenvalue, and $\boldsymbol{U}\in\mathcal{C}^{N\times(N\text{-}M)}$ is such that $\boldsymbol{U}^*\,\boldsymbol{V}_1=\boldsymbol{0}$. We now define the following quantity:

$$\rho = \big\|\boldsymbol{U}^*\ \mathrm{diag}(\boldsymbol{U}\,\boldsymbol{U}^*)\,\boldsymbol{U}\big\|_2, \tag{6}$$

which will play a crucial role in the analysis of convergence. Notice that ρ does not depend on the particular selection of the basis matrix $\boldsymbol{U}$. Just the column space of $\boldsymbol{U}$ determines its values. Furthermore, one can verify that the parameter ρ satisfies the following inequality:

$$0<\rho\leqslant 1. \tag{7}$$

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

$$\boldsymbol{r}_k = \boldsymbol{x}_k - \boldsymbol{V}_1\,\boldsymbol{V}_1^*\,\boldsymbol{x}_k. \tag{8}$$

Then, the convergence of $\boldsymbol{x}_k$ to an eigenvector of the unit eigenvalue is equivalent to the convergence of $\boldsymbol{r}_k$ to zero. The following theorem, whose proof is presented in [9], provides bounds for $\boldsymbol{r}_k$ as follows:

Theorem 1. *The expected squared ℓ_2-norm of the residual at the k^{th} iteration is bounded as follows:*

$$\mathbb{E}\big[\,\|\boldsymbol{r}_k\|_2^2\,\big]\ \leqslant\ \Psi^k\,\|\boldsymbol{r}_0\|_2^2, \tag{9}$$

where

$$\Psi = \max_{1\leqslant i\leqslant N\text{-}M}\ 1+\frac{\mu_T}{N}\Big(|\lambda_i|^2-1+\delta_T\,(\rho-1)\,|\lambda_i-1|^2\Big). \tag{10}$$

As an immediate consequence of Theorem 1, the following corollary presents a sufficiency condition under which the randomized asynchronous iterations are guaranteed to converge in the mean-squared sense.

Corollary 1. *Assume that all non-unit eigenvalues of $\boldsymbol{A}$ satisfy the following condition:*

$$\left|\lambda-\frac{\alpha}{\alpha+1}\right|<\frac{1}{\alpha+1}, \qquad \textit{where} \qquad \alpha=\delta_T\,(\rho-1). \tag{11}$$

Then,

$$\lim_{k\to\infty}\quad \mathbb{E}\big[\,\|\boldsymbol{r}_k\|_2^2\,\big]=0. \tag{12}$$

Proof: From (10) it is clear that $\Psi<1$ if and only if

$$|\lambda|^2-1+\alpha\,|\lambda-1|^2<0, \tag{13}$$

for all non-unit eigenvalues λ. The inequality in (13) can be equivalently written as in (11). Since it implies that $\Psi<1$, Theorem 1 guarantees the convergence of $\mathbb{E}[\|\boldsymbol{r}_k\|_2^2]$ to zero as the number of updates, k, goes to infinity. ∎

The condition given by (11) describes an open disk on the complex plane centered at $\alpha/(\alpha+1)$ with radius $1/(\alpha+1)$. This region is visualized in Figure 1. Due to the inequities in (4) and (7), the parameter α always satisfies $-1<\alpha\leqslant 0$.

Thus, one can easily verify the following:

$$-\infty < \frac{\alpha - 1}{\alpha + 1} \leqslant -1, \tag{14}$$

which shows that the region defined by (11) is never smaller than the unit disk (See Figure 1). In fact, the convergence region is the smallest (and corresponds to the open unit disk) when $\alpha = 0$, and the region gets *larger* as α gets smaller. Thus, the convergence region can be maximized by minimizing the value of α as we shall explain next.

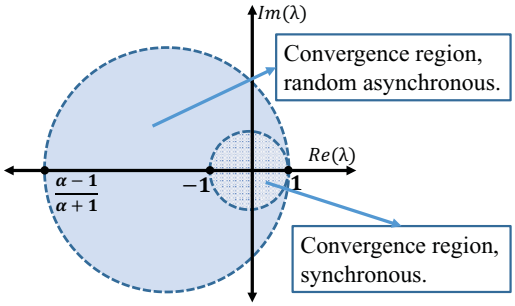


Fig. 1. The convergence region given in (11) for the eigenvalues such that random asynchronous updates are guaranteed to converge.

Given the matrix $\boldsymbol{A}$, the eigenspace parameter ρ has a fixed value (depending on $\boldsymbol{A}$) satisfying $\rho - 1 \leqslant 0$ due to (7). Therefore, one can minimize the value of α by maximizing the value of δ_T. Thus, we conclude that the convergence region of the eigenvalues gets larger as the updates become *more asynchronous*, and the region is the smallest (and becomes the unit disk) when the updates are synchronous. This conclusion shows also that randomized asynchronous updates are guaranteed to converge whenever the synchronous updates converge, which is formally presented as follows:

Corollary 2. *If the synchronous updates on $\boldsymbol{A}$ converge, then*

$$\lim_{k \to \infty} \quad \mathbb{E}\big[\, \|\boldsymbol{r}_k\|_2^2 \,\big] = 0, \tag{15}$$

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

Proof: If the synchronous updates converge, then all non-unit eigenvalues of $\boldsymbol{A}$ satisfy $|\lambda| < 1$. Hence, they also satisfy (11) for any value of α. Therefore, Corollary 1 ensures the convergence of the updates irrespective of the value of δ_T. ■

It is important to note that the convergence characteristics of the randomized asynchronous updates depend not only on the eigenvalues of the operator $\boldsymbol{A}$, but depend also on the eigenspace properties of the operator, which is quantified by ρ defined in (6). As a result, asynchronous updates may have different convergence behavior on similar matrices (matrices having the same eigenvalues). We refer the interested readers to [9] for further discussion on the effect of the eigenspace geometry.

III. Use of Polynomial Filters

The previous section discussed that the operator $\boldsymbol{A}$ is required to have eigenvalue 1 in order for the random asynchronous updates to converge to a non-zero point since the fixed points of the update scheme in (3) correspond to the eigenspace of $\boldsymbol{A}$ of the unit eigenvalue. Under the assumption that $\boldsymbol{A}$ has the unit eigenvalue, Corollary 1 provided a condition under which random asynchronous updates are guaranteed to converge to an eigenvector of eigenvalue 1 in the mean-squared sense.

In a typical application, however, the operator $\boldsymbol{A}$ is unlikely to have the eigenvalue 1 exactly. Moreover, even if $\boldsymbol{A}$ has the unit eigenvalue, it may be of interest to find the eigenvectors corresponding to *non-unit* eigenvalues. In order to address these points, we consider the randomized asynchronous updates running on a polynomial of the given operator. More precisely, we consider the following model:

$$\big(\boldsymbol{x}_k\big)_i = \begin{cases} \big(h(\boldsymbol{A})\,\boldsymbol{x}_{k\text{-}1}\big)_i, & i \in \mathcal{T}_k, \\ \big(\boldsymbol{x}_{k\text{-}1}\big)_i, & i \notin \mathcal{T}_k, \end{cases} \tag{16}$$

where $h(\boldsymbol{A})$ is an L^{th} order polynomial of $\boldsymbol{A}$ as follows:

$$h(\boldsymbol{A}) = \sum_{n=0}^{L} h_n\, \boldsymbol{A}^n. \tag{17}$$

Polynomials of a graph operator are useful to consider because of the following two reasons. Firstly, $\boldsymbol{A}$ and $h(\boldsymbol{A})$ have the same eigenvectors. For an arbitrary eigenpair $(\boldsymbol{v}_j, \lambda_j)$ of $\boldsymbol{A}$, when the polynomial is selected to satisfy the following:

$$h(\lambda_j) = 1 \qquad \text{and} \qquad |h(\lambda_i)| < 1 \quad \forall\, \lambda_i \neq \lambda_j, \tag{18}$$

then $\boldsymbol{v}_j$ is an eigenvector of $h(\boldsymbol{A})$ with eigenvalue 1. Furthermore, Corollary 2 ensures that the signal $\boldsymbol{x}_k$ in (16) converges to $\boldsymbol{v}_j$ in the mean-squared sense. Secondly, polynomials are localized operators on graphs. That is, computation of $\big(h(\boldsymbol{A})\,\boldsymbol{x}\big)_i$ requires the node i to retrieve information only from its L-hop neighbors. If the polynomial is of low order (L has a small value), then $h(\boldsymbol{A})\,\boldsymbol{x}$ can be computed locally, which is crucial to the autonomous model we consider in this study.

In the following sections we will assume that eigenvalues of $\boldsymbol{A}$ are real valued, which is the case for undirected graphs, and assume that $h_n \in \mathbb{R}$. As discussed in [9], some of the following results do not extend to the complex case.

Although it is favorable to use low order filters to satisfy the condition in (18), the order L cannot be arbitrarily small. For example, the case of $L = 1$ is not sufficient to ensure the condition for an arbitrary eigenvalue λ_j. 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 (18).

Theorem 2. *Assume that the operator $\boldsymbol{A}$ has real eigenvalues $\lambda_i \in \mathcal{R}$. For a given target eigenvalue λ_j, the condition in* (18) *is satisfied by the following second order polynomial:*

$$h(\lambda) = 1 - 2\,\epsilon\,(\lambda - \lambda_j)^2 / s_j^2, \tag{19}$$

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

$$s_j \geqslant \max_{1 \leqslant i \leqslant N} |\lambda_i - \lambda_j|. \tag{20}$$

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,

$$1 - h(\lambda_i) = 2\,\epsilon\,(\lambda_i - \lambda_j)^2 / s_j^2 > 0, \tag{21}$$

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 (20). Therefore we have

$$h(\lambda_i) = 1 - 2\,\epsilon\,(\lambda_i - \lambda_j)^2 / s_j^2 \geqslant 1 - 2\epsilon > -1, \tag{22}$$

for all λ_i. ■

One should note that ϵ in (19) can be selected arbitrarily and tuned in order to increase the eigenvalue gap. In general, the polynomial constructed according to (19) is not expected to be optimal in terms of the rate of convergence. In fact, one can consider the problem of finding the optimal filter of an arbitrary order L, which is addressed in [9].

Although the polynomial in (19) ensures the convergence of the signal in (16) to an arbitrary eigenvector of $\boldsymbol{A}$, the construction of the polynomial itself requires further attention. Notice that the coefficients of (19) depend on the eigenvalue of the target eigenvector λ_j, and the parameter s_j. Assuming that λ_j is known, the parameter s_j can be selected using only bounds on the spectrum of the operator. More precisely, let λ_{upp} and λ_{low} be *arbitrary* upper and lower bounds on the spectrum of $\boldsymbol{A}$, respectively. That is,

$$\lambda_{\text{low}} \leqslant \lambda(\boldsymbol{A}) \leqslant \lambda_{\text{upp}}. \tag{23}$$

Then, the following selection of s_j satisfies (20):

$$s_j = \max\{\lambda_{\text{upp}} - \lambda_j,\ \ \lambda_j - \lambda_{\text{low}}\}. \tag{24}$$

It should be noted that there is no unique way of selecting a value for s_j. On the contrary, the construction in (19) requires the exact value of λ_j. In general, it is difficult to obtain the eigenvalues of the underlying graph in an autonomous setting. Thus, the polynomial required to ensure the convergence to an arbitrary eigenvector may not be constructed. In the following section, we will discuss how we can deal with inexact values of λ_j by introducing a non-linearity in the update scheme.

IV. Use of Nonlinearity

In this section, we will focus on the construction of a second order polynomial when the target eigenvalue λ_j is not known exactly. In this regard, we first assume that we are given an interval $[a\ \ b]$ to which *only* the eigenvalue λ_j belongs. More precisely, we assume the following:

$$\lambda_{j\text{-}1} < a \leqslant \lambda_j \leqslant b < \lambda_{j+1}, \tag{25}$$

where we consider only the distinct eigenvalues indexed in ascending order, and assume that eigenvalues are real. Then, we consider the following polynomial:

$$h(\lambda) = 1 - 2\epsilon \frac{(\lambda - a)(\lambda - b)}{\big(\max\{\lambda_{\text{upp}} - b,\ a - \lambda_{\text{low}}\} + (b-a)/2\big)^2} \tag{26}$$

for some ϵ in $0 < \epsilon < 1$. It should be noted that when the target eigenvalue λ_j is known exactly, we can take $a = b = \lambda_j$, in which case the polynomial in (26) reduces to the one in (19) with s_j selected as in (24). In the case of $a \neq b$, one can observe that the polynomial in (26) maps the eigenvalues of the operator $\boldsymbol{A}$ as follows:

$$h(\lambda_j) > 1 \qquad \text{and} \qquad |h(\lambda_i)| < 1 \quad \forall\ \lambda_i \neq \lambda_j, \tag{27}$$

which shows that $h(\boldsymbol{A})$ does not have a unit eigenvalue, thus the asynchronous updates running on $h(\boldsymbol{A})$ do not converge. In fact, the signal would diverge due to the dominant eigenvalue, $h(\lambda_j)$, being strictly larger than 1. (See Figure 1). In order to prevent the signal from diverging, we consider the following *saturated* update model:

$$\big(\boldsymbol{x}_k\big)_i = \begin{cases} f\Big(\big(h(\boldsymbol{A})\,\boldsymbol{x}_{k\text{-}1}\big)_i\Big), & i \in \mathcal{T}_k, \\ \big(\boldsymbol{x}_{k\text{-}1}\big)_i, & i \notin \mathcal{T}_k, \end{cases} \tag{28}$$

where $f(\cdot)$ is the "saturation nonlinearity" defined as follows:

$$f(x) = \text{sign}(x) \cdot \min\{|x|,\, 1\}, \tag{29}$$

which is visualized in Figure 2.

Notice that the boundedness of the function $f(\cdot)$ ensures that the signal $\boldsymbol{x}_k$ in (28) does not diverge. In fact, it is numerically observed that randomized asynchronous updates in (28) indeed converge to the fixed point of the model. That is, $\boldsymbol{x}_k$ converges to $\widehat{\boldsymbol{x}}$, where $\widehat{\boldsymbol{x}}$ satisfies the following equation:

$$f\big(h(\boldsymbol{A})\,\widehat{\boldsymbol{x}}\big) = \widehat{\boldsymbol{x}}, \tag{30}$$

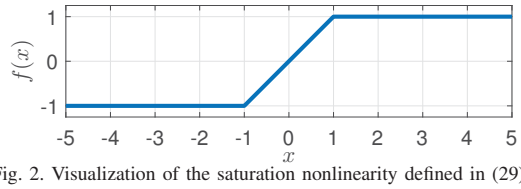


Fig. 2. Visualization of the saturation nonlinearity defined in (29).

where $f(\cdot)$ is assumed to operate element-wise on a vector.

The key observation regarding the solution of (30) is the following: when the interval $[a\ \ b]$ in (25) is small, then $\widehat{\boldsymbol{x}}$ is a good approximation of the target eigenvector $\boldsymbol{v}_j$. That is,

$$\widehat{\boldsymbol{x}} \approx \gamma\, \boldsymbol{v}_j \tag{31}$$

for some scale factor $\gamma \in \mathbb{R}$. In fact, when the target eigenvalue is known exactly, i.e., $a = b = \lambda_j$, then the approximation in (31) becomes equality. Therefore, we conclude that the asynchronous saturated update scheme in (28) allows us to find an arbitrary eigenvector approximately when the corresponding eigenvalue is not known exactly. Furthermore, as we have a better approximation of the eigenvalue, we get a better approximation of the corresponding eigenvector. The following section will make use of this observation to obtain an autonomous clustering of a network.

V. An Application: Autonomous Clustering

In this section we will consider the problem of autonomous clustering. In this regard, we will use the well-known spectral clustering [10] together with the asynchronous polynomial filtering studied in Sections III and IV.

Given a network, the second smallest eigenvalue of its graph Laplacian, λ_2, is known as the *algebraic connectivity* of the graph [11]. Roughly speaking graphs with larger λ_2 tend to be more "connected" than the others. Furthermore, the corresponding eigenvector $\boldsymbol{v}_2$, also known as the *Fiedler vector*, can be utilized to cluster the graph into two partitions. The signal $\boldsymbol{x}$ computed as

$$\boldsymbol{x} = \text{sign}(\boldsymbol{v}_2), \tag{32}$$

indicates the corresponding cluster of the nodes. Similar spectral ideas are used in [12] to obtain approximate graph coloring and in [13] to identify the hidden M-Block cyclic structure from noisy measurements under random permutations.

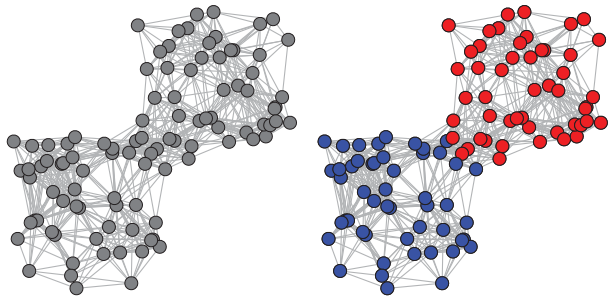
As a test example, we will use the graph visualized in Figure 3(a) in this section, some of whose Laplacian eigenvalues are numerically computed as follows:

$$\lambda_2 = 0.5814, \qquad \lambda_3 = 3.2544, \qquad \lambda_{100} = 30.3753, \tag{33}$$

and $\lambda_1 = 0$ follows from the basic properties of the Laplacian matrix. Furthermore, the graph in Figure 3(a) has the largest degree $d_{\text{max}} = 29$.

The result of the spectral clustering of Figure 3(a) based on (32) is visualized in Figure 3(b) where the clusters are represented with different colors, and we will refer this set of labels as the correct labels. In the following we will consider second order polynomial filters designed in such a way that λ_2 is the target eigenvalue so that the asynchronous iterations converge to the eigenvector $\boldsymbol{v}_2$ of the Laplacian. More precisely, we consider the following four filters:

1) The optimal second order filter considered in [9].
2) The filter in (19) by selecting s_2 as in (24) using λ_2, and setting $\lambda_{\text{low}} = 0$, $\lambda_{\text{upp}} = \lambda_{100}$.
3) The filter in (19) by selecting s_2 as in (24) using λ_2, and setting $\lambda_{\text{low}} = 0$, $\lambda_{\text{upp}} = 2\, d_{\text{max}}$.
4) The filter in (26) and setting $a = 0.3$, $b = 1.5$, $\lambda_{\text{low}} = 0$ $\lambda_{\text{upp}} = 2\, d_{\text{max}}$.



(a) (b)

Fig. 3. (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 (32) with colors representing the clusters.

Notice that parameters a, b in filter #4 satisfy the assumption in (25). In general, there is no unique way to select these parameters. However, as the interval $[a\ \ b]$ becomes smaller, the saturated updates in (28) provide a better approximation of the eigenvector $\boldsymbol{v}_2$.

When these filters are simulated on the graph, we use the update model (16) for the first three filters, but we use the saturated model (28) for filter #4. In the simulations all the nodes are initialized randomly, and the Laplacian is used as the graph operator. 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 (32). The average fraction of incorrect labels versus number of iterations is presented in Figure 4 for the four filters mentioned above.

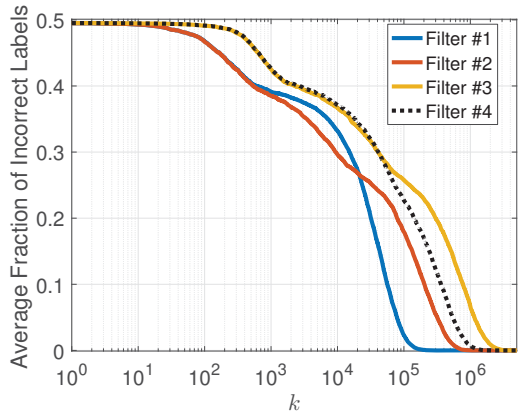


Fig. 4. Results of the autonomous clustering experiment, which are obtained by averaging over 10^4 independent experiments.

Figure 4 shows that the number of incorrect labels go down to zero as iterations progress, which is proven to be the case for filters #1 - #3 by Corollary 1 due to the construction of the filters in Section III. For filter #4, the saturated model does not converge to $\boldsymbol{v}_2$ exactly as explained in Section IV. Nevertheless, the approximation in (31) is good enough so that the sign pattern of the fixed point of the updates, defined in (30), matches the correct labels.

The figure also illustrates the trade-off between the amount of spectral information used and the rate of convergence. As discussed in [9], computation of the optimal filter requires to know all the eigenvalues of the graph operator. Filters #1, #2, and #3 use lesser and lesser amounts of spectral information. As a result, they yield lower and lower rates of convergence. Interestingly, filter #4 converges faster than filter #3 although it uses the least amount of spectral information. This is an interesting consequence of the fact that the filter in (26) results in a larger spectral gap than the one in (19) for the amount of spectral information used.

VI. Concluding Remarks & Future Work

In this paper, we considered an autonomous network model in which all the agents receive data from their neighbors repeatedly and asynchronously and update their values accordingly. Convergence of signals on such networks was analyzed using a randomized asynchronous variant of the synchronous power iteration, which performs the regular power iteration (or, the graph shift) but only a random subset of the indices are updated. It is shown that the signal under this model converges to an eigenvector of the eigenvalue 1 even in the case of operator having other eigenvalues with magnitudes larger than one. In the case of the operator not having the eigenvalue 1, the use of polynomial filters was considered. It is shown that second order polynomials are sufficient to achieve convergence to an arbitrary eigenvector. Since the design of such polynomials requires to know the spectrum of the graph, we considered an alternative construction that uses only bounds on the eigenvalues. Since random asynchronous iterations diverge on this polynomial, we considered the use of saturation in the update model, for which the iterations were observed to converge to a point that is approximately equivalent to the desired eigenvector. As an application, we combined the randomized asynchronous updates with polynomial graph filtering to compute the Fiedler vector of a network in order to achieve autonomous clustering. Simulations verified that asynchronous updates indeed cluster the network successfully even for the saturated nonlinear update model.

In future work we plan to investigate the convergence properties of the nonlinear update model more rigorously. We also plan to consider different type of nonlinearities and their effect on the convergence behavior.

References

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

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

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

[6] ——, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, April 2013.

[7] 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.

[8] O. Teke and P. P. Vaidyanathan, "The asynchronous power iteration: A graph signal perspective," in *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*, Apr. 2018, pp. 4059–4063.

[9] ——, "Random node-asynchronous updates on graphs," *Submitted to IEEE Trans. Signal Process.*, 2018.

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

[11] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973.

[12] B. Aspvall and J. R. Gilbert, "Graph coloring using eigenvalue decomposition," *SIAM Journal on Algebraic Discrete Methods*, vol. 5, no. 4, pp. 526–538, 1984.

[13] O. Teke and P. P. Vaidyanathan, "Extending classical multirate signal processing theory to graphs," in *Proc. SPIE, Wavelets and Sparsity XVII*, vol. 10394, Aug 2017.