THE ASYNCHRONOUS POWER ITERATION: A GRAPH SIGNAL PERSPECTIVE

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

This paper considers an autonomous network in which the nodes communicate only with their neighbors at random time instances, repeatedly and independently. Polynomial graph filters studied in the context of graph signal processing are inadequate to analyze signals on this type of networks. This is due to the fact that the basic shift on a graph requires all the nodes to communicate at the same time, which cannot be assumed in an autonomous setting. In order to analyze these type of networks, this paper studies an asynchronous power iteration that updates the values of only a subset of nodes. This paper further reveals the close connection between asynchronous updates and the notion of smooth signals on the graph. The paper also shows that a cascade of random asynchronous updates smooths out any arbitrary signal on the graph.

Index Terms— Graph signal processing, asynchronous iterations, autonomous networks.

1. INTRODUCTION

In recent years there is an elevated interest in network structured data where the underlying network models the dependency structure between the data sources. This is a very broad model 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 [3–5] studied the processing of signals defined over graphs. In these studies the analysis is based on the "graph operator," whose eigenvectors serve as the graph Fourier basis (GFB). With the use GFB, sampling, reconstruction, multirate processing of graph signals and some uncertainty results have been extended to the case of graphs in [6–15].

In the context of graph signal processing there is no specific definition for a "graph signal" that is agreed upon by all. In the most well-known viewpoint a typical graph signal is assumed to be smooth over the graph, that is, adjacent nodes of the graph have similar values [4, 5, 16]. The amount of smoothness is quantified with respect to the eigenvectors of the graph operator. If a signal has a large projection onto an eigenvector with large variation, then the signal is said to be non-smooth, hence anomalous on the graph [17].

Another interpretation for graph signals comes from physical processes defined over graphs. A well-known example of this category is the heat diffusion [18, 19], where the signal represents the temperature of the nodes and it evolves over time with respect to the solution of the heat equation on the underlying graph structure. Diffusions are not limited to the heat. Other kernels have been considered as well [20]. This category also includes wavelets defined over graphs [21, 22].

A different perspective is to interpret graph signals as data points. In this approach, the underlying graph operator is assumed to represent the data at hand [23–25]. The graph (operator) is selected in such way that it provides a useful (sparse, smooth, etc.) representation for the data [26, 27].

In this study we will interpret graph signals in a different problem model. We will consider an autonomous network in which there is no centralized control mechanism (e.g. a base station). An example is wireless ad-hoc networks [28]. We assume that all the nodes hold a value. Nodes communicate with their neighbors and update their value at random time instances, repeatedly and independently from each other. In this model we are interested in *the general behavior of the values held on the nodes.*

It is important to note that the notion of the graph shift is not directly applicable to this problem formulation. This is due to the fact that the graph shift needs all the nodes to communicate at the same time instance. Even though it is a local operator (requires communication only with neighbors) synchronization of the nodes is essential, which cannot be achieved in an autonomous setting. As a result, polynomial filters based on the graph shift are not applicable as well.

In order to analyze this problem we will consider a variation of the graph shift referred to as asynchronous power iteration. It will be similar to the graph shift except the fact that only a subset of nodes are updated in each iteration. Such a scheme is useful to analyze autonomous networks, and will be shown to be closely related to the notion of smooth signals.

In the following, we first provide an overview of the notation. In Section 2 we define the asynchronous power iteration precisely and elaborate on its connection to smooth graph signals. In Section 3 we consider random cascades of asynchronous iterations and study their convergence behavior.

1.1. Preliminaries and Notation

We will assume that $\boldsymbol{A} \in \mathcal{C}^{N\times N}$ is an operator on the graph of interest on N nodes. We only require $\boldsymbol{A}$ to be a *local* operator, that is, $A_{i,j} = 0$ when the nodes i and j are not neighbors. We allow $A_{i,i}$ to be non-zero. Hence, the operator $\boldsymbol{A}$ can be the adjacency matrix, the Laplacian, the normalized Laplacian, and so on. We assume that $\boldsymbol{A}$ is a normal matrix and has the following eigenvalue decomposition:

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^*, \tag{1}$$

where $\boldsymbol{V}$ is a unitary matrix consisting of eigenvectors of $\boldsymbol{A}$, and $\boldsymbol{\Lambda}$ is the diagonal matrix with the eigenvalues. We will use $\boldsymbol{V}^*$ to denote the conjugate transpose of $\boldsymbol{V}$. We allow eigenvalues to have complex values in general. Given a signal $\boldsymbol{x}$, its graph Fourier transform (GFT), $\widehat{\boldsymbol{x}}$, on the operator $\boldsymbol{A}$ is defined as:

$$\widehat{\boldsymbol{x}} = \boldsymbol{V}^*\,\boldsymbol{x}, \qquad \text{or,} \qquad \boldsymbol{x} = \sum_{i=1}^{N} \widehat{x}_i\;\boldsymbol{v}_i, \tag{2}$$

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

The number of elements in a set $\mathcal{T}$ is denoted by $|\mathcal{T}|$. For a vector $\boldsymbol{x}$ we use $\|\boldsymbol{x}\|_\infty$ to denote its largest element in absolute sense. For a matrix $\boldsymbol{V}$ we use $\|\boldsymbol{V}\|_\infty$ to denote the largest absolute row-sum, and use $\|\boldsymbol{V}\|_{\max}$ to denote the largest element. We will use $\boldsymbol{e}_i$ to denote the i^{th} standard vector that has 1 only at the i^{th} index.

This work was supported in parts by the ONR grants N00014-15-1-2118 and N00014-17-1-2732, the NSF grant CCF-1712633, and the Electrical Engineering Carver Mead Research Seed Fund of the California Institute of Technology.

2. ASYNCHRONOUS POWER ITERATION

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

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\,\boldsymbol{x}_k, \qquad \text{so that} \qquad \boldsymbol{x}_k = \boldsymbol{A}^k\,\boldsymbol{x}_0. \tag{3}$$

Assuming that the eigenvalues of $\boldsymbol{A}$ satisfy $|\lambda| \leqslant 1$ (or, normalizing the vector $\boldsymbol{x}_k$ after each iteration) the power iteration can be utilized to compute the eigenvector of the matrix $\boldsymbol{A}$ with the largest eigenvalue. One such application is Google's ranking algorithm [29].

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 [4]. From this perspective $\boldsymbol{x}_{k+1}$ in (3) is the graph shifted version of $\boldsymbol{x}_k$. 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. However, all the nodes need to send and receive data at the same time. Therefore, the graph shift does not have a "distributed" implementation since it still requires a centralized timing/control mechanism over the underlying graph.

In this study we will consider a variation of the power iteration, where a subset of indices, $\mathcal{T}$, are updated simultaneously and the remaining ones stay as the same. More precisely, given an update set $\mathcal{T}$ we consider the following *asynchronous* power iteration:

$$y_i = \begin{cases} (\boldsymbol{A}\boldsymbol{x})_i, & i \in \mathcal{T}, \\ x_i, & i \notin \mathcal{T}, \end{cases} \tag{4}$$

where $\boldsymbol{x}$ is the initial vector and $\boldsymbol{y}$ is the vector after the iteration. In words, this iteration computes the multiplication $\boldsymbol{A}\boldsymbol{x}$, but it only updates the values of the elements indexed by the set $\mathcal{T}$, and keeps the remaining elements the same. Eq. (3) will also be referred to as "synchronous update" to distinguish it from (4). Both (3) and (4) are also referred to as *state recursions*, where the graph signal $\boldsymbol{x}$ is regarded as the state of a system [30]. The model in (4) can also be considered as the Hopfield network without the non-linearity [31].

In the actual implementation of this scheme only the elements in the set $\mathcal{T}$ need to be synchronized. When the update set is selected as $\mathcal{T} = \{1, \cdots, N\}$, the asynchronous iteration in (4) reduces to the classical power iteration in (3). On the other extreme, if a single node is updated, $|\mathcal{T}| = 1$, then no synchronization is required at all.

The asynchronous iteration defined in (4) can be written as a matrix-vector multiplication as follows:

$$\boldsymbol{y} = \sum_{i \notin \mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*\,\boldsymbol{x} + \sum_{i \in \mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*\,\boldsymbol{A}\,\boldsymbol{x} = \Big[\boldsymbol{I} + \sum_{i \in \mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*(\boldsymbol{A} - \boldsymbol{I})\Big]\,\boldsymbol{x}, \tag{5}$$

which can be converted into the graph Fourier domain as follows:

$$\widehat{\boldsymbol{y}} = \widehat{\boldsymbol{x}} + \sum_{i \in \mathcal{T}} \boldsymbol{V}^*\boldsymbol{e}_i\,\boldsymbol{e}_i^*\,\boldsymbol{V}\,(\boldsymbol{\Lambda} - \boldsymbol{I})\,\widehat{\boldsymbol{x}}. \tag{6}$$

2.1. Interpretation in the Context of Graph Signal Processing

In order to relate the asynchronous iteration to graph signal processing we first discuss the notion of smooth signals on graphs. For this purpose we will consider the following definition:

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

$$|\widehat{x}_j|\;|\lambda_j - 1| \;\leqslant\; \epsilon \qquad \forall\, j \tag{7}$$

on the given graph operator.

A signal $\boldsymbol{x}$ belongs to $\mathcal{S}_\epsilon$ if the difference between the graph Fourier coefficients of $\boldsymbol{x}$ and $\boldsymbol{A}\boldsymbol{x}$ are not larger than ϵ in absolute sense. Small values of ϵ implies that $\boldsymbol{x}$ and $\boldsymbol{A}\boldsymbol{x}$ are similar to each other. Hence, we can interpret ϵ as a scale of the smoothness of the signal $\boldsymbol{x}$ on the operator $\boldsymbol{A}$. Here, the smoothness is quantified with respect to the total variation (TV) of each eigenvector, $\text{TV}(\boldsymbol{v}_i) = |\lambda_i - 1|$, as in [17]. For a given value of ϵ, the set $\mathcal{S}_\epsilon$ describes the signals with $|\widehat{x}_j| \leqslant \epsilon\,/\,|\lambda_j - 1|$, that is, the amount of projection of $\boldsymbol{x}$ onto an eigenvector should decrease with respect to the total variation of the eigenvector. For the case of $\epsilon = 0$, it implies that $\widehat{x}_j = 0$ for any $\lambda_j \neq 1$, that is, the signal $\boldsymbol{x}$ cannot have any non-smooth (or, high frequency) component.

The condition in (7) is equivalent to upper bounding a weighted max-norm of GFT of $\boldsymbol{x}$, that is,

$$\boldsymbol{x} \in \mathcal{S}_\epsilon \quad \Longleftrightarrow \quad \big\|\,(\boldsymbol{\Lambda} - \boldsymbol{I})\,\widehat{\boldsymbol{x}}\,\big\|_\infty \leqslant \epsilon, \tag{8}$$

where the weight matrix is selected as $|\boldsymbol{\Lambda} - \boldsymbol{I}|$. Therefore, $\mathcal{S}_\epsilon$ is a convex set. More importantly, 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.*

The following theorem reveals the relationship between the smooth graph signals and the asynchronous update in (4).

Theorem 1. *Assume that the signal $\boldsymbol{x}$ belongs to $\mathcal{S}_\epsilon$ of a graph with operator $\boldsymbol{A}$. Then the signal $\boldsymbol{y}$ computed as in* (4) *satisfies the following*

$$\|\widehat{\boldsymbol{y}} - \widehat{\boldsymbol{x}}\|_\infty \;\leqslant\; \epsilon\,|\mathcal{T}|\,\|\boldsymbol{V}\|_{\max}\|\boldsymbol{V}\|_\infty \tag{9}$$

Proof: Assume that $\boldsymbol{x} \in \mathcal{S}_\epsilon$. Then, we can write the following set of inequalities:

$$|\widehat{y}_j - \widehat{x}_j| = \Big|\boldsymbol{e}_j^* \sum_{i \in \mathcal{T}} \boldsymbol{V}^*\boldsymbol{e}_i\,\boldsymbol{e}_i^*\,\boldsymbol{V}\,(\boldsymbol{\Lambda} - \boldsymbol{I})\,\widehat{\boldsymbol{x}}\Big| \tag{10}$$

$$\leqslant \sum_{i \in \mathcal{T}} \big|\boldsymbol{e}_j^*\boldsymbol{V}^*\boldsymbol{e}_i\big|\;\Big|\boldsymbol{e}_i^*\,\boldsymbol{V}\,(\boldsymbol{\Lambda} - \boldsymbol{I})\,\widehat{\boldsymbol{x}}\Big| \tag{11}$$

$$\leqslant \sum_{i \in \mathcal{T}} \|\boldsymbol{V}\|_{\max}\;\big\|\boldsymbol{V}^*\,\boldsymbol{e}_i\big\|_1\,\big\|(\boldsymbol{\Lambda} - \boldsymbol{I})\,\widehat{\boldsymbol{x}}\big\|_\infty \tag{12}$$

$$\leqslant \epsilon\,|\mathcal{T}|\,\|\boldsymbol{V}\|_{\max}\,\|\boldsymbol{V}\|_\infty, \tag{13}$$

where we use Hölder inequality in (11). The step in (12) uses (8), and the fact that $\|\boldsymbol{V}\|_\infty$ is the largest ℓ_1-norm of the rows of $\boldsymbol{V}$. Then, we have the following:

$$\|\widehat{\boldsymbol{y}} - \widehat{\boldsymbol{x}}\|_\infty = \max_j |\widehat{y}_j - \widehat{x}_j| \leqslant \epsilon\,|\mathcal{T}|\,\|\boldsymbol{V}\|_{\max}\,\|\boldsymbol{V}\|_\infty, \tag{14}$$

where the inequality follows from the fact that the bound in (13) is valid for any j. ∎

Corollary 1. *Assume that the signal $\boldsymbol{x}$ belongs to $\mathcal{S}_\epsilon$ of a circulant graph with the operator being either the adjacency or the Laplacian. Then, the signal $\boldsymbol{y}$ computed as in* (4) *satisfies the following*

$$\|\widehat{\boldsymbol{y}} - \widehat{\boldsymbol{x}}\|_\infty \;\leqslant\; \epsilon\,|\mathcal{T}|. \tag{15}$$

Proof: For circulant graphs the adjacency matrix and the Laplacian can be diagonalized by the unitary DFT matrix of size N, in which case we have $\|\boldsymbol{V}\|_{\max} = 1/\sqrt{N}$ and $\|\boldsymbol{V}\|_\infty = \sqrt{N}$, hence the result according to (13). ∎

The bound given by Theorem 1 (and Corollary 1) is not tight in general. Nevertheless, it provides a useful interpretation: after a single asynchronous update, the amount of change in *each* GFT coefficient is limited by the smoothness, ϵ, of the signal (up to some graph dependent constant). In particular, if the signal is smooth on the graph (belongs to $\mathcal{S}_\epsilon$ with ϵ being small), then amount of change in each GFT coefficient is also small, that is, *a smooth signal remains to be (relatively) smooth on the graph after an asynchronous update*.

This observation motivates us to use the model in (4) in order to analyze the behavior of autonomous networks. When the update set $\mathcal{T}$ has $|\mathcal{T}| = 1$ a single iteration corresponds to the case of a node updating its value. Here the local graph operator $\boldsymbol{A}$ models the way nodes update their values. If a node computes the sum of its

neighbors, then $\boldsymbol{A}$ is the adjacency matrix; if a node computes the sum of differences with its neighbors, then $\boldsymbol{A}$ is the graph Laplacian; if a node computes a weighted average of its neighbors, then $\boldsymbol{A}$ is a weighted adjacency matrix. Therefore, the matrix $\boldsymbol{A}$ describes what the nodes compute, and the update scheme in (4) describes the dynamics of the network in time.

In this context Theorem 1 shows that if the graph signal is smooth with respect to the underlying computation scheme (the matrix $\boldsymbol{A}$), then values remain to be smooth after a node updates its value. Therefore, the values of the nodes are expected to form a smooth signal on the operator $\boldsymbol{A}$ in general.

A single update, however, is not sufficient to model the general behavior of a network of nodes. Since nodes communicate and update their values randomly and repeatedly, we need to consider a *random cascade of updates* in the form of (4). As a result, general behavior of the values held on the nodes is mathematically equivalent to the limit of a random cascade of these updates. Since Theorem 1 considers the difference between only two consecutive iterations, it is inconclusive regarding the limiting behavior of a random cascade. A curious question is as follows: if the initial signal is *not* smooth on $\boldsymbol{A}$, does it get smoother over iterations? In the next sections we will show that this is indeed the case.

3. CONVERGENCE OF ITERATIONS

In the following we will consider a general case and assume that t nodes are updated at the same time. That is, the update set $\mathcal{T}$ is of size $|\mathcal{T}| = t$. Notice that $t = 1$ corresponds to the case where only a single node is updated asynchronously, and $t = N$ corresponds to the power iteration defined in (3).

We first consider a single *random* iteration, which is written as:

$$\boldsymbol{y} = \boldsymbol{Q}\,\boldsymbol{x} \tag{16}$$

where $\boldsymbol{x}$ is a known deterministic initial vector, whereas $\boldsymbol{Q}$ is a matrix random variable due to the fact that the underlying update set $\mathcal{T}$ is random. We do not assume any prior information on $\mathcal{T}$ and assume that all the sets of size t are equally likely. In the case of $t = 1$, uniform prior means that all the nodes update their values at the same rate. Then, the iteration matrix $\boldsymbol{Q}$ has the following distribution:

$$\boldsymbol{Q} = \boldsymbol{I} + \sum_{i\in\mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*\,(\boldsymbol{A}-\boldsymbol{I}) \qquad \text{with probability} \qquad \frac{1}{\binom{N}{t}}. \tag{17}$$

Lemma 1. *Expectation of the random matrix $\boldsymbol{Q}$ in* (17) *is:*

$$\mathbb{E}[\boldsymbol{Q}] = \frac{t}{N}\,\boldsymbol{A} + \left(1-\frac{t}{N}\right)\boldsymbol{I}. \tag{18}$$

Proof: We have

$$\mathbb{E}[\boldsymbol{Q}] = \sum_{\substack{\mathcal{T}\ \text{s.t.}\\ |\mathcal{T}|=t}} \frac{1}{\binom{N}{t}}\left[\boldsymbol{I} + \sum_{i\in\mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*\,(\boldsymbol{A}-\boldsymbol{I})\right] \tag{19}$$

$$= \frac{1}{\binom{N}{t}} \sum_{\substack{\mathcal{T}\ \text{s.t.}\\ |\mathcal{T}|=t}} \boldsymbol{I} + \frac{1}{\binom{N}{t}}\left[\sum_{\substack{\mathcal{T}\ \text{s.t.}\\ |\mathcal{T}|=t}} \sum_{i\in\mathcal{T}} \boldsymbol{e}_i\,\boldsymbol{e}_i^*\right](\boldsymbol{A}-\boldsymbol{I}) \tag{20}$$

$$= \boldsymbol{I} + \frac{1}{\binom{N}{t}}\binom{N\text{-}1}{t\text{-}1}\,\boldsymbol{I}\,(\boldsymbol{A}-\boldsymbol{I}), \tag{21}$$

which gives the result in (18). ∎

Notice that t/N is the fraction of the nodes that are being updated concurrently, and it appears as the weight of the graph operator $\boldsymbol{A}$ in $\mathbb{E}[\boldsymbol{Q}]$. When $t = 0$, we get $\mathbb{E}[\boldsymbol{Q}] = \boldsymbol{I}$, that is, no node is updated, hence the signal stays the same. When $t = N$ it means all the nodes are updated at the same time and we get $\mathbb{E}[\boldsymbol{Q}] = \boldsymbol{A}$, which corresponds to the case of power iteration in (3).

Next, we consider a *cascade* of k updates in the following form:

$$\boldsymbol{y}[k] = \boldsymbol{Q}_k\,\boldsymbol{Q}_{k\text{-}1}\cdots\,\boldsymbol{Q}_2\,\boldsymbol{Q}_1\,\boldsymbol{x}, \tag{22}$$

where $\boldsymbol{y}[k]$ represents the vector after k iterations with $\boldsymbol{x}$ being the initial vector. For the sake of clarity $\boldsymbol{y}[k]$ will be denoted simply with $\boldsymbol{y}$, but it should be clear that $\boldsymbol{y}$ depends on k in general.

Let $\mathcal{T}_i$ be the set of nodes that are updated in the i^{th} iteration in (22). We assume that the size of $\mathcal{T}_i$'s stays as the same through updates, that is, $|\mathcal{T}_1| = \cdots |\mathcal{T}_k| = t$. Nevertheless, sets themselves are different from one iteration to another and they are selected uniformly at random. Therefore $\boldsymbol{Q}_i$'s have the same distribution, which is given in (17). We also assume that $\mathcal{T}_i$'s are selected independently from each other, hence $\boldsymbol{Q}_i$'s are independent as well.

For the specific case of $t = 1$ the problem set-up in (22) is suitable to analyze the scenario discussed in the previous section, that is, nodes repeatedly communicate with their neighbors at random time instances and update their value according to the graph operator $\boldsymbol{A}$. Here k denotes the total number of updates, and the question at hand is the behavior of $\boldsymbol{y}$ as k goes to infinity. The following theorem studies the limit of the expectation of the GFT of $\boldsymbol{y}$.

Theorem 2. *Let $\boldsymbol{v}_j$ and λ_j be an eigenpair of $\boldsymbol{A}$, and let $\widehat{y}_j = \boldsymbol{v}_j^*\,\boldsymbol{y}$ be the j^{th} graph Fourier coefficient of $\boldsymbol{y}$. Then the following is true:*

$$\left|1 + t/N\,(\lambda_j - 1)\right| < 1 \qquad \Longrightarrow \qquad \lim_{k\to\infty} \mathbb{E}[\widehat{y}_j] = 0, \tag{23}$$

where t is the size of the update set of each iteration.

Proof: We have

$$\mathbb{E}[\widehat{y}_j] = \mathbb{E}[\boldsymbol{v}_j^*\,\boldsymbol{y}] = \boldsymbol{v}_j^*\,\mathbb{E}[\boldsymbol{Q}_k\,\boldsymbol{Q}_{k\text{-}1}\cdots\,\boldsymbol{Q}_1]\,\boldsymbol{x} \tag{24}$$

$$= \boldsymbol{v}_j^*\,\mathbb{E}[\boldsymbol{Q}_k]\,\mathbb{E}[\boldsymbol{Q}_{k\text{-}1}]\cdots\mathbb{E}[\boldsymbol{Q}_1]\,\boldsymbol{x} = \boldsymbol{v}_j^*\,\left(\mathbb{E}[\boldsymbol{Q}]\right)^k\,\boldsymbol{x} \tag{25}$$

where (25) follows from the fact that $\boldsymbol{Q}_i$'s are independent and identical random variables. By writing the k^{th} power of $\mathbb{E}[\boldsymbol{Q}]$ using (18) and the spectral decomposition of $\boldsymbol{A}$ given in (1), we get the following:

$$\mathbb{E}[\widehat{y}_j] \;=\; \boldsymbol{v}_j^*\,\boldsymbol{V}\left(\frac{t}{N}\,\boldsymbol{\Lambda} + \left(1-\frac{t}{N}\right)\boldsymbol{I}\right)^k\,\boldsymbol{V}^*\,\boldsymbol{x} \tag{26}$$

$$\;=\; \left(1 + t/N\,(\lambda_j - 1)\right)^k\,\widehat{x}_j. \tag{27}$$

As a result, $\left|1 + t/N\,(\lambda_j - 1)\right| < 1$ implies that $\mathbb{E}[\widehat{y}_j]$ goes to zero as k goes to infinity. ∎

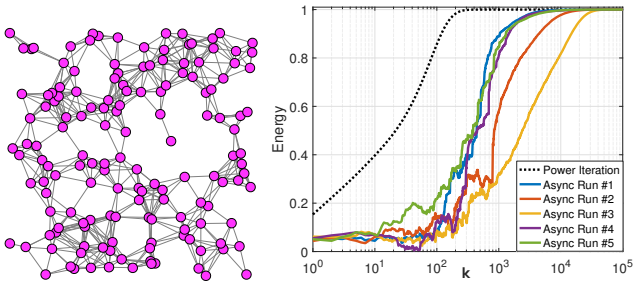
The result of Theorem 2 suggests that some graph Fourier coefficients of the signal $\boldsymbol{y}$ in (22) converge to zero as the number of asynchronous updates go to infinity if the corresponding eigenvalues are bounded in some sense. This is similar to the power iteration in (3), where eigenvalues lying inside the unit complex circle are known to converge to zero as iterations continue. However, the convergence of a cascade of random asynchronous updates behaves in some expected and unexpected ways as we shall discuss in the next sections.

3.1. An Example on a Random Geometric Graph

In this section we will simulate the convergence behavior of asynchronous iterations in (4) on the graph visualized in Figure 1(a). This is a random geometric graph on $N = 150$ nodes, in which nodes are distributed over the region $[0\ \ 1] \times [0\ \ 1]$ uniformly at random. Two nodes are connected to each other if the distance between them is less than 0.15. The graph operator is selected as the normalized adjacency matrix, that is $\boldsymbol{D}^{-1/2}\,\boldsymbol{A}\,\boldsymbol{D}^{-1/2}$, where $\boldsymbol{A}$ is the adjacency matrix and $\boldsymbol{D}$ is the diagonal degree matrix. Eigenvalues of $\boldsymbol{D}^{-1/2}\,\boldsymbol{A}\,\boldsymbol{D}^{-1/2}$ can be arranged as follows: $1 = \lambda_1 > \lambda_2 \geqslant \cdots \geqslant \lambda_N >$ -1, where $\lambda_1 = 1$ and $\lambda_2 < 1$ follow from the fact that the graph in Figure 1(a) is connected.

Notice that λ_j's for $j \geqslant 2$ satisfy the condition in Theorem 2, that is $|1{+}t/N(\lambda_j{-}1)| < 1$ for any $t \geqslant 1$. Therefore, as the total number of random iterations, k, goes to infinity, expectation of $\widehat{y}_j$ goes to zero for all $j \geqslant 2$. However, $\lambda_1 = 1$ does *not* satisfy the condition, hence $\mathbb{E}[\widehat{y}_1]$ does not converge to zero. As a result, we expect that $\boldsymbol{y}$ approaches $\boldsymbol{v}_1$ as k goes to infinity.

In order to validate this expectation we perform the following experiment. We consider the case $t = 1$, that is, a randomly selected single node updates its value at each iteration, and we observe the value of $|\boldsymbol{v}_1^* \boldsymbol{y}| / \|\boldsymbol{y}\|_2$ through iterations. This quantity corresponds to the fraction of energy of $\boldsymbol{y}$ contained in the subspace spanned by $\boldsymbol{v}_1$. Starting from the same initial vector $\boldsymbol{x}$, we consider 5 different runs each having $k = 10^5$ iterations. The fraction of energy through iterations is given in Figure 1(b). For the sake of comparison, the result of the power iteration in (3) (which is equivalent to the case of $t = N$) is also included.



(a) (b)

Fig. 1. (a) Random geometric graph with $d = 0.15$ on $N = 150$ nodes. (b) Fraction of the energy over iterations.

Unlike the power iteration in (3) in the case of random asynchronous updates the fraction of energy does not increase monotonically with the total number of iterations. This is due to the random selection (lack of centralization) of the nodes. Some updates might be adversarial to the convergence of the signal. In some cases adversarial updates might occur more often, which slow down the convergence (notice the difference between run#1 and run#3 in Figure 1(b)). Nevertheless, some other updates cancel them out in the long run. Therefore, as k goes to infinity the signal $\boldsymbol{y}$ converges to $\boldsymbol{v}_1$ up to a scale since $|\boldsymbol{v}_1^* \boldsymbol{y}| = \|\boldsymbol{y}\|_2$ implies that $\boldsymbol{y}$ is in the span of $\boldsymbol{v}_1$. This observation is consistent with the result of Theorem 2.

It is important to note that the results in Figure 1(b) do *not* suggest that the power iteration converges faster than asynchronous updates. In a single power iteration all N nodes update their values, whereas only a single node updates its value in the asynchronous case. Therefore the total number of updated nodes, which is found by $k \cdot t$, is the correct way to compare the convergence rates. Notice that asynchronous updates in Figure 1(b) converge around $k \approx 10^5$ and have $t = 1$, whereas the power iteration has $t = N = 150$ and converges around $k \approx 700$. This observation suggests that both the asynchronous and the synchronous iterations require roughly the same number of node updates for the convergence.

3.2. Convergence Regions

The condition $|1+t/N(\lambda\text{-}1)| < 1$ in Theorem 2 defines a region of convergence for the eigenvalues of the operator $\boldsymbol{A}$ for the case of asynchronous updates. This region is a disk in the complex plane centered at $1\text{-}N/t$ with radius N/t. These regions are visualized in Figure 2 for various different values of t.

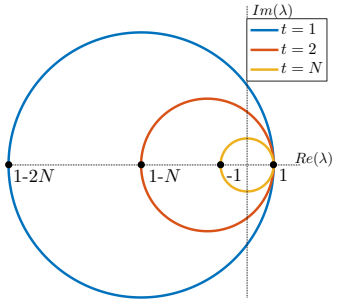


Fig. 2. Convergence circles.

In the case of $t = N$ the region reduces to the unit disk, which is the condition on the eigenvalues for the power iteration to converge. This is not surprising since the case of $t = N$ corresponds to the power iteration itself. However, *as t gets smaller (less number of nodes are updated concurrently), the convergence region gets larger.* This is an interesting result since the coefficients (in the decomposition of the signal in terms of the eigenvectors) of a larger set of eigenvectors converge to zero if some of the nodes are not updated with the rest. Even if a single node remains the same in each iteration, $t = N$-1, the coefficients of some eigenvectors converge to zero although their corresponding eigenvalues have absolute value larger than unity. This is a remarkable property of the asynchronous iterations since such coefficients (hence the signal itself) would blow up in the case of the power iteration. In particular, the coefficients of eigenvectors with eigenvalues located on the unit circle (except $\lambda = 1$) converge to zero, which cannot happen in the case of the power iteration. In order to verify this behavior we will consider the asynchronous updates in the context of graph signals. For this purpose, let $\boldsymbol{A}$ be the adjacency matrix of the directed cycle graph:

$$\boldsymbol{A} = \begin{bmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix}. \tag{28}$$

Eigenvalues of $\boldsymbol{A}$ in (28) are known to be $\lambda_n = e^{j2\pi(n\text{-}1)/N}$ for $1 \leqslant n \leqslant N$, and the corresponding eigenvectors are the columns of the unitary DFT matrix [17].

The power iteration in (3) on the directed cycle graph corresponds to the case that all the nodes update their values with the values of the previous nodes. This operation is equivalent to the circular shift of the graph signal $\boldsymbol{x}$. As power iterations are applied repeatedly the initial signal cycles through N different states indefinitely and do not converge to any point. This is a direct consequence of the eigenvalues being on the unit circle: no eigenvalue is in the convergence region, hence the coefficient of no eigenvector dies out through iterations.

On the contrary, in the case of asynchronous iterations all the eigenvalues except $\lambda_1 = 1$ fall into the convergence region, that is, $|1 + t/N(\lambda_n - 1)| < 1$ for any $1 \leqslant t < N$. Notice that the eigenvector that corresponds to $\lambda_1 = 1$ is the constant vector, that is, $\boldsymbol{v}_1 = \mathbb{1}$. Therefore, we expect the initial signal to converge to the constant signal through asynchronous iterations. On the directed cycle graph, a random asynchronous update with $t = 1$ is equivalent to randomly selecting a node and assigning its value to the next one. This operation results in two nodes with the same value. As random updates are applied repeatedly the signal will have more and more duplicate elements until all the elements are the same. This shows that *any* initial signal converges to $\boldsymbol{v}_1$ (up to a scale) indeed. Such a convergence cannot be achieved with the power iteration.

4. CONCLUDING REMARKS & FUTURE DIRECTIONS

In this study we assumed an autonomous network in which the nodes update their values at random times, repeatedly and independently via communicating with their neighbors. Since the notion of the graph shift was not applicable to this problem setting, we considered an asynchronous update scheme. We showed the close relation between smooth signals and the update scheme. The behavior of asynchronous updates revealed that an arbitrary signal converges to the eigenvector with eigenvalue $\lambda = 1$, which has zero total-variation. This shows that an arbitrary signal gets smoothed out by the asynchronous updates throughout the iterations. Therefore, a typical signal on an autonomous network is the smoothest signal with respect to the graph operator.

In future we will analyze the convergence of asynchronous iterations more rigorously and elaborate on the rate of convergence. We will consider a more general scenario where nodes have different rate of updates, which is equivalent to having a non-uniform prior on the update sets. We will examine the effect of the noise as well.

5. REFERENCES

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

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

[3] 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 Processing Magazine*, vol. 31, no. 5, pp. 80–90, Sept. 2014.

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

[5] 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 Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[6] 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.

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

[8] A. Anis, A. Gadde, and A. Ortega, "Towards a sampling theorem for signals on arbitrary graphs," in *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*, May 2014, pp. 3864–3868.

[9] A. Gadde and A. Ortega, "A probabilistic interpretation of sampling theory of graph signals," in *Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP)*, April 2015, pp. 3257–3261.

[10] 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.

[11] ——, "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.

[12] 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.

[13] 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.

[14] 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, 2016.

[15] 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.

[16] 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, April 2016.

[17] 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, June 2014.

[18] F. R. K. Chung, *Spectral Graph Theory*. AMS, 1997.

[19] O. Teke and P. P. Vaidyanathan, "Time estimation for heat diffusion on graphs," in *Asilomar Conf. on Signals, Systems and Computers*, 2017.

[20] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *In Proceedings of the ICML*, 2002, pp. 315–322.

[21] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53–94, 2006.

[22] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129 – 150, 2011.

[23] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[24] D. Thanou, D. I. Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Trans. on Sig. Process.*, vol. 62, no. 15, pp. 3849–3862, Aug. 2014.

[25] S. Chen, D. Tian, C. Feng, A. Vetro, and J. Kovacevic, "Fast resampling of 3d point clouds via graphs," *arXiv:1702.06397*, Feb. 2017.

[26] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under laplacian and structural constraints," *IEEE Journal of Sel. Top. in Sig. Process.*, vol. 11, no. 6, pp. 825–841, Sept. 2017.

[27] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Trans. on Sig. and Inf. Process. over Net.*, vol. 3, no. 3, pp. 467–483, Sept. 2017.

[28] Z. J. Haas, J. Deng, B. Liang, P. Papadimitratos, and S. Sajama, *Wireless ad hoc Networks*. John Wiley Sons, Inc., 2003.

[29] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Technical Report, November 1999.

[30] L. Chua and D. Green, "A qualitative analysis of the behavior of dynamic nonlinear networks: Stability of autonomous networks," *IEEE Transactions on Circuits and Systems*, vol. 23, no. 6, pp. 355–379, June 1976.

[31] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.