Signal Processing 196 (2022) 108530

Contents lists available at ScienceDirect

Signal Processing

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

Distributed algorithms to determine eigenvectors of matrices on spatially distributed networks

Nazar Emirov[a], Cheng Cheng[b,*], Qiyu Sun[c], Zhihua Qu[d]

[a] Department of Computer Science, Boston College, Chestnut Hill, Massachusetts 02467, USA
[b] School of Mathematics, Sun Yat-sen University, Guangzhou, Guangdong 510275, China
[c] Department of Mathematics, University of Central Florida, Orlando, Florida 32816, USA
[d] Department of Electrical and Computer Engineering, University of Central Florida, Orlando, Florida 32816, USA

ARTICLE INFO

Article history:
Received 4 August 2021
Revised 2 March 2022
Accepted 3 March 2022
Available online 4 March 2022

Keywords:
Eigenvector
Matrix on graphs
Preconditioned gradient descent algorithm
Spatially distributed network

ABSTRACT

Eigenvectors of matrices on a network have been used for understanding influence of a vertex and spectral clustering. For matrices with small geodesic-width and their given eigenvalues, we propose preconditioned gradient descent algorithms in this paper to find eigenvectors. We also consider synchronous implementation of the proposed algorithms at vertex/agent level in a spatially distributed network in which each agent has limited data processing capability and confined communication range.

1. Introduction

Spatially distributed networks (SDNs) consist of a large amount of agents, and each agent is equipped with subsystems for limited data processing and direct communication link to its “neighboring” agents within communication range. SDNs appear in (wireless) sensor networks, smart grids, social network and many real world applications [1–9]. In this paper, we describe the topological structure of an SDN by a finite graph $\mathcal{G} := (V, E)$ with a vertex in V representing an agent, an edge in E between vertices indicating that the corresponding agents are within some range in the spatial space, and the communication range L being the maximal geodesic distance such that direct communication link between agents $i, j \in V$ exists whenever $\rho(i, j) \le L$, where the geodesic distance $\rho(i, j)$ is the number of edges in a shortest path connecting $i, j \in V$. As SDNs may not have a central server, data processing on SDNs should be designed at the agent/vertex level with direct data exchanging between neighboring vertices within the communication range L.

Matrices on SDNs appear as filters in graph signal processing, transition matrices in Markov chains, state matrices of dynamic systems in control theory, sensing matrices in sampling theory, and in many more applications [6,9–16]. In this paper, we consider complex-valued graph filters represented by matrices $\mathbf{A} = (A(i, j))_{i,j\in V}$ on the graph $\mathcal{G} = (V, E)$ with limited geodesic-width $\omega(\mathbf{A})$, which is the smallest nonnegative integer such that $A(i, j) = 0$ for all $i, j \in V$ satisfying $\rho(i, j) > \omega(\mathbf{A})$ [9,16–19]. Our illustrative examples of graph filters with small geodesic-width are the graph Laplacian matrix with geodesic-width one and polynomial graph filters with geodesic-width no more than the degree of the polynomial [15]. The concept of geodesic-width for graph filters can be considered as the correspondence of the duration for finite impulse response filters in the graph setting.

Eigenspaces of matrices with limited geodesic-width have been used to understand the communicability between vertices, spectral clustering for the network, and influence of a vertex on the network, see [3,6,20–32] and references therein. Most of distributed algorithms proposed in the literature, such as the conventional power method and gradient descent method, are based on the consensus approach and required to have some prior global parameters other than the eigenvalue, such as the largest eigenvalue or its estimation, for the normalization, see Remarks 2.4 and 3.3. Based on the preconditioned gradient descent approach in [16] for inverse filtering, in Section 2 we propose a preconditioned gradient descent algorithm (PGDA) to determine eigenvectors associated with an eigenvalue λ of a given matrix $\mathbf{A}$ with limited geodesic-width $\omega(\mathbf{A})$, which is proved to converge exponentially to the desired eigenvector without any global assumptions on spectrum of the matrix and the magnitudes of its entries, see Theorem 2.1 and Remark 2.3. More importantly, the proposed algorithm can be im-

[*] Corresponding author.
E-mail addresses: nazar.emirov@bc.edu (N. Emirov), chengch66@mail.sysu.edu.cn (C. Cheng), qiyu.sun@ucf.edu (Q. Sun), Zhihua.Qu@ucf.edu (Z. Qu).

https://doi.org/10.1016/j.sigpro.2022.108530
0165-1684/© 2022 Elsevier B.V. All rights reserved.

Algorithm 1 Realization of the PGDA at a vertex $i \in V$.

```
Inputs: The totaliteration number $M$, the geodesic-width $\omega(\mathbf{H})$ of the matrix $\mathbf{H} = (H(i,j))_{i,j \in V}$,the set $B(i, \omega(\mathbf{H}))$ of $\omega(\mathbf{H})$-hop neighbors of the vertex $i$,the eigenvalue $\lambda$ of the matrix $\mathbf{H}$,entries $H(i,j)$ and $H(j,i), j \in B(i, \omega(\mathbf{H}))$ in the $i$th row and column of the matrix $\mathbf{H}$,and the $i$th diagonal entry $Q(i,i)$ of the matrix $\mathbf{Q}$.
Pre-iteration: Compute $A(i,j) = H(i,j) - \lambda\delta(i,j)$and $\tilde{A}(j,i) = (Q(i,i))^{-2}\big(\overline{H(j,i)} - \bar{\lambda}\delta(j,i)\big)$ for $j \in B(i, \omega(\mathbf{H}))$,where $\delta$ is the Kronecker delta.
Initial:Select the $i$th component $x_0(i) \in [0,1]$ of the initial vector $\mathbf{x}_0$ randomly, and set $n = 0$.
Iteration:

  1. Send $x_n(i)$ to all neighbors $k \in B(i, \omega(\mathbf{H}))\backslash\{i\}$ andreceive $x_n(k)$ from neighbors $k \in B(i, \omega(\mathbf{H}))\backslash\{i\}$.
  2. Evaluate $\tilde{x}_n(i) = \sum_{j \in B(i,\omega(\mathbf{H}))} A(i,j) x_n(j)$.
  3. Send $\tilde{x}_n(i)$ to all neighbors $k \in B(i, \omega(\mathbf{H}))\backslash\{i\}$ andreceive $\tilde{x}_n(k)$ from neighbors $k \in B(i, \omega(\mathbf{H}))\backslash\{i\}$.
  4. Evaluate $\widehat{x}_n(i) = \sum_{j \in B(i,\omega(\mathbf{H}))} \tilde{A}(j,i)\tilde{x}_n(j)$.
  5. Set $x_{n+1}(i) = x_n(i) - \widehat{x}_n(i)$ and $n = n+1$.
  6. Return to Step 1 if $n \le M$, go to Output otherwise.

Output: $u(i) \approx x_M(i)$, where $\mathbf{u} = (u(i))_{i \in V}$ is the eigenvector.
```

plemented distributively and synchronously at the vertex level, see Algorithm 2.1, and the preconditioning matrix could be explicitly constructed at the vertex level (and hence on SDNs with communication range $L \ge \omega(\mathbf{A})$). For the implementation of Algorithm 1 on the network, every vertex i is **only** required to have the information of its $\omega(\mathbf{A})$-hop neighbors, it is equipped direct communication links with its $\omega(H)$-hop neighbors, and it needs memory to store the eigenvalue λ, the iteration number M, the ith diagonal entry of the preconditioning matrix and entries in the ith row and column of the matrix $\mathbf{A}$. Therefore the memory, computational cost and communication expense at each vertex to implement the preconditioned gradient descent algorithm are **independent** on the order of the graph $\mathcal{G}$, while they depend on the order of the graph in the distributed algorithms based on the consensus approach almost linearly [3,22] and quadratically [29,33,34]. We believe that the proposed PGDA and its symmetric version in Section 3 are important for the scenario that each agent of the network has limited memory, computing power and communication bandwidth, and also limited access to the whole matrix due to various reasons (such as privacy and limitation of storage), and the only global information available is the eigenvalue.

In Sections 3, we propose the symmetric preconditioned gradient descent algorithm (SPGDA), a symmetric version of the PGDA in Section 2, to find principal eigenvectors of a Hermitian matrix. Similar to the PGDA, it can be implemented distributively and synchronously at the vertex level, see Algorithm 3.1. Moreover, comparing with Algorithm 1 to find eigenvectors of an arbitrary matrix, the Algorithm 3.1 has less computational cost and communication expense in each iteration and our numerical simulations also indicate that it may converge faster. In Section 4, we modify the PGDA and SPGDA to find eigenvectors of a polynomial filter of graph shifts, which can be implemented at the vertex level with each agents being required to exchange data between **adjacent** vertices only. In Section 5, we demonstrate the performance of the proposed algorithms on finding eigenvectors associated with the (second) largest eigenvalue of lowpass-spline-like filters and compare their performance with the gradient descent method with optimal step size and the conventional power method without normalization.

Notation: Set $\|\mathbf{x}\|_2 = (\sum_{j \in V} |x(j)|^2)^{1/2}$ for graph signals $\mathbf{x} = (x_j)_{j \in V}$, denote the set of all s-hop neighbors of a vertex $i \in V$ by $B(i,s) = \{j \in V, \ \rho(j,i) \le s\}, s \ge 0$, denote the identity matrix by $\mathbf{I}$ and Hermitian transpose of a complex-valued matrix $\mathbf{A}$ by $\mathbf{A}^*$, and let $\lambda_{\max}(\mathbf{A}), \lambda_{\max,2}(\mathbf{A}), \lambda_{\min}(\mathbf{A})$ and $\lambda_{\min,0}(\mathbf{A})$ be the maximal eigenvalue, second largest eigenvalue, minimal eigenvalue and minimal nonzero eigenvalue of a positive semidefinite matrix $\mathbf{A}$ respectively.

2. A distributed iterative algorithm for determining eigenvectors

Let $\mathcal{G} = (V, E)$ be an undirected and unweighted graph of order N. For a graph matrix $\mathbf{A} = (A(i,j))_{i,j \in V}$ with geodesic-width $\omega(\mathbf{A})$, we define the diagonal matrix $\mathbf{P_A}$ with diagonal elements

$$P_{\mathbf{A}}(i,i) := \max_{k \in B(i,\omega(\mathbf{A}))} \Big\{ \max\Big(\sum_{j \in B(k,\omega(\mathbf{A}))} |A(j,k)|, \sum_{j \in B(k,\omega(\mathbf{A}))} |A(k,j)| \Big) \Big\}, \ i \in V \tag{2.1}$$

Cheng et al. [16]. Denote the standard inner product on $\mathbb{C}^N$ by $\langle \cdot, \cdot \rangle$. In this section, we consider the problem of finding eigenvectors of a complex-valued matrix associated with **any** of its eigenvalues.

Theorem 2.1. *Let* $\mathbf{A}$ *be a complex-valued matrix on the graph* $\mathcal{G}$ *of order* N, $\mathbf{P_A}$ *be the diagonal matrix in (2.1), and* $\mathbf{Q}$ *be a nonsingular diagonal matrix such that*

$$(\mathbf{Q} - \mathbf{P_A}) \text{ is positive semidefinite.} \tag{2.2}$$

Then for $\mathbf{x}_0 \in \mathbb{C}^N$, *the sequence* $\mathbf{x}_n$ *defined inductively by*

$$\mathbf{x}_{n+1} = (\mathbf{I} - \mathbf{Q}^{-2}\mathbf{A}^*\mathbf{A})\mathbf{x}_n, \ \ n \ge 0, \tag{2.3}$$

converges exponentially to some vector $\mathbf{u}$ *being either the zero vector or an eigenvector associated with the zero eigenvalue of the matrix* $\mathbf{A}$,

$$\|\mathbf{Q}(\mathbf{x}_n - \mathbf{u})\|_2 \le \|\mathbf{Q}\mathbf{x}_0\|_2 r^n, \quad n \ge 0, \tag{2.4}$$

where $\lambda_i(\mathbf{B}), 1 \le i \le N$, *are eigenvalues of the Hermitian matrix* $\mathbf{B} = \mathbf{I} - \mathbf{Q}^{-1}\mathbf{A}^*\mathbf{A}\mathbf{Q}^{-1}$, $\mathbf{u}_i, 1 \le i \le N$ *are orthonormal eigenvectors associated with eigenvalues* $\lambda_i(\mathbf{B})$,

$$\mathbf{B}\mathbf{u}_i = \lambda_i(\mathbf{B})\mathbf{u}_i, \quad 1 \le i \le N, \tag{2.5}$$

the convergence rate r *is given by*

$$r := \max_{0 \le \lambda_i(\mathbf{B}) < 1} \lambda_i(\mathbf{B}) \in [0, 1), \tag{2.6}$$

and the limit vector $\mathbf{u}$ *is defined by*

$$\mathbf{u} = \sum_{\lambda_i(\mathbf{B}) = 1} \langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i \rangle \mathbf{Q}^{-1}\mathbf{u}_i. \tag{2.7}$$

Proof. Following the argument in [16, Theorem II.1] and applying (2.2), we obtain that $\mathbf{Q}^2 - \mathbf{A}^*\mathbf{A}$ is positive semidefinite,

$$\mathbf{A}^*\mathbf{A} \preceq \mathbf{P}_{\mathbf{A}}^2 \preceq \mathbf{Q}^2, \tag{2.8}$$

where the first inequality in the real setting is established in [16, Theorem II.1] and the second inequality follows from (2.2). This together with nonsingularity of the matrix $\mathbf{Q}$ implies that eigenvalues $\lambda_i(\mathbf{B}), 1 \le i \le N$, of the matrix $\mathbf{B}$ are contained in the interval $[0, 1]$. Therefore the convergence rate r in (2.6) satisfies

$$r = \max_{\lambda_i(\mathbf{B}) \ne 1} \lambda_i(\mathbf{B}) \in [0, 1).$$

Write $\mathbf{Q}\mathbf{x}_0 = \sum_{i=1}^N \langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i \rangle \mathbf{u}_i$. By (2.3), we have that

$$\mathbf{Q}\mathbf{x}_n = (\mathbf{Q} - \mathbf{Q}^{-1}\mathbf{A}^*\mathbf{A})\mathbf{x}_{n-1} = (\mathbf{I} - \mathbf{Q}^{-1}\mathbf{A}^*\mathbf{A}\mathbf{Q}^{-1})\mathbf{Q}\mathbf{x}_{n-1} = \mathbf{B}\mathbf{Q}\mathbf{x}_{n-1}, \ \ n \ge 1,$$

where the first equality follows from (2.3). This together with (2.5) implies that

$$\mathbf{Q}\mathbf{x}_n = \mathbf{B}^n\mathbf{Q}\mathbf{x}_0 = \sum_{i=1}^N (\lambda_i(\mathbf{B}))^n \langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i \rangle \mathbf{u}_i, \ \ n \ge 0. \tag{2.9}$$

Combining (2.7), (2.9) and the orthonormality of $\mathbf{u}_i, 1 \le i \le N$, we obtain

$$\begin{aligned}\|\mathbf{Q}(\mathbf{x}_n-\mathbf{u})\|_2 &= \Big\| \sum_{0\le\lambda_i(\mathbf{B})<1} (\lambda_i(\mathbf{B}))^n \langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i\rangle \mathbf{u}_i \Big\| \\ &= \Big(\sum_{0\le\lambda_i(\mathbf{B})<1} |\langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i\rangle|^2 (\lambda_i(\mathbf{B}))^{2n} \Big)^{1/2} \\ &\le \|\mathbf{Q}\mathbf{x}_0 - \mathbf{Q}\mathbf{u}\|_2 r^n \le \|\mathbf{Q}\mathbf{x}_0\|_2 r^n, \ \ n \ge 0.\end{aligned}$$

This proves (2.4) and the desired exponential convergence of the sequence $\mathbf{x}_n, n \ge 0$.

Taking the limit in (2.3) and applying the convergence in (2.4) yields $\mathbf{Q}^{-2}\mathbf{A}^*\mathbf{A}\mathbf{u} = \mathbf{0}$. This proves the desired conclusion that $\mathbf{u}$ in (2.7) is either the zero vector or an eigenvector associated with eigenvalue zero. □

Take a positive constant $c > 0$ and define the diagonal matrix $\mathbf{Q}_c = \mathrm{diag}(Q_c(i,i))_{i\in V}$ by

$$Q_c(i,i) = \max\big(P_{\mathbf{A}}(i,i), c\big), \ \ i \in V. \tag{2.10}$$

Then $\mathbf{Q}_c$ is a nonsingular diagonal matrix satisfying (2.2), it has condition number

$$\kappa(\mathbf{Q}_c) := \frac{\max_{i\in V} Q_c(i,i)}{\min_{i\in V} Q_c(i,i)} = \frac{\max\big(c, \max_{i\in V} P_{\mathbf{A}}(i,i)\big)}{\max\big(c, \min_{i\in V} P_{\mathbf{A}}(i,i)\big)}, \tag{2.11}$$

and it can be constructed at the vertex level (and hence on SDNs with communication range $L \ge \omega(\mathbf{A})$), since diagonal entries $P_{\mathbf{A}}(i,i), i \in V$, of the preconditioning matrix $\mathbf{P}_{\mathbf{A}}$ can, cf. [16, Algorithm II.1]. Let $\mathbf{e}_i, i \in V$, be the standard unit vector taking value zero except value one at ith component. Under the assumption that $\mathbf{A} = (A(i,j))_{i,j\in V}$ is nonsingular, one may verify that

$$P_{\mathbf{A}}(i,i) \ge \sum_{j\in B(i,\omega(\mathbf{A}))} |A(i,j)| \ge \|\mathbf{A}\mathbf{e}_i\|_2 = \Big\| \mathbf{A}^{-1} \frac{\mathbf{A}\mathbf{e}_i}{\|\mathbf{A}\mathbf{e}_i\|_2} \Big\|_2^{-1} \ge \Big(\sup_{\|\mathbf{u}\|_2=1} \|\mathbf{A}^{-1}\mathbf{u}\|_2 \Big)^{-1}$$

and hence for all $0 \le c \le (\sup_{\|\mathbf{u}\|_2=1} \|\mathbf{A}^{-1}\mathbf{u}\|_2)^{-1}$, the preconditioning matrices $\mathbf{Q}_c$ in (2.10) are the same. We remark that the preconditioning matrix $\mathbf{Q}_c$ with $c = 0$ is used in [16] to solve the linear system $\mathbf{A}\mathbf{x} = \mathbf{y}$ in a distributed manner, where $\mathbf{A}$ is assumed to be nonsingular, see [16,34] and references therein for distributed algorithms to solving a linear system on graphs.

Remark 2.2. In Theorem 2.1, we do not assume that zero is the eigenvalue of the matrix $\mathbf{A}$ with multiplicity one. For the case that the eigenspace associated with the zero eigenvalue of the matrix $\mathbf{A}$ is a one-dimensional space spanned by a unit vector $\mathbf{w}$, the eigenvector $\mathbf{u}_i$ in (2.5) associated with the eigenvalue 1 of the matrix $\mathbf{B}$ is given by $\mathbf{u}_i = \mathbf{Q}\mathbf{w}/\|\mathbf{Q}\mathbf{w}\|_2$. Hence it follows from (2.7) and $\mathbf{Q}\mathbf{x}_0 = \sum_{i=1}^N \langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i\rangle \mathbf{u}_i$ that the limit of the sequence $\mathbf{x}_n, n \ge 0$, is given by $\mathbf{u} = \langle \mathbf{Q}\mathbf{x}_0, \mathbf{Q}\mathbf{w}\rangle \mathbf{w}/\|\mathbf{Q}\mathbf{w}\|_2^2$.

Remark 2.3. Let $\mathbf{H} = (H(i,j))_{i,j\in V}$ be a matrix with geodesic-width $\omega(\mathbf{H})$ and $\lambda \in \mathbb{C}$ be its eigenvalue. By selecting a random initial $\mathbf{x}_0$ with entries i.i.d. on $[0,1]$, and applying the iterative algorithm (2.3) to the matrix $\mathbf{A} = \mathbf{H} - \lambda\mathbf{I}$ or $\lambda\mathbf{I} - \mathbf{H}$, we obtain from Theorem 2.1 that the limit of the sequence $\mathbf{x}_n, n \ge 0$, is the nonzero vector $\mathbf{u}$ in (2.7) (and hence an **eigenvector** associated with the given eigenvalue λ) with probability one, where $\mathbf{u}$ is not a zero vector with probability one because for the selection of the random initial $\mathbf{x}_0$, the event $\langle \mathbf{Q}\mathbf{x}_0, \mathbf{u}_i\rangle = \langle \mathbf{x}_0, \mathbf{Q}\mathbf{u}_i\rangle = 0$ happens with probability zero. Following the terminology in [16] where $\mathbf{Q} = \mathbf{P}_{\mathbf{A}}$, we call the above algorithm (2.3) to find eigenvectors associated with a given eigenvalue as a *preconditioned gradient descent algorithm*, PGDA for abbreviation. By Theorem 2.1, the proposed PGDA converges exponentially with the convergence rate

$$r_{\mathrm{PGDA}} = \max_{0\le\lambda_i<1} \lambda_i, \tag{2.12}$$

where $\lambda_i, 1 \le i \le N$, are eigenvalues of the Hermitian matrix $\mathbf{I} - \mathbf{Q}(\lambda\mathbf{I} - \mathbf{H})^*(\lambda\mathbf{I} - \mathbf{H})\mathbf{Q}^{-1}$. Moreover, the proposed PGDA can be implemented **distributively** and **synchronously** at the vertex level, see Algorithm 1. For the implementation of Algorithm 1, every vertex $i \in V$ is required to have the information of its $\omega(\mathbf{H})$-hop neighbors, equipped direct communication links with its $\omega(\mathbf{H})$-hop neighbors, and needs memory to store the eigenvalue λ, the iteration number M, the ith diagonal entry of the matrix $\mathbf{Q}$, and entries $H(i,j)$ and $H(j,i), j \in B(i,\omega(\mathbf{H}))$ in the ith row and column of the matrix $\mathbf{H}$. Moreover, it is observed that the computational cost and communication expense for each vertex are independent on the order N of the graph $\mathcal{G}$, cf. [15] for the detailed explanation.

Remark 2.4. The preconditioning matrix $\mathbf{Q}$ in (2.3) is closely related to the conventional gradient descent algorithm [10,13,15,16,37]. Replacing the preconditioning matrix $\mathbf{Q}$ in (2.3) by $\gamma^{-1/2}\mathbf{I}$ with step size $\gamma > 0$, the corresponding PGDA becomes the conventional gradient descent algorithm, GDA for abbreviation,

$$\mathbf{x}_{n+1} = (\mathbf{I} - \gamma\mathbf{A}^*\mathbf{A})\mathbf{x}_n, \ \ n \ge 0, \tag{2.13}$$

which has been widely used for inverse filtering [10,13,15,16,30,37]. Applying the similar argument used to prove Theorem 2.1, one may show that the sequence $\mathbf{x}_n, n \ge 0$, in (2.13) with step size $\gamma \in (0, 2/\lambda_{\max}(\mathbf{A}^*\mathbf{A}))$ converges exponentially to some vector $\mathbf{u}$ satisfying $\mathbf{A}\mathbf{u} = \mathbf{0}$, and the convergence rate is $\max(|1 - \gamma\lambda_{\min,0}(\mathbf{A}^*\mathbf{A})|, |1 - \gamma\lambda_{\max}(\mathbf{A}^*\mathbf{A})|)$. Therefore the optimal step size in the GDA (2.13) is

$$\begin{aligned}\gamma_{\mathrm{OpGDA}} :&= \frac{2}{\lambda_{\min,0}(\mathbf{A}^*\mathbf{A}) + \lambda_{\max}(\mathbf{A}^*\mathbf{A})} \\ &= \arg\min_{\gamma} \max(|1 - \gamma\lambda_{\min,0}(\mathbf{A}^*\mathbf{A})|, |1 - \gamma\lambda_{\max}(\mathbf{A}^*\mathbf{A})|)\end{aligned}$$

and the corresponding convergence rate is

$$r_{\mathrm{OpGDA}} = \frac{\lambda_{\max}(\mathbf{A}^*\mathbf{A}) - \lambda_{\min,0}(\mathbf{A}^*\mathbf{A})}{\lambda_{\min,0}(\mathbf{A}^*\mathbf{A}) + \lambda_{\max}(\mathbf{A}^*\mathbf{A})} \tag{2.14}$$

Shi et al. [48]. We call the above GDA with optimal step size γ_{OpGDA} by the *optimal gradient descent algorithm* and use the notion OpGDA for abbreviation, see Section 5 for numerical demonstrations.

2.1. Principal eigenvectors of the hyperlink matrix

Principal eigenvectors of a left stochastic matrix on a network have been used to measure the influence of a vertex on the whole network, and a node with the larger entry value has a greater influence on the network which has been used in Google's page ranking algorithm and some other applications, see [1,6,35,36] and references therein. The left stochastic matrix has 1 as the leading eigenvalue and the principal eigenvector associated with the eigenvalue 1 has positive entries by Perron-Frobenius theorem. The conventional approach to find principal eigenvectors of a left stochastic matrix $\mathbf{A}$ is the power iteration method, Power for abbreviation,

$$\mathbf{x}_n = \mathbf{A}\mathbf{x}_{n-1}, n \ge 1, \tag{2.15}$$

with the initial $\mathbf{x}_0$ has positive entries, c.f. Remark 3.3 for power iteration method for positive definite matrices. One may verify that the convergence rate r_{Power} of the power iteration method is the largest magnitude of non-one eigenvalues λ_i of the matrix $\mathbf{A}$,

$$r_{\mathrm{Power}} = \max_{\lambda_i \ne 1} |\lambda_i|.$$

Let $\mathbf{H} = \mathbf{W}\mathbf{D}^{-1}$ be the hyperlink matrix on a network described by a weighted (un)directed graph $\mathcal{G} = (V, E, \mathbf{W})$, where $\mathbf{W} = (w(i,j))_{i,j\in V}$ is the weight and $\mathbf{D} = \mathrm{diag}(d_i)_{i\in V}$ is the in-degree matrix given by $d_i = \sum_{j\in V} w(j,i), i \in V$. The matrix $\mathbf{H}$ is a left stochastic matrix with 1 being the leading eigenvalue and hence the principal eigenvectors associated with the eigenvalue 1 have positive

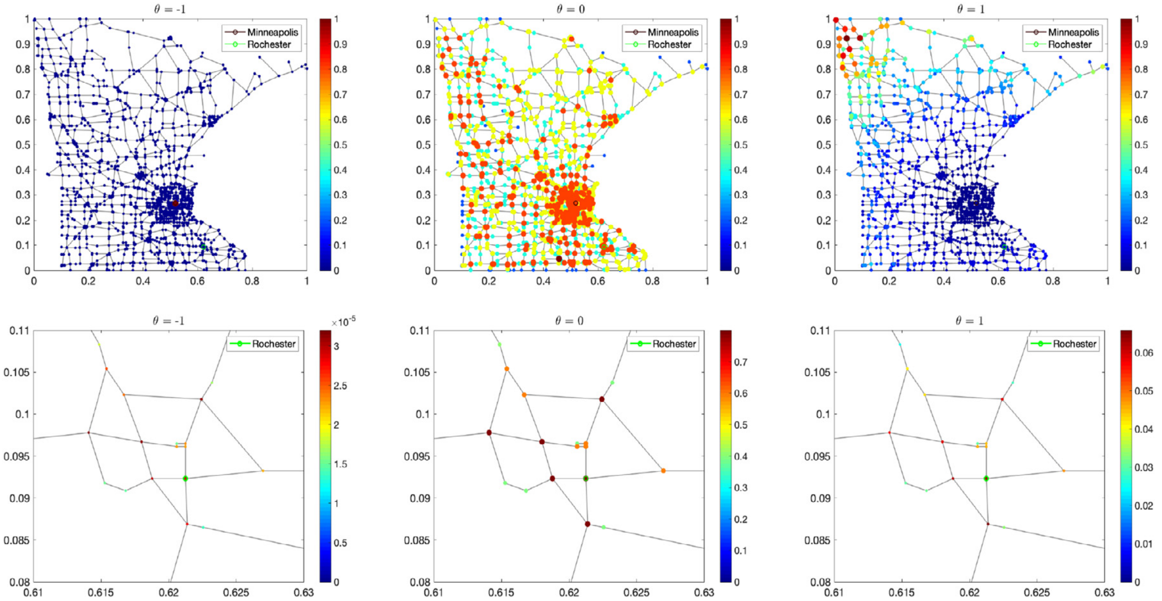

Fig. 1. Plotted from the left to the right on the top row are principal eigenvectors of the hyperlink matrix $\mathbf{H}_{\theta,\mathbf{z}_0}$ for $\theta = -1, 0, 1$ respectively, where the principal eigenvectors are normalized so that the maximal entry takes value 1. Plotted from the left to the right on the bottom row are entries of the principal eigenvectors in a neighborhood around downtown Rochester at position $(0.6212, 0.0923)$.

entries. For the case that the graph $\mathcal{G}$ is connected and undirected, one may verify that the principal eigenvectors associated with the eigenvalue 1 are proportional to $\mathbf{D1}$, where $\mathbf{1}$ is the vector with all entries taking value 1. The distributed PGDA proposed in this section is an applicable distributed algorithm to evaluate the principal eigenvectors of the hyperlink matrix $\mathbf{H}$ and hence identify the global influence of a vertex in the whole network and also local influence of a vertex on its neighborhood.

Take $\theta \in [-2, 2]$ and $\mathbf{z}_0 \in [0, 1]^2$. Let $\mathcal{M}_{\theta,\mathbf{z}_0} = (V, E, \mathbf{W}_{\theta,\mathbf{z}_0})$ be the Minnesota traffic graph of order $N = 2642$ with vertices $i \in V$ being deployed on the unit square $[0, 1]^2$ [18,45], and the weighted adjacency matrix $\mathbf{W}_{\theta,\mathbf{z}_0} := (w_{\theta,\mathbf{z}_0}(i, j)a(i, j))_{i,j\in V}$ being scaled from the unweighted adjacency matrix $\mathbf{A} = (a(i, j))_{i,j\in V}$ on the Minnesota traffic graph, where $(x_i, y_i), i \in V$, are the coordinates of the vertex $i \in V$, and the weights

$$w_{\theta,\mathbf{z}_0}(i, j) = (1 + N\|(x_i, y_i) - \mathbf{z}_0\|_2)^{\theta}(1 + N\|(x_j, y_j) - \mathbf{z}_0\|_2)^{\theta}, i, j \in V,$$

depend on the distances between the location of vertices $i, j \in V$ and the given center $\mathbf{z}_0$. Shown in Fig. 1 are the principal eigenvectors of the hyperlink matrix

$$\mathbf{H}_{\theta,\mathbf{z}_0} = \mathbf{W}_{\theta,\mathbf{z}_0}(\mathbf{D}_{\theta,\mathbf{z}_0})^{-1}$$

on the weighted undirected Minnesota traffic graph $\mathcal{M}_{\theta,\mathbf{z}_0}$, where $\mathbf{D}_{\theta,\mathbf{z}_0}$ is the degree matrix and the coordinate $(0.5185, 0.2675)$ of a vertex representing a traffic intersection in downtown Minneapolis is selected as the center $\mathbf{z}_0$ of the weights $w_{\theta,\mathbf{z}_0}(i, j), i, j \in V$. As $\mathbf{H}_{\theta,\mathbf{z}_0}\mathbf{D}_{\theta,\mathbf{z}_0}\mathbf{1} = \mathbf{D}_{\theta,\mathbf{z}_0}\mathbf{1}$, principal eigenvectors of the hyperlink matrix $\mathbf{H}_{\theta,\mathbf{z}_0}$ are proportional to the degree vector $\mathbf{D}_{\theta,\mathbf{z}_0}\mathbf{1}$. The above conclusion is confirmed from Fig. 1 that the large entries of the principal eigenvectors are mainly concentrated in a neighborhood of the chosen center $\mathbf{z}_0$ for small $\theta < 0$, and far away from the chosen center $\mathbf{z}_0$ for large $\theta > 0$. Shown in the second row of Fig. 1 is the local influence of an intersection at downtown Rochester to its neighborhood. We observe that its local influence to intersections in a small neighborhood does not change much for $-1 \le \theta \le 1$, even its global influence to the whole traffic network changes dramatically for $-1 \le \theta \le 1$.

We compare the performance of the conventional power iteration method (2.15), the proposed PGDA and the OpGDA in Remark 2.4 to find principal eigenvectors of the hyperlink matrix $\mathbf{H}_{\theta,\mathbf{z}_0}, -2 \le \theta \le 2$, see Fig. 2. It is observed that the proposed PGDA has better performance than the conventional power iteration method (Power), the OpGDA has better performance than the proposed PGDA does, while their performances are still comparable. We remark that the PGDA and Power can be implemented in a distributed manner, while the OpGDA is not always considered as a distributed algorithm since one may need some global information to determine the optimal step size.

3. Principal eigenvectors of Hermitian matrices

In this section, we consider finding principal eigenvectors associated with the minimal/maximal eigenvalue of a Hermitian matrix on a graph $\mathcal{G} = (V, E)$ of order N in a distributed manner. We propose a distributed algorithm to find principal eigenvectors, which converges faster and has less computational and communication expense in each iteration than Algorithm 1 to find eigenvectors of an arbitrary matrix does.

Theorem 3.1. *Let $\mathbf{A} = (A(i, j))_{i,j\in V}$ be a positive semidefinite matrix on the graph $\mathcal{G}$ of order N with geodesic-width $\omega(\mathbf{A})$, and $\mathbf{Q}^{\rm sym} = {\rm diag}(Q^{\rm sym}(i, i))_{i\in V}$ be a nonsingular diagonal matrix satisfying*

$$Q^{\rm sym}(i, i) \ge \sum_{j\in B(i,\omega(\mathbf{A}))} |A(i, j)|, \ \ i \in V. \tag{3.1}$$

Then for any initial $\mathbf{x}_0 \in \mathbb{C}^N$, the sequence $\mathbf{x}_n$ defined by

$$\mathbf{x}_{n+1} = (\mathbf{I} - (\mathbf{Q}^{\rm sym})^{-1}\mathbf{A})\mathbf{x}_n, \ \ n \ge 0, \tag{3.2}$$

converges exponentially to either the zero vector or an eigenvector $\mathbf{u}$ associated with the zero eigenvalue of the matrix $\mathbf{A}$.

Proof. Set $\mathbf{B}^{\rm sym} = \mathbf{I} - (\mathbf{Q}^{\rm sym})^{-1/2}\mathbf{A}(\mathbf{Q}^{\rm sym})^{-1/2}$. Following the argument in [16, Theorem III.1] and applying (3.1), we obtain that $\mathbf{Q}^{\rm sym} - \mathbf{A}$ is positive semidefinite. This together with the positive semidefiniteness of the matrix $\mathbf{A}$ implies that all eigenvalues of the Hermitian matrix $\mathbf{B}^{\rm sym}$ are in the unit interval $[0, 1]$. Applying similar argument used in the proof of Theorem 2.1 with $\mathbf{Q}$ and $\mathbf{A}^*\mathbf{A}$ replaced by $(\mathbf{Q}^{\rm sym})^{1/2}$ and $\mathbf{A}$ respectively, we obtain

$$\|(\mathbf{Q}^{\rm sym})^{1/2}(\mathbf{x}_n - \mathbf{u})\|_2 \le \|(\mathbf{Q}^{\rm sym})^{1/2}\mathbf{x}_0\|_2 r^n, \quad n \ge 0 \tag{3.3}$$

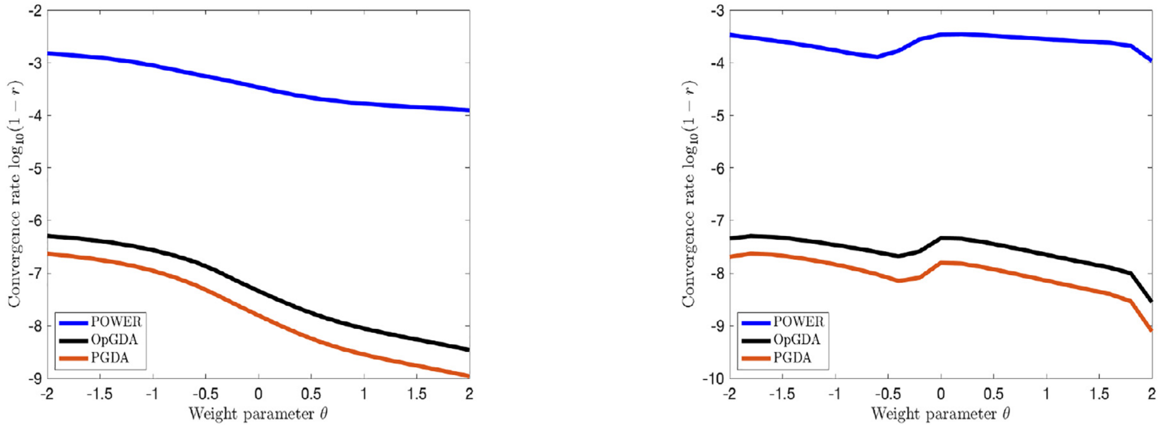

Fig. 2. Plotted are the convergence rate r in the logarithmic scale $\log_{10}(1-r)$ for the Power, OpGDA and PGDA, see (2.15), (2.14) and (2.12), to find the principal eigenvectors of the hyperlink matrices $\mathbf{H}_{\theta,\mathbf{z}_0}, -2\le\theta\le 2$, on the weighted Minnesota traffic graphs $\mathcal{M}_{\theta,\mathbf{z}_0}=(V,E,\mathbf{W}_{\theta,\mathbf{z}_0})$, where the center $\mathbf{z}_0$ of the weights $w_{\theta,\mathbf{z}_0}(i,j), i,j\in V$ on the left and right figures are traffic intersections in downtown Minneapolis at position $\mathbf{z}_0=(0.5185, 0.2675)$ and Duluth at position $\mathbf{z}_0=(0.6655, 0.5936)$ respectively.

for some vector $\mathbf{u}\in\mathbb{C}^N$, where r is the largest eigenvalue of $\mathbf{B}^{\rm sym}$ in $[0,1)$. This together with the nonsingularity of the matrix $\mathbf{Q}^{\rm sym}$ proves the exponential convergence of $\mathbf{x}_n, n\ge 0$.

Taking limit in (3.2) proves $\mathbf{Au}=\mathbf{0}$, and hence completes the proof. □

We remark that the normalized matrix $\mathbf{Q}^{\rm sym}$ in (3.1) associated with a diffusion matrix has been used to understand diffusion process [38], and the one corresponding to the Laplacian $\mathbf{L}$ on the connected graph is twice of the degree matrix $\mathbf{D}$ [16,39]. For a positive semidefinite matrix $\mathbf{A}=(A(i,j))_{i,j\in V}$ with geodesic-width $\omega(\mathbf{A})$, a nonsingular diagonal matrix $\mathbf{Q}_c^{\rm sym}=\mathrm{diag}(Q_c^{\rm sym}(i,i))_{i\in V}$ satisfying (3.1) can be constructed at the vertex level by setting

$$Q_c^{\rm sym}(i,i)=\max\Big(\sum_{j\in B(i,\omega(\mathbf{A}))}|A(i,j)|, c\Big),\ \ i\in V, \tag{3.4}$$

where c is a positive constant, cf. (2.10). One may verify that the above preconditioning matrix has its condition number given by

$$\kappa\left(\mathbf{Q}_c^{\rm sym}\right)=\frac{\max_{i\in V}Q_c^{\rm sym}(i,i)}{\min_{i\in V}Q_c^{\rm sym}(i,i)}=\frac{\max\left(c, \max_{i\in V}\sum_{j\in B(i,\omega(\mathbf{A}))}|A(i,j)|\right)}{\max\left(c, \min_{i\in V}\sum_{j\in B(i,\omega(\mathbf{A}))}|A(i,j)|\right)}. \tag{3.5}$$

With the above selection of the preconditioning matrix in (3.2), we can find eigenvectors associated with minimal/maximal eigenvalues of a Hermitian matrix by the distributed iterative algorithm (3.2) implementable at the vertex level, see Algorithm 2. Following the terminology in [16], we call the algorithm (3.2) with a random initial having entries i.i.d on $[0,1]$ as a *symmetric preconditioned gradient descent algorithm*, SPGDA for abbreviation. By (3.3), the proposed SPGDA converges exponentially with the convergence rate

$$r_{\rm SPGDA}=\max_{0\le\tilde\lambda_i<1}\tilde\lambda_i, \tag{3.6}$$

where $\tilde\lambda_i, 1\le i\le N$, are eigenvalues of the Hermitian matrix $\mathbf{I}-(\mathbf{Q}^{\rm sym})^{1/2}\mathbf{A}(\mathbf{Q}^{\rm sym})^{-1/2}$. Similar to the PGDA, we can apply Algorithm 2 to implement the proposed SPGDA **distributively** and **synchronously**. Moreover, comparing with Algorithm 1 to find eigenvectors of an arbitrary matrix, the Algorithm 2 to find principal eigenvectors of a Hermitian matrix has less computational cost and communication expense in each iteration. Our numerical simulations in Section 5 also indicate that it may converge faster.

Let $\mathbf{H}$ be a Hermitian matrix with minimal eigenvalue $\lambda_{\min}(\mathbf{H})$ and maximal eigenvalue $\lambda_{\max}(\mathbf{H})$. Then $\mathbf{A}_1=\mathbf{H}-\lambda_{\min}(\mathbf{H})\mathbf{I}$ and $\mathbf{A}_2=\lambda_{\max}(\mathbf{H})\mathbf{I}-\mathbf{H}$ have eigenvalue zero and they are positive semidefinite. Then applying the SPGDA to $\mathbf{A}_1$ (resp. $\mathbf{A}_2$) with a random initial $\mathbf{x}_0$ having entries i.i.d on $[0,1]$, we obtain the **principal** eigenvectors associated with minimal (resp. maximal) eigenvalues of the Hermitian matrix $\mathbf{H}$ by Theorem 3.1.

Algorithm 2 Realization of the SPGDA at a vertex $i\in V$.

```
Inputs: The totaliteration number $M$,the geodesic-width $\omega(\mathbf{A})$ of the positive semidefinite matrix $\mathbf{A}$,the set $B(i,\omega(\mathbf{A}))$ of $\omega(\mathbf{A})$-hop neighbors of the vertex $i$,entries $A(i,j), j\in B(i,\omega(\mathbf{A}))$, in the $i$th row of the matrix $\mathbf{A}$,and the $i$th entry $Q^{\rm sym}(i,i)$ of the diagonal matrix $\mathbf{Q}^{\rm sym}$.
Initial:Select$x_0(i)$ randomly in $[0,1]$,and set $n=0$.
Iteration:
    1. Send $x_n(i)$ to all neighbors $k\in B(i,\omega(\mathbf{A}))\backslash\{i\}$and receive $x_n(k)$ from neighbors $k\in B(i,\omega(\mathbf{A}))\backslash\{i\}$.
    2. Evaluate $x_{n+1}(i)=x_n(i)-\sum_{j\in B(i,\omega(\mathbf{A}))}(Q^{\rm sym}(i,i))^{-1}A(i,j)x_n(j)$ and set $n=n+1$.
    3. Return to Step 1 if $n\le M$, otherwise go to Output.
Output: $u(i)\approx y_M(i)$, where $\mathbf{u}=(u(i))_{i\in V}$.
```

Remark 3.2. Replacing the preconditioning matrix $\mathbf{Q}^{\rm sym}$ in (3.2) by $(\gamma^{\rm sym})^{-1}\mathbf{I}$ with step size $\gamma^{\rm sym}>0$, the corresponding SPGDA becomes

$$\mathbf{x}_{n+1}=(\mathbf{I}-\gamma^{\rm sym}\mathbf{A})\mathbf{x}_n, n\ge 0, \tag{3.7}$$

the symmetric version of the gradient descent algorithm (2.13) with step size $\gamma^{\rm sym}>0$ [15,16]. By a similar proof of Theorem 3.1, we can show that the above sequence $\mathbf{x}_n, n\ge 0$, with step size $\gamma^{\rm sym}\in(0, 2/\lambda_{\max}(\mathbf{A}))$ converges exponentially to some vector $\mathbf{u}$ satisfying $\mathbf{Au}=\mathbf{0}$ and the convergence rate is $\max(|1-\gamma\lambda_{\min,0}(\mathbf{A})|, |1-\gamma\lambda_{\max}(\mathbf{A})|)$. Moreover, the optimal step size is $\gamma_{\rm op}^{\rm sym}=2/(\lambda_{\min,0}(\mathbf{A})+\lambda_{\max}(\mathbf{A}))$ and the corresponding convergence rate is

$$r_{\rm OpSGDA}=\frac{\lambda_{\max}(\mathbf{A})-\lambda_{\min,0}(\mathbf{A})}{\lambda_{\min,0}(\mathbf{A})+\lambda_{\max}(\mathbf{A})}. \tag{3.8}$$

We call the symmetric gradient descent algorithm (3.7) with the optimal step size $\gamma_{\rm op}^{\rm sym}$ by the *optimal symmetric gradient descent algorithm* and use the notion OpSGDA for abbreviation, see Section 5 for numerical demonstrations.

Remark 3.3. For a positive semidefinite matrix $\mathbf{H}$ with maximal eigenvalue $\lambda_{\max}(\mathbf{H})$, applying (3.7) with $\mathbf{A}$ and $\gamma^{\rm sym}$ replaced by

$\lambda_{\max}(\mathbf{H})\mathbf{I}-\mathbf{H}$ and $(\lambda_{\max}(\mathbf{H}))^{-1}$ respectively yields

$$\mathbf{x}_{n+1}=(\lambda_{\max}(\mathbf{H}))^{-1}\mathbf{H}\mathbf{x}_n,\ \ n\ge 0. \tag{3.9}$$

The above sequence $\mathbf{x}_n, n\ge 0$, converges exponentially with the convergence rate

$$r_{\mathrm{CAPower}}=\frac{\lambda_{\max,2}(\mathbf{H})}{\lambda_{\max}(\mathbf{H})}, \tag{3.10}$$

and the limit $\mathbf{u}$ is a principal eigenvector of the matrix $\mathbf{H}$ associated with the eigenvalue $\lambda_{\max}(\mathbf{H})$ if $\mathbf{u}\ne\mathbf{0}$, where $\lambda_{\max,2}(\mathbf{H})$ is the second largest eigenvalue of the positive semidefinite matrix $\mathbf{H}$. The above approach can be considered as the conventional power iteration method without normalization [40] and its variations have been discussed in [22,41–43]. Following the terminology *Chaotic Asynchronous Power Iteration Algorithm* in [41], we use CAPower for abbreviation of the above approach with entries of the initial $\mathbf{x}_0$ randomly selected in $[0,1]$, see Section 5 for numerical demonstrations.

4. Eigenvectors of polynomial filters

Graph filter is a fundamental concept in graph signal processing and it has been used in many applications such as denoising, smoothing, and consensus of multi-agent systems [4,8,12,13,15,16,18,44–50]. An elementary graph filter is a *graph shift*, which has 1 as its geodesic-width. Graph filters in most of literature are designed to be polynomials

$$\mathbf{A}=h(\mathbf{S}_1,\ldots,\mathbf{S}_d)=\sum_{l_1=0}^{L_1}\cdots\sum_{l_d=0}^{L_d}h_{l_1,\ldots,l_d}\mathbf{S}_1^{l_1}\cdots\mathbf{S}_d^{l_d} \tag{4.1}$$

of commutative graph shifts $\mathbf{S}_1,\ldots,\mathbf{S}_d$, i.e., $\mathbf{S}_k\mathbf{S}_{k'}=\mathbf{S}_{k'}\mathbf{S}_k$ for all $1\le k,k'\le d$, where the multivariate polynomial

$$h(t_1,\ldots,t_d)=\sum_{l_1=0}^{L_1}\cdots\sum_{l_d=0}^{L_d}h_{l_1,\ldots,l_d}t_1^{l_1}\ldots t_d^{l_d}$$

of degree $\sum_{k=1}^d L_k$ has polynomial coefficients $h_{l_1,\ldots,l_d}$, $0\le l_k\le L_k, 1\le k\le d$ [10–14,18,51,52]. In this section, we propose iterative algorithms to determine eigenvectors associated with a polynomial graph filter, which can be implemented on an SDN with **1** as its communication range, i.e., direct communication exists between all adjacent vertices only.

Observe that

$$\mathbf{A}^*=\sum_{l_1=0}^{L_1}\cdots\sum_{l_d=0}^{L_d}\overline{h_{l_1,\ldots,l_d}}(\mathbf{S}_d^*)^{l_d}\cdots(\mathbf{S}_1^*)^{l_1} \tag{4.2}$$

is a polynomial graph filter of commutative shifts $\mathbf{S}_1^*,\ldots,\mathbf{S}_d^*$. Then applying [15, Algorithm II.2] to implement the filtering procedure associated with polynomial graph filters $\mathbf{A}$ and $\mathbf{A}^*$, and then applying the diagonal preconditioning matrices $\mathbf{Q}^{-2}$ and $(\mathbf{Q}^{\mathrm{sym}})^{-1}$, we can implement the iteration (2.3) of the PGDA and (3.2) of the SPGDA in finite steps with each step including data exchanging between adjacent vertices only. Using the above implementation, the PGDA and SPGDA can be applied to find eigenvectors of a polynomial graph filter on SDNs with communication range **one**.

Now it remains to construct diagonal matrices satisfying (2.2) and (3.1) on SDNs with communication range 1. For the polynomial graph filter $\mathbf{A}$ in (4.1), define diagonal matrices $\widehat{\mathbf{Q}}_c=\mathrm{diag}(\widehat{Q}_c(i,i))_{i\in V}$ and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}=\mathrm{diag}(\widehat{Q}_c^{\mathrm{sym}}(i,i))_{i\in V}$ by

$$\widehat{Q}_c(i,i)=\max_{\rho(j,i)\le\omega(\mathbf{A})}\max\Big\{\sum_{k\in V}\widehat{A}(j,k),\sum_{k\in V}\widehat{A}(k,j),c\Big\} \tag{4.3}$$

and

$$\widehat{Q}_c^{\mathrm{sym}}(i,i)=\max\Big\{\sum_{k\in V}\widehat{A}(i,k),c\Big\},\ \ i\in V, \tag{4.4}$$

where $c>0$ is a positive number, $|\mathbf{S}_k|=(|S_k(i,j)|)_{i,j\in V}, 1\le k\le d$, and

$$(\widehat{A}(i,j))_{i,j\in V}=:\widehat{\mathbf{A}}:=\sum_{l_1=0}^{L_1}\cdots\sum_{l_d=0}^{L_d}|h_{l_1,\ldots,l_d}|\ |\mathbf{S}_1|^{l_1}\cdots|\mathbf{S}_d|^{l_d}.$$

One may verify that

$$|A(i,j)|\le\widehat{A}(i,j)\ \text{for all}\ i,j\in V. \tag{4.5}$$

Therefore the matrices $\widehat{\mathbf{Q}}_c$ in (4.3) and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}$ in (4.4) satisfy (2.2) and (3.1) respectively. Moreover following [15, Algorithm II.2] to implement the filtering procedure, we can construct diagonal preconditioning matrices $\widehat{\mathbf{Q}}_c$ and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}$ at the vertex level in finite steps such that in each step, every vertex needs to exchange data with adjacent vertices only.

5. Numerical simulations

Let $\mathcal{G}_N, N\ge 2$, be random geometric graphs with N vertices whose coordinates $(x_i,y_i), 1\le i\le N$, randomly deployed on the unit square $[0,1]^2$ and an undirected edge between two vertices in V_N exists if their physical distance is not larger than $\sqrt{2/N}$ [18,53]. In this section, we consider finding eigenvectors associated with maximal eigenvalue 1 of graph matrices

$$\mathbf{H}_{\theta,m}=(\mathbf{I}-C_\theta\mathbf{W}_\theta\mathbf{L}^{\mathrm{sym}}\mathbf{W}_\theta)^m$$

of order $m\ge 1$, where $\theta\ge 0$ is a weight parameter, $\mathbf{L}^{\mathrm{sym}}$ is the symmetric normalized Laplacian matrix on the graph $\mathcal{G}_N$, $\mathbf{W}_\theta=\mathrm{diag}(w_\theta(i))$ is a diagonal weighted matrix with diagonal entries $w_\theta(i)=(1+N\|(x_i,y_i)-(1/2,1/2)\|_2)^\theta$, and $C_\theta=(1+N/\sqrt{2})^{-2\theta}/2$. The matrices $\mathbf{H}_{\theta,m}$ with $\theta=0$ are lowpass spline filters of $\mathbf{H}_{0,m}^{\mathrm{spln}}, m\ge 1$, introduced in [18,54] for nonsubsampled filter banks. In the simulations, we use PGDA and PGDA1h to denote the PGDA with $\mathbf{A}$ replaced by $\mathbf{I}-\mathbf{H}_{\theta,m}$ and $\mathbf{Q}$ by $\mathbf{Q}_c$ in (2.10) and $\widehat{\mathbf{Q}}_c$ in (4.3) respectively, and similarly we use SPGDA and SPGDA1h to denote the SPGDA with $\mathbf{A}$ replaced by $\mathbf{I}-\mathbf{H}_{\theta,m}$ and $\mathbf{Q}$ by $\mathbf{Q}_c^{\mathrm{sym}}$ in (3.4) and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}$ in (4.4) respectively. Our numerical simulations indicate that the iterative PGDA and SPGDA have slower convergence speed for larger thresholding constant $c>0$ and their exponential convergence rates do not change for small $c>0$, see Fig. 3. However we cannot confirm the above phenomenon mathematically. One may verify that the matrices $\mathbf{Q}_c^{\mathrm{sym}}$ in (3.4) and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}$ in (4.4) are nonsingular if $c>0$, and they could be singular if $c=0$ is selected, see (2.11) and (3.5) for the condition numbers of those preconditioning matrices. Based on the above observations, we take $c=0.001$ in our simulations from now on.

For the sequences $\mathbf{x}_n, n\ge 0$, in the PGDA, SPGDA, PGDA1h and SPGDA1h and their limits $\mathbf{u}$, define relative convergence errors

$$E(n)=\log_{10}(\|\mathbf{x}_n-\mathbf{u}\|_2/\|\mathbf{u}\|_2)$$

in the logarithmic scale to measure their performance. Shown in Fig. 4 is the average over 200 trials for the convergence rate with $0\le\theta\le 2$ and $m=1,4$, and the relative convergence errors $E(n)$ for $\theta=0,3/2$ and $m=4$, where the convergence rates r for the PGDA and PGDA1h are given in (2.12) with $\mathbf{A}$ replaced by $\mathbf{I}-\mathbf{H}_{\theta,m}$ and $\mathbf{Q}$ by $\mathbf{Q}_c$ in (2.10) and $\widehat{\mathbf{Q}}_c$ in (4.3) respectively, and similarly the convergence rates for the SPGDA and SPGDA1h are given in (3.6) with $\mathbf{A}$ replaced by $\mathbf{I}-\mathbf{H}_{\theta,m}$ and $\mathbf{Q}^{\mathrm{sym}}$ are replaced by $\mathbf{Q}_c^{\mathrm{sym}}$ in (3.4) and $\widehat{\mathbf{Q}}_c^{\mathrm{sym}}$ in (4.4) respectively. This demonstrates the exponential convergence of the sequence $\mathbf{x}_n, n\ge 0$, established in Theorems 2.1 and 3.1, and hence the proposed PGDA, SPGDA, PGDA1h and SPGDA1h can be applied to find eigenvectors associated with a given eigenvalue.

One may verify that 1 is the maximal eigenvalue of matrices $\mathbf{H}_{\theta,m}, m\ge 1$. Therefore the OpGDA in Remark 2.4 and the OpSGDA in Remark 3.2 with $\mathbf{A}$ replaced by $\mathbf{I}-\mathbf{H}_{\theta,m}$, and the

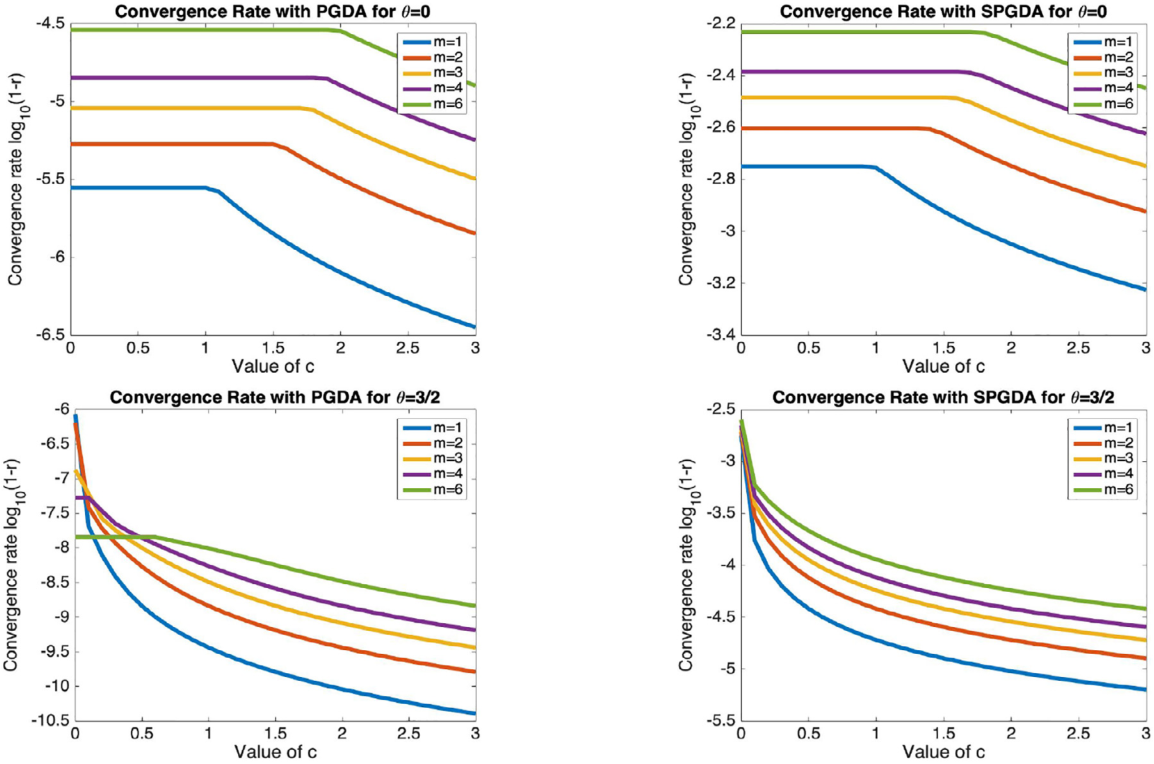

Fig. 3. Plotted are the convergence rate r in the logarithmic scale $\log_{10}(1-r)$ for using PGDA (left) and SPGDA (right), see (2.12) and (3.6), to find the principal eigenvectors of graph filters $\mathbf{H}_{\theta,m}$ on a random geometric graph $\mathcal{G}_{512}$, where the weight parameter $\theta = 0$ (top) and $\theta = 3/2$ (bottom), $m = 1, 2, 3, 4, 6$ and $0 \le c \le 3$. It is observed that the PGDA and SPGDA converge slow for large thresholding constant $c > 0$.

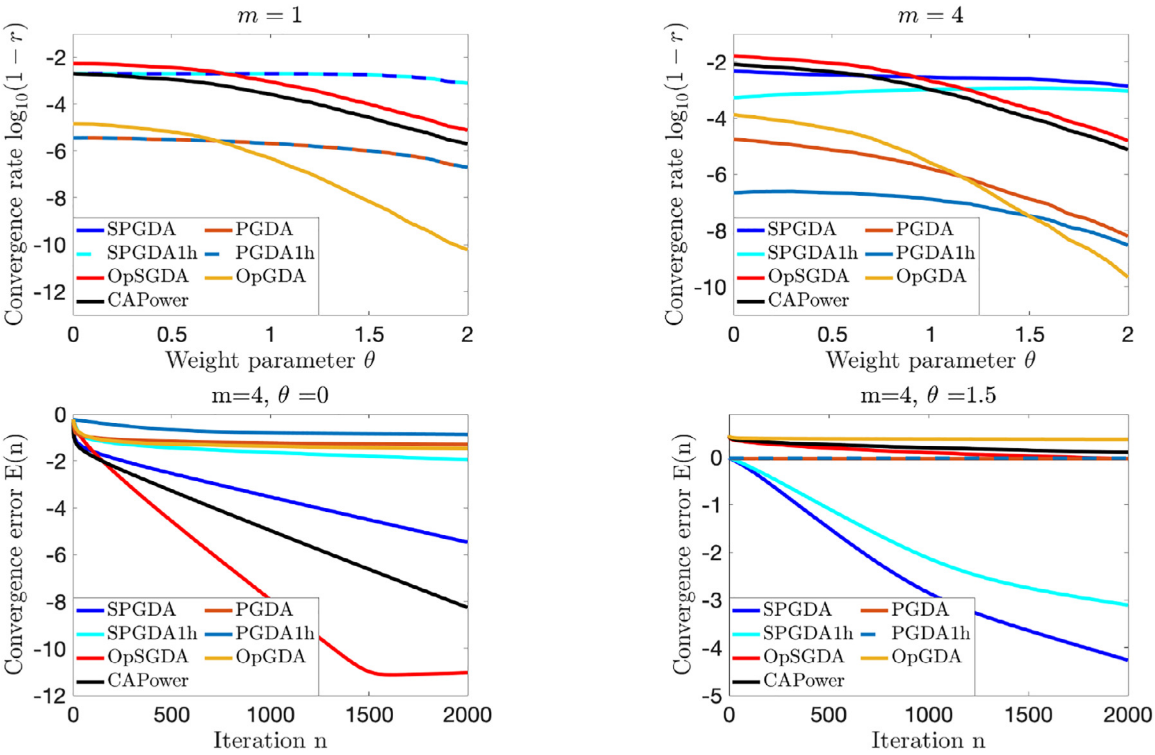

Fig. 4. Plotted on the top are the average of convergence rate r in the logarithmic scale $\log_{10}(1-r)$ over 200 trials versus the weight parameter $0 \le \theta \le 2$ for $m = 1$ (left) and $m = 4$ (right), where the average of the convergence rate for the algorithms, SPGDA, SPGDA1h, OpSGDA, CAPower, PGDA, PGDA1h and OpGDA, with $m = 4$ and $\theta = 3/2$ are 0.997548570, 0.998814592, 0.999783754, 0.999896610, 0.999999867, 0.999999967, 0.999999968 respectively. On the second row are the average of the convergence errors $E(n)$, $1 \le n \le 2000$, in the logarithmic scale over 200 trials, where $m = 4$ and $\theta = 0, 3/2$ from the left to the right respectively.

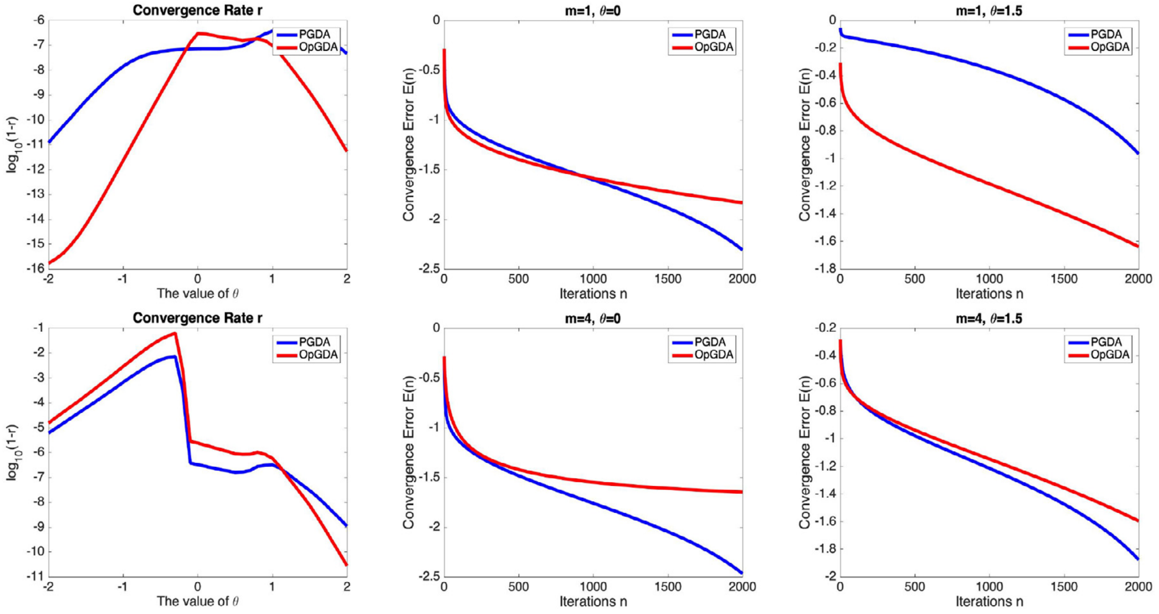

Fig. 5. Plotted are the average of convergence rate r in the logarithmic scale $\log_{10}(1-r)$ over 200 trials versus the weight parameter $-2 \le \theta \le 2$ for $m = 1$ (top left) and $m = 4$ (bottom left), and the average of the convergence errors $E(n), 1 \le n \le 2000$, in the logarithmic scale over 200 trials, where $(m, \theta) = (1, 0)$ (top middle), $(1, 3/2)$ (top right), $(4, 0)$ (bottom middle) and $(4, 3/2)$ (bottom right). The averages of the convergence rate for the algorithms PGDA and OpGDA are 0.9999998612 and 0.9999995487 (top middle), 0.9999994358 and 0.9999999970 (top right), 0.9999993358 and 0.9999999496 (bottom middle), and 0.9999951127 and 0.9999999828 (bottom right) respectively.

CAPower in Remark 3.3 with $\mathbf{H}$ and $\lambda_{\max}(\mathbf{H})$ replaced by $\mathbf{H}_{\theta,m}$ and 1 can be used to find principal eigenvectors associated with eigenvalue 1 of matrices $\mathbf{H}_{\theta,m}, m \ge 1$. Shown in Fig. 4 are their performance and comparison with the proposed PGDA, SPGDA, PGDA1h and SPGDA1h, where the convergence rate for the CAPower is given in (3.10) with $\mathbf{H}$ and $\lambda_{\max}(\mathbf{H})$ replaced by $\mathbf{H}_{\theta,m}$ and 1, and the convergence rates for the OpGDA and OpSGDA are given in (2.14) and (3.8) with $\mathbf{A}$ replaced by $\mathbf{I} - \mathbf{H}_{\theta,m}$ respectively.

From the above numerical simulations, we observe that SPGDA, SPGDA1h, OpSGDA, CAPower outperform OpGDA, PGDA and PGDA1h to find principal eigenvectors associated with maximal eigenvalue of positive semidefinite matrices $\mathbf{H}_{\theta,m}, m \ge 1$, the CAPower has slightly slower convergence speed than the OpSGDA does, and the OpSGDA has the best performance for small $\theta \ge 0$ while the distributed SPGDA has the fastest convergence rate for large $\theta \le 2$. We remark that OpSGDA and OpGDA are usually not considered as distributed algorithms as we need some global information to determine the optimal step size.

Numerical simulations are also conducted on the Minnesota traffic graph with coordinates of vertices being rescaled such that it can be deployed on the unit square $[0, 1]^2$ [18,45]. We observe similar performances of the algorithms to find principal eigenvectors associated with eigenvalue 1 of matrices $\mathbf{H}_{\theta,m}, m \ge 1$, on the Minnesota traffic graph.

In addition to find the eigenvector associated with the principal eigenvalue of the filters $\mathbf{H}_{\theta,m}$, we conduct the numerical experiments to evaluate the eigenvector associated with the second largest eigenvalue $\lambda_{\max,2}$ of matrices $\mathbf{H}_{\theta,m}$, which is also the eigenvector associated with the first positive eigenvalue of weighted Laplacian $\mathbf{W}_\theta \mathbf{L}^{\rm sym} \mathbf{W}_\theta$, by the algorithm PGDA in (2.12), and to compare the performance with the OpGDA in Remark 2.4 with $\mathbf{A}$ and $\mathbf{Q}$ replaced by $\lambda_{\max,2}\mathbf{I} - \mathbf{H}_{\theta,m}$ and $\mathbf{Q}_c$ in (2.10) respectively. Shown in Fig. 5 is the average over 200 trials for the convergence rate in the logarithmic scale with $-2 \le \theta \le 2$ and $m = 1, 4$, and the relative convergence errors $E(n)$ for $\theta = 0, 3/2$ and $m = 1, 4$, where the convergence rates r for the PGDA given in (2.12) and the OpGDA given in (2.14). This demonstrates that the sequence $\mathbf{x}_n, n \ge 0$, in Theorem 2.1 has the exponential convergence and that the proposed distributed PGDA can be applied to find eigenvectors associated with any given eigenvalue and it has comparable performance with the global OpGDA algorithm on finding eigenvectors associated with a given eigenvalue.

6. Conclusions and future study

On an SDN with communication range L, we can use the distributed PGDA with the preconditioning matrix in (2.2) to find eigenvectors associated with an **arbitrary** given eigenvalue for a matrix $\mathbf{H}$ with geodesic-width $\omega(\mathbf{H}) \le L$, and the distributed SPGDA with the preconditioning matrix in (3.4) to find eigenvectors associated with **maximal/minimal** eigenvalue of a positive semidefinite matrix $\mathbf{H}$ with geodesic-width $\omega(\mathbf{H}) \le L$.

In [35,36], asynchronous power iteration method is proposed to find principal eigenvectors of some left stochastic matrix. We do not know whether asynchronous power iteration method can be extended to find eigenvectors of arbitrary matrix with small geodesic-width, or whether we can find a synchronous version of the distributed algorithms proposed in this paper, which be considered in our future works.

The importance of distributively estimating the eigenvalues of a matrix on a graph $\mathcal{G}$ has been illustrated in many applications, see [29] and references therein. Most of distributed algorithms available are based on the consensus approach, and hence the memory, computational cost and communication expense for each vertex depend linearly or quadratically on the order of the graph. The distributed estimation of eigenvalues of matrices on graphs with small geodesic-width will be considered in our future work so that, similar to the proposed algorithms in this paper to evaluate eigenvectors, the memory, computational cost and communication expense for each vertex depend independently on the order of the graph, c.f. [55,56] for distributed criteria on the stability of matrices on graphs.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

CRediT authorship contribution statement

Nazar Emirov: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Cheng Cheng:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Qiyu Sun:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Zhihua Qu:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Acknowledgement

This work is partially supported by the National Natural Science Foundation of China (Grant No. 12171490), the National Science Foundation (DMS-1816313), and Guangdong Natural Science Foundation (Grant No. 2022A1515011060). The authors would like to thank the anonymous reviewers and the editor to provide many constructive comments and additional references for the improvement of the paper.

References

[1] J.X. Parreira, D. Donato, S. Michel, G. Weikum, Effcient and decentralized pagerank approximation in a peer-to-peer web search network, in: Proceedings of the 32nd International Conference on Very Large Data Bases, 2006, pp. 415–426.
[2] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor network survey, Comput. Netw. 52 (12) (2008) 2292–2330.
[3] A. Bertrand, M. Moonen, Distributed computation of the Fiedler vector with application to topology inference in ad hoc networks, Signal Process. 93 (5) (2013) 1106–1117.
[4] D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, 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. 30 (3) (2013) 83–98.
[5] A. Sandryhaila, 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. 31 (5) (2014) 80–90.
[6] D.F. Gleich, PageRank beyond the web, SIAM Rev. 57 (3) (2015) 321–363.
[7] R. Hebner, The power grid in 2030, IEEE Spectr. 54 (4) (2017) 50–55.
[8] A. Ortega, P. Frossard, J. Kovacevic, J.M.F. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, Proc. IEEE 106 (5) (2018) 808–828.
[9] C. Cheng, Y. Jiang, Q. Sun, Spatially distributed sampling and reconstruction, Appl. Comput. Harmon. Anal. 47 (1) (2019) 109–148.
[10] E. Isufi, A. Loukas, A. Simonetto, G. Leus, Autoregressive moving average graph filtering, IEEE Trans. Signal Process. 65 (2) (2017) 274–288.
[11] S. Segarra, A.G. Marques, A. Ribeiro, Optimal graph-filter design and applications to distributed linear network operators, IEEE Trans. Signal Process. 65 (15) (2017) 4117–4131.
[12] W. Waheed, D.B.H. Tay, Graph polynomial filter for signal denoising, IET Signal Process. 12 (3) (2018) 301–309.
[13] D.I. Shuman, P. Vandergheynst, D. Kressner, P. Frossard, Distributed signal processing via Chebyshev polynomial approximation, IEEE Trans. Signal Inf. Process. Netw. 4 (4) (2018) 736–751.
[14] M. Coutino, E. Isufi, G. Leus, Advances in distributed graph filtering, IEEE Trans. Signal Process. 67 (9) (2019) 2320–2333.
[15] N. Emirov, C. Cheng, J. Jiang, Q. Sun, Polynomial graph filter of multiple shifts and distributed implementation of inverse filtering, Sampling Theory, Signal Processing, and Data Analysis, 20, 2022.
[16] C. Cheng, N. Emirov, Q. Sun, Preconditioned gradient descent algorithm for inverse filtering on spatially distributed networks, IEEE Signal Process. Lett. 27 (2020) 1834–1838.
[17] V.N. Ekambaram, G.C. Fanti, B. Ayazifar, K. Ramchandran, Circulant structures and graph signal processing, in: Proc. IEEE Int. Conf. Image Process., 2013, pp. 834–838.
[18] J. Jiang, C. Cheng, Q. Sun, Nonsubsampled graph filter banks: theory and distributed algorithms, IEEE Trans. Signal Process. 67 (15) (2019) 3938–3953.
[19] J. Jiang, D.B. Tay, Q. Sun, S. Ouyang, Design of nonsubsampled graph filter banks via lifting schemes, IEEE Signal Process. Lett. 27 (2020) 441–445.
[20] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[21] B. Ruhnau, Eigenvector-centrality – a node-centrality? Soc. Netw. 22 (4) (2000) 357–365.
[22] A. Knyazev, Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method, SIAM J. Sci. Comput. 23 (2) (2001) 517–541.
[23] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. (2002) 849–856.
[24] U.V. Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416.
[25] T. Xiang, S. Gong, Spectral clustering with eigenvector selection, Pattern Recognit. 41 (3) (2008) 1012–1029.
[26] X. Ma, L. Gao, X. Yong, Eigenspaces of networks reveal the overlapping and hierarchical community structure more precisely, J. Stat. Mech. Theory Exp. 2010 (8) (2010) P08012.
[27] R.J. Sánchez-García, M. Fennelly, S. Norris, N. Wright, G. Niblo, J. Brodzki, J.W. Bialek, Hierarchical spectral clustering of power grids, IEEE Trans. Power Syst. 29 (5) (2014) 2229–2237.
[28] D.I. Shuman, M.J. Faraji, P. Vandergheynst, A multiscale pyramid transform for graph signals, IEEE Trans. Signal Process. 64 (8) (2016) 2119–2134.
[29] A. Gusrialdi, Z. Qu, Distributed estimation of all the eigenvalues and eigenvectors of matrices associated with strongly connected digraphs, IEEE Control Syst. Lett. 1 (2) (2017) 329–333.
[30] Z. Xu, X. Cao, X. Gao, Convergence analysis of gradient descent for eigenvector computation, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18), 2018, pp. 2933–2939.
[31] F. Tudisco, F. Arrigo, A. Gautier, Node and layer eigenvector centralities for multiplex networks, SIAM J. Appl. Math. 78 (2) (2018) 853–876.
[32] L. Lv, K. Zhang, T. Zhang, D. Bardou, Y. Zhang, Y. Cai, PageRank centrality for temporal networks, Phys. Lett. A 383 (12) (2019) 1215–1222.
[33] S. Mou, A.S. Morse, A fixed-neighbor, distributed algorithm for solving a linear algebraic equation, in: 2013 European Control Conference (ECC), IEEE, 2013, pp. 2269–2273.
[34] O.M. Duru, Analysis of a distributed algorithm for solving linear equations, Marmara Universitesi (Turkey), 2019 Doctoral Dissertation.
[35] B. Lubachevsky, D. Mitra, J. ACM, A chaotic asynchronous algorithm for computing the fixed point of a nonnegative matrix of unit spectral radius 33 (1) (1986) 130–150.
[36] M. Jelasity, G. Canright, K. Engo-Monsen, Asynchronous distributed power iteration with gossip-based normalization, in: European Conference on Parallel Processing, Springer, 2007, pp. 514–525.
[37] S. Chen, A. Sandryhaila, J.M.F. Moura, J. Kovacevic, Signal recovery on graphs: variation minimization, IEEE Trans. Signal Process. 63 (17) (2015) 4609–4624.
[38] B. Nadler, S. Lafon, I. Kevrekidis, R. Coifman, Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators, in: Y. Weiss, B. Schölkopf, J. Platt (Eds.), Advances in Neural Information Processing Systems 18, MIT Press, Cambridge, 2006, pp. 955–962.
[39] D. Cvetković, P. Rowlinson, S. Simic, An introduction to the theory of graph spectra, London Mathematical Society Student Texts, Cambridge University Press, 2009.
[40] G.H. Golub, C.F.V. Loan, Matrix Computations, The Johns Hopkins University Press, 2013.
[41] M. Jelasity, G. Canright, K. Engo-Monsen, Asynchronous distributed power iteration with gossip-based normalization, in: Proceeding of European Conference on Parallel Processing, Springer, Berlin, Heidelberg, 2007.
[42] P. Lorenzo, S. Barbarossa, Distributed estimation and control of algebraic connectivity over random graphs, IEEE Trans. Signal Process. 62 (21) (2014) 5615–5628.
[43] G. Muniraju, C. Tepedelenlioglu, A. Spanias, Consensus based distributed spectral radius estimation, IEEE Signal Process. Lett. 27 (2020) 1045–1049.
[44] D.K. Hammod, P. Vandergheynst, R. Gribonval, Wavelets on graphs via spectral graph theory, Appl. Comput. Harmon. Anal. 30 (4) (2011) 129–150.
[45] S.K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data, IEEE Trans. Signal Process. 60 (6) (2012) 2786–2799.
[46] A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs, IEEE Trans. Signal Process. 61 (7) (2013) 1644–1656.
[47] A. Sandryhaila, J.M.F. Moura, Discrete signal processing on graphs: frequency analysis, IEEE Trans. Signal Process. 62 (12) (2014) 3042–3054.
[48] X. Shi, H. Feng, M. Zhai, T. Yang, B. Hu, Infinite impulse response graph filters in wireless sensor networks, IEEE Signal Process. Lett. 22 (8) (2015) 1113–1117.
[49] O. Teke, P.P. Vaidyanathan, Extending classical multirate signal processing theory to graphs Part II: m-channel filter banks, IEEE Trans. Signal Process. 65 (2) (2017) 423–437.

[50] J. Yi, L. Chai, Graph filter design for multi-agent system consensus, in: IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, 2017, pp. 1082–1087.
[51] K. Lu, A. Ortega, D. Mukherjee, Y. Chen, Efficient rate-distortion approximation and transform type selection using laplacian operators, in: 2018 Picture Coding Symposium (PCS), San Francisco, CA, 2018, pp. 76–80.
[52] C. Cheng, J. Jiang, N. Emirov, Q. Sun, Iterative Chebyshev polynomial algorithm for signal denoising on graphs, in: Proceeding 13th Int. Conf. on SampTA, Bordeaux, France, 2019, pp. 1–5.
[53] P. Nathanael, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, D.K. Hammond, GSPBOX: a toolbox for signal processing on graphs, 2014, arXiv:1408.5781.
[54] M.S. Kotzagiannidis, P.L. Dragotti, Sampling and reconstruction of sparse signals on circulant graphs - an introduction to graph-FRI, Appl. Comput. Harmon. Anal. 47 (3) (2019) 539–565.
[55] Q. Sun, Stability criterion for convolution-dominated infinite matrices, Proc. Amer. Math. Soc. 138 (11) (2010) 3933–3943.
[56] N. Motee, Q. Sun, Localized stability certificates for spatially distributed systems over sparse proximity graphs, SIAM J. Control Optim. (2022). Accepted