Iterative Chebyshev Polynomial Algorithm for Signal Denoising on Graphs

Cheng Cheng, Junzheng Jiang, Nazar Emirov, and Qiyu Sun

***Abstract*—In this paper, we consider the inverse graph filtering process when the original filter is a polynomial of some graph shift on a simple connected graph. The Chebyshev polynomial approximation of high order has been widely used to approximate the inverse filter. In this paper, we propose an iterative Chebyshev polynomial approximation (ICPA) algorithm to implement the inverse filtering procedure, which is feasible to eliminate the restoration error even using Chebyshev polynomial approximation of lower order. We also provide a detailed convergence analysis for the ICPA algorithm and a distributed implementation of the ICPA algorithm on a spatially distributed network. Numerical results are included to demonstrate the satisfactory performance of the ICPA algorithm in graph signal denoising.**

Keywords: Graph signal processing, Chebyshev polynomial approximation, Distributed algorithm, Laplacian matrix.

I. Introduction

Spatial distributed networks (SDNs) have been widely used in (wireless) sensor networks [1], smart power grids [2], drone fleets and multirobot networks, and many real world applications. An SDN has large amount of agents spatially deployed, with each of them having some data processing capability and exchanging data only with its adjacent neighbors owing to its limited communication capacity and/or privacy concern. Due to its irregular structure, topology of an SDN is often described by a graph $\mathcal{G} := (V, E)$, where a vertex $i \in V$ represents an agent in the SDN and an edge $(i, j) \in E$ means that data collected/stored at the agent $j \in V$ can be sent/shared to the agent $i \in V$ via some direct communication link [3], [4].

Data collected on an SDN are often modeled by a graph signal $\mathbf{x} = (x(i))_{i\in V}$ indexed by vertices $i \in V$ [3], [4]. Borrowed from classical signal processing, some basic building blocks have been introduced for graph signal processing [4]–[5]. One of fundamental concepts is graph filtering which maps a graph signal $\mathbf{x}$ linearly to another graph signal

$$\mathbf{b} = \mathbf{H}\mathbf{x}, \tag{I.1}$$

where the graph filter $\mathbf{H}$ can be represented by a matrix $\mathbf{H} = (H(i,j))_{i,j\in V}$ indexed by vertices in $\mathcal{G}$. The graph filters have been widely used in signal denoising, inpainting, smoothing, reconstructing and semi-supervised learning [4], [6], [7], [5].

In this paper, we consider graph filters $\mathbf{H}$ being polynomials of a graph shift $\mathbf{S}$,

$$\mathbf{H} = h(\mathbf{S}) = h_0\mathbf{I} + \sum_{j=1}^{L} h_l\mathbf{S}^l, \tag{I.2}$$

where $h(t) = \sum_{l=0}^{L} h_l t^l$ for some coefficients $h_l, 0 \le l \le L$, and a graph shift $\mathbf{S} = (s(i,j))_{i,j\in V}$ on a graph $\mathcal{G} = (V, E)$ satisfies

$$s(i,j) = 0 \quad \text{if } (i,j) \notin E.$$

Illustrative examples of graph shifts are the adjacency matrix $\mathbf{A}_\mathcal{G}$, the degree matrix $\mathbf{D}_\mathcal{G}$ of the graph $\mathcal{G}$, Laplacian matrix $\mathbf{L}_\mathcal{G} := \mathbf{D}_\mathcal{G} - \mathbf{A}_\mathcal{G}$, and normalized Laplacian matrix $\mathbf{L}^{\rm sym}_\mathcal{G} = \mathbf{D}_\mathcal{G}^{-1/2}\mathbf{L}_\mathcal{G}\mathbf{D}_\mathcal{G}^{-1/2}$ [6], [8], [7], [5].

An SDN does not have a strong central node to govern all data processing over entire networks, and then signal processing on an SDN should be implemented in a distributed manner, which involves local data processing and interchanging only. Many graph signal processing can be naturally implemented in a distributed manner, such as the graph filtering with filters being the polynomials of a graph shift [9]. However, not all procedures can be realized in a distributed manner, for instance, the inverse filtering to restore a signal $\mathbf{x}$ from its observation in (I.1)

$$\mathbf{x} = \mathbf{H}^{-1}\mathbf{b}, \tag{I.3}$$

since $\mathbf{H}^{-1}$ is not usually a polynomial of small degree [10].

To tackle the above problem, the Chebyshev polynomial approximation has been employed to approximate the inverse filter $\mathbf{H}^{-1}$, which has been successfully applied in graph signal processing, including graph wavelet filter bank [7], [5], denoising, smoothing, and semi-supervised learning on graphs [10] etc. However, to achieve a reasonable accuracy of the restoration, the truncated Chebyshev polynomial may have high order and the graph filter $\mathbf{H}$ should have certain regularity [7], [11], [12].

In this paper, we introduce an iterative Chebyshev polynomial approximation algorithm (II.7) and (II.8), ICPA in short, for the inverse filtering procedure when the filter $\mathbf{H}$ is a polynomial of a graph shift. The proposed ICPA algorithm achieves any restoration accuracy at the expense of additional iterations, while the truncated Chebyshev polynomial approximation of lower degree can be selected.

In this paper, we show that the ICPA algorithm has exponential convergence and it can be implemented in a distributed manner with each agent exchanging data information with its adjacent agents only. Numerical simulations on applying ICPA algorithm to graph denoising with Tikhonov regularization have been shown to demonstrate its satisfactory performance.

II. Iterative Chebyshev Polynomial Approximation Algorithm

In this section, we introduce an iterative algorithm to implement the inverse filtering procedure (I.3), when the graph shift $\mathbf{S}$ has its spectrum $\sigma(\mathbf{S})$ contained in an interval $[a, b]$, and the filter $\mathbf{H} = h(\mathbf{S})$ is a polynomial h of the graph shift $\mathbf{S}$ that satisfies

$$h(t) \neq 0 \quad \text{for all} \quad t \in [a, b]. \tag{II.1}$$

Define Chebyshev polynomials $T_k, k \geq 0$, iteratively by

$$T_k(s) = \begin{cases} 1 & \text{if } k = 0, \\ s & \text{if } k = 1, \\ 2sT_{k-1}(s) - T_{k-2}(s) & \text{if } k \geq 2. \end{cases} \tag{II.2}$$

By (II.1), we know that $1/h$ is an analytic function on a neighborhood of the interval $[a, b]$, and hence it has Fourier expansion of shifted Chebyshev polynomials,

$$\frac{1}{h(t)} = \frac{1}{2}c_0 + \sum_{k=1}^{\infty} c_k T_k\Big(\frac{2t - a - b}{b - a}\Big), \ t \in [a, b], \tag{II.3}$$

where

$$c_k = \frac{2}{\pi} \int_0^{\pi} \frac{\cos k\theta}{h\Big(\frac{a+b}{2} + \frac{b-a}{2} \cos \ \theta\Big)} d\theta, \ k \geq 0. \tag{II.4}$$

Set

$$g_K(t) = \frac{c_0}{2} + \sum_{k=1}^{K} c_k T_k\Big(\frac{2t - a - b}{b - a}\Big), \ K \geq 0.$$

By (II.1), the truncated Chebeshev polynomials $g_K, K \geq 0$, approximate the analytic function $1/h$ uniformly on $[a, b]$,

$$\lim_{K \to \infty} \max_{t \in [a,b]} |1 - h(t) g_K(t)| = 0. \tag{II.5}$$

In this paper, we will select an integer $K \geq 0$ such that g_K is a good approximation to $1/h$ in the sense that

$$\varepsilon_K := \sup_{t \in [a,b]} \big|1 - h(t) g_K(t)\big| < 1. \tag{II.6}$$

The existence of such an integer K follows from (II.5) [7].

With the above selection of the truncated Chebeshev polynomial g_K, we propose an iterative algorithm to implement the inverse filtering procedure (I.3):

$$\begin{cases} \mathbf{z}^{(m)} = \mathbf{G}_K \mathbf{b}^{(m-1)} \\ \mathbf{x}^{(m)} = \mathbf{x}^{(m-1)} + \mathbf{z}^{(m)} \\ \mathbf{b}^{(m)} = \mathbf{b}^{(m-1)} - \mathbf{H}\mathbf{z}^{(m)}, \ m \geq 1, \end{cases} \tag{II.7}$$

with initials

$$\mathbf{b}^{(0)} = \mathbf{b} \quad \text{and} \quad \mathbf{x}^{(0)} = \mathbf{0}, \tag{II.8}$$

where $\mathbf{G}_K = g_K(\mathbf{S})$. We call the above algorithm by *iterative Chebyshev polynomial approximation algorithm* and **ICPA** for abbreviation.

In the following theorem, we show that the proposed ICPA algorithm (II.7) and (II.8) converges exponentially to the output of the inverse filtering procedure (I.3).

Theorem II.1. Let $\mathbf{b}$ be a graph signal, $\mathbf{S}$ be a symmetric graph shift with its spectrum $\sigma(\mathbf{S}) \subset [a, b]$, and $\mathbf{H} = h(\mathbf{S})$ for some polynomial h satisfying (II.1). Take a Chebyshev polynomial approximation g_K to the function h^{-1} on $[a, b]$ such that (II.6) holds. Then $\mathbf{x}^{(m)}, m \geq 0$, in the ICPA algorithm (II.7) and (II.8) converges exponentially to $\mathbf{H}^{-1}\mathbf{b}$,

$$\|\mathbf{x}^{(m)} - \mathbf{H}^{-1}\mathbf{b}\|_2 \leq \Big(\inf_{t \in [a,b]} |h(t)|\Big)^{-1} \epsilon_K^m \|\mathbf{b}\|_2, \ m \geq 0, \tag{II.9}$$

where ϵ_K is given in (II.6).

Proof: Set $r(t) = 1 - h(t) g_K(t)$. Then

$$r(\mathbf{S}) = \mathbf{I} - \mathbf{H}\mathbf{G}_K \tag{II.10}$$

and

$$\sigma\big(r(\mathbf{S})\big) = \big\{r(z), \ z \in \sigma(\mathbf{S})\big\}. \tag{II.11}$$

By (II.7) and (II.8), we can prove by induction on $m \geq 0$ that

$$\mathbf{b}^{(m)} = (r(\mathbf{S}))^m \mathbf{b} \tag{II.12}$$

and

$$\mathbf{H}^{-1}\mathbf{b} - \mathbf{x}^{(m)} = \mathbf{H}^{-1}\mathbf{b}^{(m)}, \ m \geq 0. \tag{II.13}$$

Recalling that the graph shift $\mathbf{S}$ is symmetric and applying (II.6) and (II.10)–(II.12), we obtain

$$\|\mathbf{b}^{(m)}\|_2 \leq \epsilon_K^m \|\mathbf{b}\|_2, \ m \geq 0.$$

This together with (II.13) and the symmetry of the graph shift $\mathbf{S}$ proves (II.9). ■

III. Distributed Implementation of ICPA Algorithm

In this section, we consider distributed implementation of the ICPA algorithm (II.7) and (II.8) on a simple connected graph $\mathcal{G} := (V, E)$ [3], [13]. Here a graph $\mathcal{G}$ is simple if it is undirected and unweighted, and it does not contain self-loops and multiple edges.

Definition III.1. *The bandwidth $\varpi := \varpi(\mathbf{A})$ of a graph filter $\mathbf{A} = (a(i, j))_{i,j \in V}$ is the minimal nonnegative integer such that $a(i, j) = 0$ for all $i, j \in V$ with $\rho(i, j) > \varpi$, where the geodesic distance $\rho(i, j)$ between vertices $i, j \in V$ is defined by the number of edges in a shortest path connecting them.*

A graph shift $\mathbf{S}$ has bandwidth one and a polynomial filter $\mathbf{H} = h(\mathbf{S})$ of a graph shift $\mathbf{S}$ has bandwidth

$$\varpi(\mathbf{H}) \leq L,$$

where L is the degree of the polynomial h. For a graph filter $\mathbf{A} = (a(i, j))_{i,j \in V}$ with bandwidth $\varpi := \varpi(\mathbf{A})$, locations of all nonzero entries $a(i, j) \neq 0$ in the i-th row are in the ϖ-hop neighbor of the agent i, which implies that the number of nonzero entries at each row of the graph filter is bounded by $((\Delta(\mathcal{G}))^{\varpi+1} - 1)/((\Delta(\mathcal{G})) - 1)$, where $\Delta(\mathcal{G})$ is the maximal degree of the graph $\mathcal{G}$ [3].

As filters $\mathbf{G}_K$ and $\mathbf{H}$ in (II.7) have bandwidths K and L respectively, the centralized implementation of the ICPA algorithm (II.7) and (II.8) needs to perform $O((\Delta(\mathcal{G}))^{\max(L,K)} n)$ multiplications and additions in each iteration, where n is the order of the graph $\mathcal{G}$. By Theorem II.1, the ICPA algorithm (II.7) and (II.8) will reach the approximation accuracy ϵ after $O(\ln(\|\mathbf{b}\|_2/\epsilon))$ iterations. Therefore the computational cost to implement the ICPA algorithm (II.7) and (II.8) in a centralized facility is about $O\big((\Delta(\mathcal{G}))^{\max(L,K)}\ln(\|\mathbf{b}\|_2/\epsilon)n\big)$.

The ICPA algorithm (II.7) and (II.8) can be implemented in a distributed manner with each agent exchanging data information with its adjacent agents only. By the recurrence relation (II.2) for Chebyshev polynomials, the first step $\mathbf{z}^{(m)} = \mathbf{G}_K(\mathbf{S})\mathbf{b}^{(m-1)}$ in the ICPA algorithm (II.7) and (II.8) can be implemented in a distributed manner as follows, see Figure 1 for the block diagram:

$$\begin{cases} \mathbf{u}_0 = \mathbf{b}^{(m-1)} \\ \mathbf{u}_1 = \tilde{\mathbf{S}}\mathbf{u}_0, \ \tilde{\mathbf{z}}_1 = \frac{c_0}{2}\mathbf{u}_0 + c_1\mathbf{u}_1 \\ \mathbf{u}_l = 2\tilde{\mathbf{S}}\mathbf{u}_{l-1} - \mathbf{u}_{l-2}, \tilde{\mathbf{z}}_l = \tilde{\mathbf{z}}_{l-1} + c_l\mathbf{u}_l, \quad 2 \le l \le K, \end{cases} \tag{III.1}$$

with the output $\mathbf{z}^{(m)} = \tilde{\mathbf{z}}_K$, where

$$\tilde{\mathbf{S}} = \frac{2}{b-a}\mathbf{S} - \frac{a+b}{b-a}\mathbf{I},$$

and $\mathbf{S}$ are graph shifts on the graph $\mathcal{G}$. Similarly, the filtering procedure $\mathbf{H}\mathbf{z}^{(m)}$ in the third step of the ICPA algorithm (II.7) and (II.8) can also be implemented in a distributed manner:

$$\mathbf{w}_0 = h_L\mathbf{z}^{(m)} \text{ and } \mathbf{w}_l = h_{L-l}\mathbf{z}^{(m)} + \mathbf{S}\mathbf{w}_{l-1}, 1 \le l \le L, \tag{III.2}$$

where $\mathbf{w}_L = \mathbf{H}\mathbf{z}^{(m)}$ is the output signal of the filtering procedure, see Figure 2 for the block diagram. In each iteration

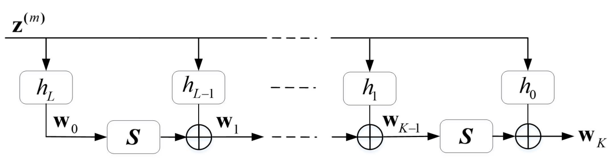

Fig. 2: Block diagram to implement $\mathbf{H}\mathbf{z}^{(m)}$.

of the above implementation for the ICPA algorithm (II.7) and (II.8), each agent $k \in V$ in an SDN only needs to exchange data information with its adjacent agents and to perform at most $2\deg(k)+3 \le 2\Delta(\mathcal{G})+3$ multiplications and additions, where $\deg(k)$ is the degree of the vertex k in the graph $\mathcal{G}$.

Presented below is the pseudo code for the ICPA algorithm (II.7) and (II.8) for every agent $k \in V$ in an SDN.

Algorithm III.1 Implementation of the ICPA Algorithm on an agent $k \in V$

Inputs: stop criterion ε, Chebyshev polynomial coefficients $c_l, l = 0, \cdots, K$ and polynomial coefficients $h_l, l = 0, \cdots, L$ of the polynomial filter $\mathbf{H} = h(\mathbf{S})$, the observation $\mathbf{b}_k = (b(i))_{i\in\mathcal{N}_k\cup k}$ around the agent k, the k-th row $\mathbf{S}_k = (S(k,i))_{i\in\mathcal{N}_k\cup k}$ and $\tilde{\mathbf{S}}_k = (\tilde{S}(k,i))_{i\in\mathcal{N}_k\cup k}$ of the shift matrices $\mathcal{S}$ and $\tilde{\mathbf{S}}$.
Initialization: $x^{(0)}(k) = 0, \mathbf{b}_k^{(0)} = \mathbf{b}_k$ and $m = 1$.
Iteration:
1a) Set $\mathbf{u}_{0,k}^{(m)} = \mathbf{b}_k^{(m-1)}$ and compute $u_{1,k}^{(m)} = \langle\tilde{\mathbf{S}}_k, \mathbf{u}_{0,k}^{(m)}\rangle$.
1b) Send $u_{1,k}^{(m)}$ to neighbors $i \in \mathcal{N}_k$ and receive $u_{1,i}^{(m)}$ from neighbors, then form vector $\mathbf{u}_{1,k}^{(m)} = (u_{1,i}^{(m)})_{i\in\mathcal{N}_k\cup k}$.
1c) Calculate $\tilde{z}_1^{(m)}(k) = \frac{1}{2}c_0u_{0,k}^{(m)}(k) + c_1u_{1,k}^{(m)}(k)$.
1d) **for** $l = 2, \cdots, K$ **do**
- Compute $u_{l,k}^{(m)} = 2\langle\tilde{\mathbf{S}}_k, \mathbf{u}_{l-1,k}^{(m)}\rangle - u_{l-2,k}^{(m)}$.
- Send $u_{l,k}^{(m)}$ to neighbors $i \in \mathcal{N}_k$ and receive $u_{l,i}^{(m)}$ from neighbors, then form vector $\mathbf{u}_{l,k}^{(m)} = (u_{l,i}^{(m)})_{i\in\mathcal{N}_k\cup k}$.
- Calculate $\tilde{z}_l^{(m)}(k) = \tilde{z}_{l-1}^{(m)}(k) + c_lu_{l,k}^{(m)}(k)$.

end for
1e) Set $z^{(m)}(k) = \tilde{z}_K^{(m)}(k)$.
2) Update $x^{(m)}(k) = x^{(m-1)}(k) + z^{(m)}(k)$.
3a) Calculate $w_0^{(m)}(k) = h_Lz^{(m)}(k)$.
3b) Send $w_0^{(m)}(k)$ to neighbors $i \in \mathcal{N}_k$ and receive $w_0^{(m)}(i)$ from neighbors, then form vector $\mathbf{w}_{0,k}^{(m)} = (w_0^{(m)}(i))_{i\in\mathcal{N}_k\cup k}$.
3c) **for** $l = 1, \cdots, L$ **do**
- Compute $w_l^{(m)}(k) = h_{L-l}z^{(m)}(k) + \langle\mathbf{S}_k, \mathbf{w}_{l-1,k}^{(m)}\rangle$.
- Send $w_l^{(m)}(k)$ to neighbors $i \in \mathcal{N}_k$ and receive $w_l^{(m)}(i)$ from neighbors, then form vector $\mathbf{w}_{l,k}^{(m)} = (w_l^{(m)}(i))_{i\in\mathcal{N}_k\cup k}$.

end for
3d) Update $b^{(m)}(k) = b^{(m-1)}(k) - w_L^{(m)}(k)$.
4) Send $b^{(m)}(k)$ to neighbors $i \in \mathcal{N}_k$ and receive $b^{(m)}(i)$ from neighbors, then form vector $\mathbf{b}_k^{(m)} = (b^{(m)}(i))_{i\in\mathcal{N}_k\cup k}$.
5) Evaluate $|z^{(m)}(k)| \le \varepsilon$. If yes, terminate the iteration and output $x^{(m)}(k)$ and m. Otherwise, set $m = m+1$ and start another iteration.
Outputs: $x^{(m)}(k)$ and m.

IV. Graph Signal Denoising

In this section, we apply the proposed ICPA algorithm (II.7) and (II.8) to the graph signal denoising with Tikhonov regularization.

In our setting on graph signal denoising, the observation is of the form $\mathbf{b} = \mathbf{x}_o + \eta$, where $\mathbf{x}_o$ is the original graph signal and η is the bounded additive noise. The original signal $\mathbf{x}_o$ is usually restored approximately through some optimization approaches which involve data fidelity terms and/or penalty terms. In this paper, we consider the graph signal denoising with Tikhonov regularization

$$\min \ \|\mathbf{x}-\mathbf{b}\|_2^2 + \alpha\mathbf{x}^T\mathbf{L}_{\mathcal{G}}^{\mathrm{sym}}\mathbf{x}, \tag{IV.1}$$

where α is a weighted factor and $\mathbf{L}_{\mathcal{G}}^{\mathrm{sym}}$ is the normalized Laplacian on the graph $\mathcal{G}$ whose spectrum is contained in $[0, 2]$ [10]. The solution to the above minimization problem is

$$\mathbf{x} = \mathbf{H}^{-1}\mathbf{b}, \tag{IV.2}$$

where $\mathbf{H} := \mathbf{I} + \alpha\mathbf{L}_{\mathcal{G}}^{\mathrm{sym}}$ is a polynomial of the graph shift $\mathbf{L}_{\mathcal{G}}^{\mathrm{sym}}$ [5], [9], [10].

In our simulations below, the original signal $\mathbf{x}_o$ is a piecewise constant function with vaules -1 and 1 [5] residing on the Minnesota traffic graph, which has 2642 vertices 3303 edges, see Figure 3. The available data $\mathbf{b} = \mathbf{x}_o + \eta$ is noisy observation of the original graph signal $\mathbf{x}_o$, where η is the

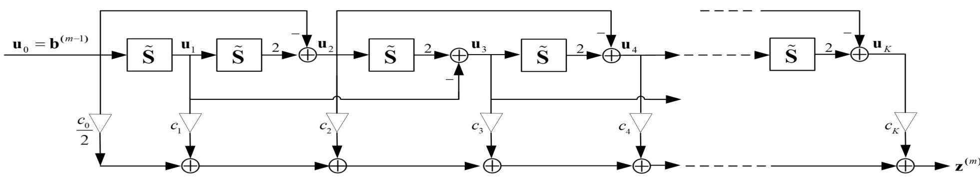

Fig. 1: Block diagram to implement $\mathbf{G}_K \mathbf{b}^{(m-1)}$.

additive bounded noise randomly distributed in $[-\varsigma, \varsigma]$.

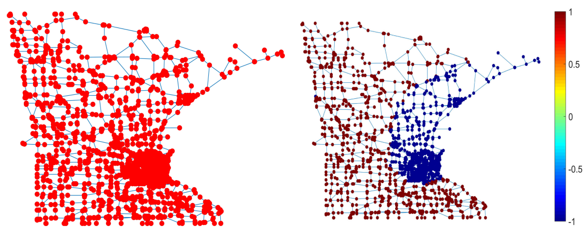

Fig. 3: Plotted on the left is the Minnesota traffic graph, and on the right is the graph signal $\mathbf{x}_o$ on the Minnesota traffic graph.

For the implementation of the ICPA algorithm (II.7) and (II.8) in a distributed manner, we need some information on the filter $\mathbf{H}$ and the truncated Chebyshev polynomial approximation g_K sent to every agent of the graph, particularly, the weighted factor α in the filter $\mathbf{H}$ and polynomial coefficients of the truncated Chebyshev polynomial approximation g_K. A toolbox to find the truncated Chebyshev polynomial approximation of order K has been developed in [7]. For instance, applying sgwt_cheby_coeff in sgwttoolbox, we obtain the first few Chebyshev polynomial approximation $g_K, K = 1, 2, 3$ when $\alpha = 2$,

$$g_1(t) = \frac{3}{7} - \frac{2}{7}t$$
$$g_2(t) = \frac{1}{3} - \frac{1}{3}t + \frac{2}{9}t^2$$
$$g_3(t) = \frac{15 - 10t + 12t^2 - 8t^3}{47}.$$

An alternative to the polynomial design of the graph filter in (I.2), the infinite impulse response (IIR) graph filters, such as the autoregressive moving average (ARMA) graph filters, are characterized by a rational graph frequency response [14], [15]. We remark that our ICPA algorithm (II.7)–(II.8) with the Chebyshev approximation $\mathbf{G}_K$ of the inverse filter $\mathbf{H}^{-1}$ being replaced by $\gamma\mathbf{H}$ and $\gamma\mathbf{I}$ becomes the Iterative Distributed IIR and Fast Iterative Distributed IIR respetively, where γ is the step length chosen appropriately. Also the ICPA algorithm (II.7)-(II.8) with $m = 1$ is equivalent to the CPA in [10]. In the extended version of this proceeding paper, we will provide more detailed comparison with other distributed filtering on graphs, cf. [10], [14], [15].

Our simulations on graph signal denosing through Tikhonov regularization (IV.1) have been carried out on Matlab platform equipped on a PC with i7-7700 CPU (3.6Hz) and 8GB memory, where the weighted factors α are $[2, 10, 40, 100]$ and the noise level ς are $[1/4, 1/2, 1, 2]$. Different weighted factors are used for different noise levels. Shown in Table I are the comparison between the Chebyshev polynomial approximation (CPA) algorithm [10] and the proposed iterative Chebyshev polynomial approximation (ICPA) algorithm (II.7) and (II.8). In all simulations for signal denoising, we use $20\log_{10}\|\mathbf{x}_o\|_2/\|\mathbf{b}-\mathbf{x}_o\|_2$ to measure the input ℓ^2-signal-to-noise ratio (ℓ^2-SNR) in dB, and $20\log_{10}\|\mathbf{x}_o\|_2/\|\mathbf{x}-\mathbf{x}_o\|_2$ to measure the output ℓ^2-SNR in dB. We observed that the proposed ICPA outperforms CPA in [10] with the same approximation order $K = 5$.

TABLE I: Denoising performance measured by the average SNR over 50 tests.

Noise Level ε	$1/4$	$1/2$	1	2
Input SNR	16.82	10.79	4.79	-1.26
CPA SNR	16.55	12.05	5.41	2.53
ICPA SNR	16.56	13.42	11.07	9.38

TABLE II: Performance comparison of the ICPA and CPA with different approximation order K over 50 tests, measured by average SNR.

K	1	2	5	10
Input SNR	10.80	10.79	10.79	10.78
CPA SNR	2.93	5.65	12.03	13.41
ICPA SNR	13.43	13.42	13.42	13.43
ICPA Iterations m	25	14	6	4

Shown in Table II are the comparison between the CPA [10] and the proposed ICPA with different polynomial orders K where the weighted factor $\alpha = 10$ and noise level $\varsigma = 1/2$. It is observed that the CPA algorithm provides worse approximations when the approximation order $K \le 10$, while the ICPA algorithm always achieves almost the same approximation accuracy at the expense of additional iterations. This suggests that for the implementation of the CPA algorithm on inverse filtering procedure, one may require to select the truncated Chebyshev polynomial g_K with large order K, while the proposed ICPA algorithm works well as long as the truncated Chebyshev polynomial g_K provides a good approximation to $1/h$, i.e. (II.6) holds.

V. Conclusions

In this paper, an iterative Chebyshev polynomial approximation (ICPA) algorithm has been proposed to implement the inverse filtering procedure when the filter is a polynomial of a graph shift. The proposed ICPA has the exponential convergence and linear computational complexity, and more importantly it can be fulfilled in a distributed manner with each agent exchanging data information with its adjacent agents only. The proposed algorithm is amenable to distributed processing of signals on sparse graphs of large orders.

Acknowledgement: This work is partially supported by the National Natural Science Foundation of China (Grant No. 61761011), and the National Science Foundation (DMS-1816313, DMS-1638521).

REFERENCES

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[2] R. Hebner, "The power grid in 2030," *IEEE Spectrum*, vol. 54, no. 4, pp. 50–55, 2017.

[3] C. Cheng, Y. Jiang, and Q. Sun, "Spatially distributed sampling and reconstruction," *Applied and Computational Harmonic Analysis, Available at https://doi.org/10.1016/j.acha.2017.07.007*.

[4] J. Jiang, C. Cheng, and Q. Sun, "Nonsubsampled graph filter banks and distributed implementation," *ArXiv Preprint, arXiv:1709.04107*, 2017.

[5] S. K. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4673–4685, 2013.

[6] D. I. Shuman, S. K. 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, 2013.

[7] 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.

[8] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

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

[10] D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard, "Distributed signal processing via chebyshev polynomial approximation," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 4, pp. 736–751, 2018.

[11] V. L. Druskin and L. A. Knizhnerman, "Two polynomial methods of calculating functions of symmetric matrices," *USSR Computational Mathematics and Mathematical Physics*, vol. 29, no. 6, pp. 112–121, 1989.

[12] L. N. Trefethen, *Approximation Theory and Approximation Practice*. SIAM, 2013, vol. 128.

[13] F. Chung and L. Lu, *Complex Graphs and Networks*. American Mathematical Society, 2006, no. 107.

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

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