DISTRIBUTED ALGORITHM FOR GRAPH SIGNAL INPAINTING

Siheng Chen[1,2]*, Aliaksei Sandryhaila*[3]*, Jelena Kovačević*[1,2,4]

[1]Dept. of ECE, [2]Center for Bioimage Informatics, [3]HP Vertica, [4]Dept. of BME,
Carnegie Mellon University,
Pittsburgh, PA, USA

ABSTRACT

We present a distributed and decentralized algorithm for graph signal inpainting. The previous work obtained a closed-form solution with matrix inversion. In this paper, we ease the computation by using a distributed algorithm, which solves graph signal inpainting by restricting each node to communicate only with its local nodes. We show that the solution of the distributed algorithm converges to the closed-form solution with the corresponding convergence speed. Experiments on online blog classification and temperature prediction suggest that the convergence speed of the proposed distributed algorithm is competitive with that of the centralized algorithm, especially when a graph tends to be regular. Since a distributed algorithm does not require to collect data to a center, it is more practical and efficient.

Index Terms— Signal processing on graphs, graph signal inpainting, distributed and decentralized computing

1. INTRODUCTION

The problem of collecting and analyzing data obtained from or represented by networks has been receiving increasing interest due to the abundance of such data in various research fields. Recently, a theoretical framework called *graph signal processing* has emerged as a new approach to analyze signals with irregular structure [1, 2, 3]. Its key idea is to represent the structure of a signal with a graph by associating signal coefficients with graph nodes and analyzing graph signals by using appropriately defined signal processing techniques, such as Fourier transform, filtering, and wavelets.

One task of analyzing data represented by graphs is to reconstruct or estimate graph signal coefficients that are missing, unmeasurable, or corrupted, called *graph signal inpainting* [4, 5, 6]. The approach via minimizing graph total variation provide a closed-form solution, which involves matrix inversion [7, 8]. To solve this, we propose a distributed and decentralized algortithm for graph signal inpainting that can be computed locally in the vertex domain without transmitting signal coefficients to a computing center. We show that the solution of the algorithm converges to the closed-form solution with certain convergence speed. We finally compare the convergence performance with a conventional centralized algorithm in the tasks of online blog classification and temperature prediction.

The authors gratefully acknowledge support from the NSF through awards 1130616 and 1017278, and CMU Carnegie Institute of Technology Infrastructure Award. This paper is an extended version.

2. DISCRETE SIGNAL PROCESSING ON GRAPHS

In this section, we briefly review relevant concepts of discrete signal processing on graphs; a thorough introduction can be found in [1, 2]. Discrete signal processing on graphs is a theoretical framework that generalizes classical discrete signal processing from regular domains, such as lines and rectangular lattices, to irregular structures that are commonly described by graphs.

Graph shift. DSP_G deals about signals with complex, irregular structure that is represented by a graph $G = (\mathcal{V}, \text{A})$, where $\mathcal{V} = \{v_i\}_{i=1}^N$ is the set of *nodes* and $\text{A} \in \mathbb{C}^{N\times N}$ is a *graph shift*, or a weighted adjacency matrix. Each $\text{A}_{i,j}$ contains the edge weight between the ith node and the jth node, which characterizes their relation. For example, the edge weight can quantify similarities and dependencies between nodes, or indicate communication patterns in networks.

Graph signals. Using this graph, we refer to a signal with structure as a graph signal, which is defined as a map that assigns a signal coefficient $\mathbf{x}_n \in \mathbb{C}$ to the graph node v_n, and write it in a vector form,

$$\mathbf{x} = \left[x_1, x_2, \cdots, x_N\right]^T \in \mathbb{C}^N.$$

Graph total variation. A concept often used in signal processing is that of smoothness. Smoothness of graph signals is expressed by a *graph total variation* function

$$\text{TV}_\text{A}(\mathbf{x}) = \left|\left|\mathbf{x} - \frac{1}{|\lambda_{max}(\text{A})|}\,\text{A}\,\mathbf{x}\right|\right|_1, \qquad (1)$$

where $\lambda_{max}(\text{A})$ denotes the eigenvalue of A with the largest magnitude. For notational simplicity, assume that the graph shift A has been normalized to satisfy $\lambda_{max}(\text{A}) = 1$. Note that when the graph shift A is the cyclic permutation matrix, it represents time-series signals, and (1) reduces to the same definition from classical signal processing. Often, the quadratic form of graph total variation is used instead [9],

$$\text{S}_2(\mathbf{x}) = ||\mathbf{x} - \text{A}\,\mathbf{x}||_2^2. \qquad (2)$$

We use the quadratic form of the graph total variation for two reasons. First, it is computationally easier to optimize than the ℓ_1-norm based graph total variation in (1). Second, the ℓ_1-norm based graph total variation, which penalizes less transient changes than the quadratic form, is good at separating smooth from non-smooth parts of graph signals; the goal here, however, is to force graph signals at each node to be smooth.

3. GRAPH SIGNAL INPAINTING VIA TOTAL VARIATION MINIMIZATION

In this section, we briefly review previous work on graph signal inpainting, which lays a foundation for the distributed algorithms shown later. For more details on graph signal inpainting, see [7, 8, 9]. We consider a corrupted graph signal measurement

$$\begin{bmatrix}\mathbf{t}_{\mathcal{M}}\\ \mathbf{t}_{\mathcal{U}}\end{bmatrix} \;=\; \mathbf{t}=\mathbf{x}+\mathbf{w} \;=\; \mathbf{x}+\begin{bmatrix}\mathbf{w}_{\mathcal{M}}\\ \mathbf{w}_{\mathcal{U}}\end{bmatrix}, \tag{3}$$

where $\mathbf{x}$ is a true graph signal, $\mathbf{t}_{\mathcal{M}} \in \mathbb{C}^M$ is the uncorrupted part of the signal and $\mathbf{t}_{\mathcal{U}} \in \mathbb{C}^{N-M}$ is the corrupted part, $\mathbf{w}_{\mathcal{M}}$ is noise with small variance, and $\mathbf{w}_{\mathcal{U}}$ is corruption. The goal of graph signal inpainting is to recover $\mathbf{x}$ from $\mathbf{t}$ by removing the corruption $\mathbf{w}$.

We assume that the true signal $\mathbf{x}$ in (3) has low variation with respect to the underlying graph, represented by the graph shift A; we recover the true graph signal by solving the following minimization problem.

$$\begin{aligned}\widehat{\mathbf{x}} \;&=\; \arg\min_{\mathbf{x}} \mathrm{S}_2(\mathbf{x}), && \text{(4a)}\\ \text{subject to}\quad & ||\mathbf{w}_{\mathcal{M}}||_2^2 \;\le\; \epsilon^2, && \text{(4b)}\\ & \mathbf{t}=\mathbf{x}+\mathbf{w}. && \text{(4c)}\end{aligned}$$

The condition (4b) controls the noise level of $\mathbf{w}_{\mathcal{M}}$.

3.1. Graph total variation regularization

To get a closed-form solution, the graph signal inpainting (4) can be formulated as an unconstrained problem as,

$$\widehat{\mathbf{x}} = \arg\min_{\mathbf{x}} \; ||\mathbf{x}_{\mathcal{M}} - \mathbf{t}_{\mathcal{M}}||_2^2 + \lambda\, \mathrm{S}_2(\mathbf{x}), \tag{5}$$

where the tuning parameter λ controls the trade-off between the two parts of the objective function. Since the quadratic form of graph total variation is a regularization term, (5) is called the regularization approach. The closed-form solution is as follows [7]:

$$\widehat{\mathbf{x}} = \left(\Phi_{\mathcal{M}} + \lambda\widetilde{\mathrm{A}}\right)^{-1}\widehat{\mathbf{t}}, \tag{6}$$

where $\Phi_{\mathcal{M}} = \begin{bmatrix}\mathrm{I}_{\mathcal{M}\mathcal{M}} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}$, $\widehat{\mathbf{t}} = \begin{bmatrix}\mathbf{t}_{\mathcal{M}}\\ \mathbf{0}\end{bmatrix}$, $\widetilde{\mathrm{A}} = (\mathrm{I}-\mathrm{A})^*(\mathrm{I}-\mathrm{A})$, and $*$ denotes the Hermitian transpose.

3.2. Graph total variation minimization

When the uncorrupted part of the signal needs to be preserved intact, we solve (4) directly for $\epsilon = 0$,

$$\begin{aligned}\widehat{\mathbf{x}} \;&=\; \arg\min_{\mathbf{x}} \mathrm{S}_2(\mathbf{x}), && \text{(7)}\\ \text{subject to}\quad & \mathbf{x}_{\mathcal{M}} = \mathbf{t}_{\mathcal{M}}.\end{aligned}$$

Since we focus on minimizing the quadratic form of graph total variation, (7) is called the minimization approach. By writing $\widetilde{\mathrm{A}}$ in a block form as

$$\widetilde{\mathrm{A}} = \begin{bmatrix}\widetilde{\mathrm{A}}_{\mathcal{M}\mathcal{M}} & \widetilde{\mathrm{A}}_{\mathcal{M}\mathcal{U}}\\ \widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}} & \widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}}\end{bmatrix},$$

the closed-form solution is as follows [7]:

$$\widehat{\mathbf{x}}_{\mathcal{U}} = -\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}}^{-1}\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}}. \tag{8}$$

4. DISTRIBUTED GRAPH TOTAL MINIMIZATION FOR SIGNAL INPAINTING

The closed-form solutions of both the regularization approach and the minimization approach requires inverting a matrix. When we deal with large graphs, that computation is expensive. To solve this, we propose the corresponding distributed and decentralized algorithms for both regularization and minimization approaches and show the properties of convergence.

4.1. Distributed graph total variation regularization

We propose the following algorithm as a distributed and decentralized counterpart of (5):

Algorithm 1. *(Distributed graph total variation regularization)*
Let $h(\mathrm{A}) = \mathrm{I} - \alpha(\Phi_{\mathcal{M}} + \lambda\widetilde{\mathrm{A}})$, with α such that $||h(\mathrm{A})||_2 < 1$. Set the intial conditions as $k = 0$, $\mathbf{x}^{(0)} = 0$, and do the following two steps iteratively:

$$\begin{aligned}\mathbf{x}^{(k+1)} \;&\leftarrow\; h(\mathrm{A})\mathbf{x}^{(k)} + \alpha\widehat{\mathbf{t}}, && \text{(9)}\\ k \;&\leftarrow\; k+1,\end{aligned}$$

until convergence.

In (9), the value of the nth node is updated as

$$\begin{aligned}\mathbf{x}_n^{(k+1)} \;&=\; \sum_{m=1}^{N} h(\mathrm{A})_{n,m}\mathbf{x}_m^{(k)} + \alpha\widehat{\mathbf{t}}_n\\ &=\; \sum_{m=1}^{N}\big((1-\alpha\lambda)\,\mathrm{I} - \alpha\Phi_{\mathcal{M}}\\ &\quad +\alpha(\mathrm{A}^* + \mathrm{A} - \mathrm{A}^*\,\mathrm{A})\big)_{n,m}\mathbf{x}_m^{(k)} + \alpha\widehat{\mathbf{t}}_n,\end{aligned}$$

where $(\mathrm{A}^*\,\mathrm{A})_{n,m} = \sum_i \mathrm{A}_{i,n}^*\,\mathrm{A}_{i,m}$. When the value of $\mathrm{A}_{n,m}$, or $\mathrm{A}_{n,m}^*$, is nonzero, there exists an edge between the nth and

mth node; when the value of $(\mathrm{A}^*\,\mathrm{A})_{n,m}$ is nonzero, there is at least one node connecting the nth and mth node. We thus see that when the geodesic distance between the nth node and the mth node is more than 2, $h(\mathrm{A})_{n,m}$ is always zero; in other words, the value of each node is updated by weighting the values of its local neighbors and adding a pre-stored constant. This algorithm can be computed by each node without transmitting signal coefficients to a computing center, which is distributed and decentralized.

The following theorem provide theoretical insight into Algorithm 1.

Theorem 1. *Let $k \to \infty$, where k is the iteration number. The solution of Algorithm 1 converges to* (6) *with the convergence error decreasing as $O(||h(\mathrm{A})||_2^k)$.*

Proof. From (9), we have

$$\begin{aligned}\mathbf{x}^{(k)} &= h(\mathrm{A})\mathbf{x}^{(k-1)} + \alpha\widehat{\mathbf{t}} \\ &\overset{(a)}{=} h(\mathrm{A})\left(h(\mathrm{A})\mathbf{x}^{(k-2)} + \alpha\widehat{\mathbf{t}}\right) + \alpha\widehat{\mathbf{t}} \\ &\overset{(b)}{=} \alpha\sum_{i=0}^{k-1} h(\mathrm{A})^i\widehat{\mathbf{t}}, \end{aligned} \tag{10}$$

where (a) follows (9) by setting k to $k-1$ and (b) comes from the induction. We then have

$$\begin{aligned}\lim_{k\to+\infty}\mathbf{x}^{(k)} &\overset{(a)}{=} \alpha\sum_{i=0}^{+\infty} h(\mathrm{A})^i\widehat{\mathbf{t}} \\ &\overset{(b)}{=} \alpha(\mathrm{I}-h(\mathrm{A}))^{-1}\widehat{\mathbf{t}} \\ &\overset{(c)}{=} \alpha(\alpha(\Phi_{\mathcal{M}} + \lambda\widetilde{\mathrm{A}}))^{-1}\widehat{\mathbf{t}} \\ &\overset{(d)}{=} (\Phi_{\mathcal{M}} + \lambda\widetilde{\mathrm{A}})^{-1}\widehat{\mathbf{t}}, \end{aligned}$$

where (a) follows (10), (b) follows the matrix inversion property, (c) follows the definition of $h(\mathrm{A})$ and (d) cancels α.

Denote $\mathbf{x} = \lim_{k\to+\infty}\mathbf{x}^{(k)}$, we have

$$\begin{aligned}||\mathbf{x}^{(k)} - \mathbf{x}||_2 &\overset{(a)}{=} ||\alpha\sum_{i=k}^{+\infty} h(\mathrm{A})^i\widehat{\mathbf{t}}||_2 \\ &= ||(h(\mathrm{A}))^k\alpha\sum_{i=0}^{+\infty} h(\mathrm{A})^i\widehat{\mathbf{t}}||_2 \\ &\overset{(b)}{=} ||(h(\mathrm{A}))^k\mathbf{x}||_2 \\ &\overset{(c)}{\leq} ||(h(\mathrm{A})||_2^k||\mathbf{x}||_2, \end{aligned}$$

where (a) and (b) follow (10), and (c) follows the definition of spectral norm. By rearranging the terms, we finally obtain

$$\frac{||\mathbf{x}^{(k)} - \mathbf{x}||_2}{||\mathbf{x}||_2} \leq ||(h(\mathrm{A})||_2^k.$$

□

4.2. Distributed graph total variation minimization

We propose the following algorithm as a distributed and decentralized counterpart of (7).

Algorithm 2. *(Distributed graph total variation minimization)*
Let $g(\mathrm{A}) = \mathrm{I} - \alpha\widetilde{\mathrm{A}}$, with α such that $||g(\mathrm{A})_{\mathcal{U}\mathcal{U}}||_2 < 1$. Set the intial conditions as $k = 0$, $\mathbf{x}_{\mathcal{M}}^{(0)} = \mathbf{t}_{\mathcal{M}}$, $\mathbf{x}_{\mathcal{U}}^{(0)} = 0$, and do the following three steps iteratively

$$\begin{aligned}\mathbf{x}_{\mathcal{M}}^{(k)} &\leftarrow \mathbf{t}_{\mathcal{M}}, \\ \mathbf{x}^{(k+1)} &\leftarrow g(\mathrm{A})\mathbf{x}^{(k)}, \\ k &\leftarrow k+1, \end{aligned} \tag{11}$$

until convergence.

In (11), the value of the nth node is updated as

$$\begin{aligned}\mathbf{x}_n^{(k+1)} &= \sum_{m=1}^{N} g(\mathrm{A})_{n,m}\mathbf{x}_m^{(k)} \\ &= \sum_{m=1}^{N} \left((1-\alpha)\,\mathrm{I} + \alpha(\mathrm{A}^* + \mathrm{A} - \mathrm{A}^*\,\mathrm{A})\right)_{n,m}\mathbf{x}_m^{(k)}, \end{aligned}$$

where $(\mathrm{A}^*\,\mathrm{A})_{n,m} = \sum_i \mathrm{A}^*_{i,n}\,\mathrm{A}_{i,m}$. Similarly $h(\mathrm{A})$ in (9), when the geodesic distance between the nth node and the mth node is more than 2, $g(\mathrm{A})_{n,m}$ is always zero. This algorithm is also distributed and decentralized.

The following theorem shows the sufficient condition for $||g(\mathrm{A})_{\mathcal{U}\mathcal{U}}||_2 \leq 1$, which provides a way of choosing α.

Theorem 2. *Let $0 < \alpha \leq 2/\lambda_{max}(\widetilde{\mathrm{A}})$. Then,*

$$||g(\mathrm{A})_{\mathcal{U}\mathcal{U}}||_2 \leq 1.$$

Proof. Since $\widetilde{\mathrm{A}}$ is real and symmetric, we decompose it as

$$\widetilde{\mathrm{A}} = \mathrm{U}\,\Lambda\,\mathrm{U}^*, \tag{12}$$

where U is an unitary matrix and Λ are a diagonal matrix whose diagonal elements are are the corresponding eigenvalues, denoted as λ_i; moreover, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N \geq 0$. Since $0 < \alpha \leq 2/\lambda_{max}(\widetilde{\mathrm{A}})$, we have

$$1 \geq 1 - \alpha\lambda_i \geq 1 - 2\lambda_i/\lambda_{max}(\widetilde{\mathrm{A}}) \geq -1. \tag{13}$$

We then bound the spectral norm of $g(\mathrm{A})_{\mathcal{U}\mathcal{U}}$ as

$$\begin{aligned}||g(\mathrm{A})_{\mathcal{U}\mathcal{U}}||_2 &\overset{(a)}{\leq} ||g(\mathrm{A})||_2 \\ &\overset{(b)}{=} ||\,\mathrm{I} - \alpha\widetilde{\mathrm{A}}||_2 \\ &\overset{(c)}{=} ||\,\mathrm{I} - \alpha\,\mathrm{U}\,\Lambda\,\mathrm{U}^*\,||_2 \\ &\overset{(d)}{=} ||\,\mathrm{U}(\mathrm{I} - \alpha\Lambda)\,\mathrm{U}^*\,||_2 \\ &\overset{(e)}{=} \max_i |1 - \alpha\lambda_i| \overset{(f)}{\leq} 1, \end{aligned}$$

where (a) comes from the property of spectral norm, (b) follows the definition of $g(\mathrm{A})$, (c) follows (12), (d) comes from the unitary property of U, (e) comes from the definition of spectral norm and (f) follows (13). □

The following theorem provides a theoretical insight into Algorithm 2.

Theorem 3. *Let $k \to \infty$, where k is the iteration number. The solution of Algorithm 2 converges to* (8) *with the convergence error decreasing as $O(||g(\mathrm{A})_{\mathcal{U}\mathcal{U}}||_2^k)$.*

Proof. From (11), we split $\mathbf{x}$ based on uncorrupted indices and corrupted indices as

$$\begin{bmatrix} \mathbf{x}_{\mathcal{M}}^{(k+1)} \\ \mathbf{x}_{\mathcal{U}}^{(k+1)} \end{bmatrix} = \begin{bmatrix} g(\mathrm{A})_{\mathcal{M}\mathcal{M}} & g(\mathrm{A})_{\mathcal{M}\mathcal{U}} \\ g(\mathrm{A})_{\mathcal{U}\mathcal{M}} & g(\mathrm{A})_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathcal{M}}^{(k)} \\ \mathbf{x}_{\mathcal{U}}^{(k)} \end{bmatrix}.$$

We focus on the uncorrupted indices of $\mathbf{x}$ and obtain

$$\begin{aligned} \mathbf{x}_{\mathcal{U}}^{(k+1)} &= g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{x}_{\mathcal{M}}^{(k)} + g(\mathrm{A})_{\mathcal{U}\mathcal{U}}\mathbf{x}_{\mathcal{U}}^{(k)} \\ &\overset{(a)}{=} \sum_{i=0}^{k} (g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^i g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{x}_{\mathcal{M}}^{(0)} \\ &\quad +(g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^{k+1}\mathbf{x}_{\mathcal{U}}^{(0)} \\ &\overset{(b)}{=} \sum_{i=0}^{k} (g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^i g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}}, \end{aligned} \tag{14}$$

where (a) comes from the induction and (b) comes from the initialization of setting the corrupted part to be zero. We then obtain

$$\begin{aligned} \lim_{k\to+\infty} \mathbf{x}_{\mathcal{U}}^{(k+1)} &= \lim_{k\to+\infty} \sum_{i=0}^{k} (g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^i g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}} \\ &\overset{(a)}{=} (\mathrm{I}_{\mathcal{U}\mathcal{U}} - g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^{-1} g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}} \\ &\overset{(b)}{=} (\alpha\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}})^{-1}(-\alpha\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}})\mathbf{t}_{\mathcal{M}} \\ &= -\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{U}}^{-1}\widetilde{\mathrm{A}}_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}}, \end{aligned}$$

where (a) comes from the property of the matrix inversion and (b) follows the definition of $g(\mathrm{A})$.

Denote $\mathbf{x}_{\mathcal{U}} = \lim_{k\to+\infty} \mathbf{x}_{\mathcal{U}}^{(k)}$,

$$\begin{aligned} ||\mathbf{x}_{\mathcal{U}}^{(k)} - \mathbf{x}_{\mathcal{U}}||_2 &\overset{(a)}{=} ||\sum_{i=k}^{+\infty} (g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^i g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}}||_2 \\ &= ||(g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^k \sum_{i=0}^{+\infty} (g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^i g(\mathrm{A})_{\mathcal{U}\mathcal{M}}\mathbf{t}_{\mathcal{M}}||_2 \\ &\overset{(b)}{=} ||(g(\mathrm{A})_{\mathcal{U}\mathcal{U}})^k \mathbf{x}_{\mathcal{U}}||_2 \\ &\overset{(c)}{\leq} ||(g(\mathrm{A})_{\mathcal{U}\mathcal{U}})||_2^k ||\mathbf{x}_{\mathcal{U}}||_2, \end{aligned}$$

where (a) and (b) follow (14), and (c) follows the definition of spectral norm. We finally obtain

$$\frac{||\mathbf{x}_{\mathcal{U}}^{(k)} - \mathbf{x}_{\mathcal{U}}||_2}{||\mathbf{x}_{\mathcal{U}}||_2} \leq ||(g(\mathrm{A})_{\mathcal{U}\mathcal{U}})||_2^k.$$

□

4.3. Discussion

The algorithms to solve graph total variation regularization as in (5) and graph total variation minimization as in (7) are distributed and decentralized because each node can be updated individually in each iteration without transmitting signal coefficients to a computing center, this, however, requires an infinite number steps to converge. Since both (5) and (7) are quadratic problems, one alternative is conjugate gradient descent [10], which produces the exact solution after a finite number of iterations. Conjugate gradient descent, however, is a centralized method and requires more storage space. Although it converges faster than the distributed algorithm, it takes more computation in each iteration. We compare the two versions in the following section.

5. EXPERIMENTAL RESULTS

We now apply the proposed algorithms to the classification of online blogs and temperature inpainting. We compare the proposed algorithms with centralized method, conjugate gradient descent. Note that in both cases, we build directed graph with asymmetric graph shift, which cannot be handled by graph Laplacian-based methods [5, 6].

As mentioned in 4.3, the conjugate gradient descent use all the information, and the distributed methods only use the local information. Therefore, the the conjugate gradient descent should converge faster, but the distributed methods compute separately at each node, which saves the storage space. Here the experiments verify that the convergence speed of the proposed distributed method catches up with that of the conjugate gradient descent when the graph trends to be regular.

5.1. Classification of online blogs

We consider the problem of classifying $N = 1224$ online political blogs as either conservative or liberal [11]. We represent conservative labels as $+1$ and liberal ones as -1. The blogs are represented by a graph in which nodes represent blogs, and directed graph edges correspond to hyperlink references between blogs. For a node v_n its outgoing edges have weights $1/\deg(v_n)$, where $\deg(v_n)$ is the out-degree of v_n (the number of outgoing edges).

We randomly labeled 1% of blogs and applied the graph total variation minimization to estimate the labels for remaining nodes. Estimated labels were thresholded around zero, so that positive values were set to $+1$ and negative to -1. We

set α to be $1/\lambda_{max}(\widetilde{\mathrm{A}})$. Note that the graph shift represents a scale-free graph whose degree distribution follows a power law.

In Figures 1 and 2, we compare the convergence performance between the proposed algorithm and the conjugate gradient descent. The convergence error is defined as

$$e^{(k)} = \frac{||\mathbf{x}^{(k)} - \mathbf{x}||_2^2}{||\mathbf{x}||_2},$$

where $\mathbf{x}^{(k)}$ is the result in the kth iteration and $\mathbf{x}$ is the closed-form solution. We see that conjugate gradient descent converges much faster than the proposed distributed algorithm. The reason is that some popular blogs connect most of the nodes and some other blogs are isolated from the others, which leads to a large $||g(\mathrm{A})||_2$. From Theorem 2, we know that a large $||g(\mathrm{A})||_2$ causes a slow convergence.

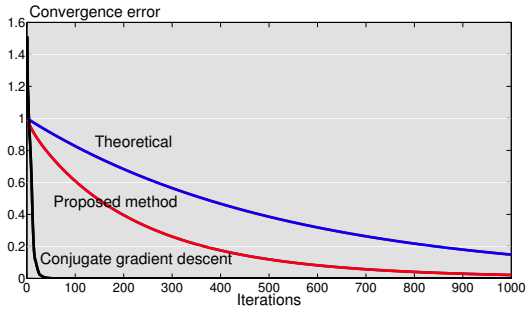

Fig. 1: Convergence error as a function of the iteration number for online blog classification.

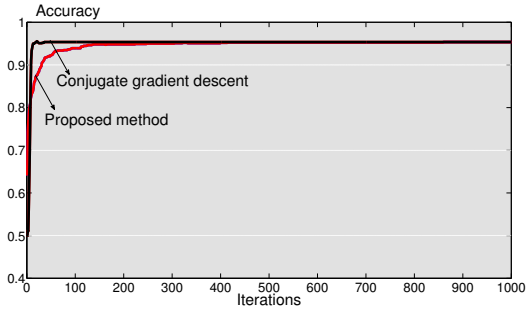

Fig. 2: Classification accuracy as a function of the iteration number for online blog classification.

5.2. Temperature inpainting

We consider 150 weather stations in United States that record their local temperatures [1, 12]. Each weather station has 365 records, in total of 54750 measurements. The graph representing these weather stations is obtained by measuring the geodesic distance between each pair of weather stations.

The nodes are represented by an 8-nearest neighbor graph, in which nodes represent weather stations, and each node is connected to eight other nodes that represent the closest weather stations. The graph shift A is constructed as $\mathrm{A}_{i,j} = \mathrm{P}_{i,j} \,/\, \sum_i \mathrm{P}_{i,j}$, where

$$\mathrm{P}_{i,j} = \exp - \frac{N^2 d_{i,j}}{\sum_{i,j} d_{i,j}},$$

where $d_{i,j}$ is the geodesic distances between the ith and the jth weather stations. Note that the graph shift represents a direct graph where each node has the same number of neighbors, which is more regular than the graph shift in Section 5.1.

For each of 365 recording, we randomly corrupted 50% of measurements and applied the graph total variation minimization to estimate those measurements. We set α to be $1/\lambda_{max}(\widetilde{\mathrm{A}})$. In Figures 3 and 4, we compare the convergence performance between the proposed algorithm and conjugate gradient descent. We see that the proposed algorithm converges by using almost the same iterations with conjugate gradient descent. The reason is that we build a regular graph that each node has the same degree, which leads to a small $||g(\mathrm{A})||_2$. From Theorem 2, we know that a small $||g(\mathrm{A})||_2$ causes a fast convergence. From these two experiments, we

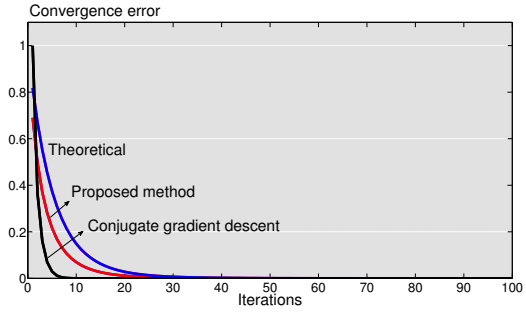

Fig. 3: Convergence error as a function of the iteration number for temperature inpainting.

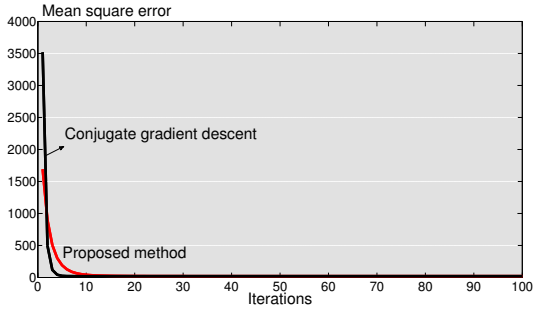

Fig. 4: Mean square error as a function of the iteration number for temperature inpainting.

conclude that on more regular graphs, the convergence speed of the proposed distributed and decentralized algorithm is almost as fast as the centralized method. Since a distributed algorithm does not require to collect data to a center, it is more practical and efficient.

6. CONCLUSION

In this paper, we presented distributed and decentralized algorithms for graph signal inpainting. The algorithms solve

graph signal inpainting by restricting each node to communicate only with its neighbors. We showed that the solution of the distributed algorithm converges to the closed-form solution with the theoretical convergence speed. Experiments on online blog classification and the temperature prediction suggest that the convergence speed of the proposed distributed algorithm is competitive with that of the centralized algorithm, especially when a graph tends to be regular. Since a distributed algorithm does not require to collect data to a center, it is more practical and efficient.

7. REFERENCES

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

[2] 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, 2014.

[3] 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 Process. Mag.*, vol. 30, pp. 83–98, 2013.

[4] A. Elmoataz, Olivier Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing," *IEEE Trans. Image Process.*, vol. 17, no. 7, pp. 1047–1060, 2008.

[5] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proc. ICML*, 2003, pp. 912–919.

[6] M. Belkin, P. Niyogi, and P. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples.," *J. Machine Learn. Research*, vol. 7, pp. 2399–2434, 2006.

[7] S. Chen, A. Sandryhaila, G. Lederman, Z. Wang, J. M. F. Moura, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević, "Signal inpainting on graphs via total variation minimization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Florence, Italy, May 2014.

[8] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal denoising on graphs via graph filtering," in *Proc. IEEE Glob. Conf. Signal Information Process.*, Atlanta, GA, Dec. 2014.

[9] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal recovery on graphs," *IEEE Trans. Signal Process.*, 2014, Submitted.

[10] Jonathan Richard Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Tech. Rep., Carnegie Mellon University, Aug. 1994.

[11] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 u.s. election: Divided they blog," in *Proc. LinkKDD*, 2005, pp. 36–43.

[12] NCDC NOAA 1981-2010 climate normals, ," 2011, ncdc.noaa.gov/oa/climate/normals/usnormals.html.