Communication-Efficient Decentralized Event Monitoring in Wireless Sensor Networks

Kun Yuan, Qing Ling, and Zhi Tian, *Fellow, IEEE*

Abstract—In this paper, we consider monitoring multiple events in a sensing field using a large-scale wireless sensor network (WSN). The goal is to develop communication-efficient algorithms that are scalable to the network size. Exploiting the sparse nature of the events, we formulate the event monitoring task as an ℓ_1 regularized nonnegative least squares problem where the optimization variable is a sparse vector representing the locations and magnitudes of events. Traditionally the problem can be reformulated by letting each sensor hold a local copy of the event vector and imposing consensus constraints on the local copies, and solved by decentralized algorithms such as the alternating direction method of multipliers (ADMM). This technique requires each sensor to exchange their estimates of the entire sparse vector and hence leads to high communication cost. Motivated by the observation that an event usually has limited influence range, we develop two communication-efficient decentralized algorithms, one is the partial consensus algorithm and the other is the Jacobi approach. In the partial consensus algorithm that is based on the ADMM, each sensor is responsible for recovering those events relevant to itself, and hence only consent with neighboring nodes on a part of the sparse vector. This strategy greatly reduces the amount of information exchanged among sensors. The Jacobi approach addresses the case that each sensor cares about the event occurring at its own position. Jacobi-like iterates are shown to be much faster than other algorithms, and incur minimal communication cost per iteration. Simulation results validate the effectiveness of the proposed algorithms and demonstrate the importance of proper modelling in designing communication-efficient decentralized algorithms.

Index Terms—Wireless sensor network (WSN), event monitoring, decentralized computation

1 Introduction

In recent years, wireless sensor networks (WSNs) have been widely applied to event monitoring tasks, which aim at discovering events of interest in sensing fields. Typical applications include target tracking [1], structural health monitoring (SHM) [2], field reconstruction [3], spectrum sensing [4], etc. Due to the easiness of deployment, WSNs are especially fit for applications in hazardous environments, such as detecting nuclear radioactive sources [5] and monitoring active volcanos [6].

One common problem arising from the event monitoring tasks is how to fuse the sensory measurements and obtain accurate information about the events occurring within the sensing field. An intuitive idea is to compare the sensory measurements with a predefined threshold; if the measurement of one sensor is larger than the threshold, the sensor reports a positive detection around its position. Performance of this binary detection approach is determined by the choice of the threshold and can be sensitive to the measurement noise [7]. Since one event may influence the measurements of multiple sensors, it is also necessary to address the spatial correlation in binary detection [8]. Another class of event monitoring algorithms take the statistical signal processing perspective by introducing prior knowledge. Examples include the expectation maximization algorithm in [9] and the Bayesian approach in [10], [11].

This paper makes use of the prior knowledge that events occurring within the sensing field are spatially sparse compared to their candidate positions (e.g., positions of the sensors or some grid points), which suggests to solve the event monitoring problem from the sparse optimization perspective. Specifically, we formulate the event monitoring task into the following ℓ_1 regularized nonnegative least squares problem:

$$\begin{aligned} \min_{\mathbf{c}} \quad & \frac{\lambda}{2}||\mathbf{H}\mathbf{c}-\mathbf{b}||_2^2 + ||\mathbf{c}||_1, \\ \text{s.t.} \quad & \mathbf{c} \geq 0. \end{aligned} \tag{1}$$

Herein, positions and amplitudes of the events are represented by a nonnegative decision variable $\mathbf{c}$ whose size is equal to the number of candidate positions. The vector $\mathbf{b}$ contains the noise-polluted measurements collected by sensors through the measurement matrix $\mathbf{H}$. In the objective function of (1), the least squares term $\|\mathbf{H}\mathbf{c}-\mathbf{b}\|_2^2$ corresponds to data fidelity and the ℓ_1 norm term $\|\mathbf{c}\|_1$ induces sparsity; the two terms are balanced through a nonnegative weight λ. Detailed description of the sparse optimization model (1) is given in Section 2. In this paper we will focus on developing communication-efficient decentralized algorithms to recover $\mathbf{c}$.

- *K. Yuan is with the Department of Automation, University of Science and Technology of China, Hefei, Anhui, China, 230027 and the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, Shannxi 710071, China. E-mail: kunyuan@mail.ustc.edu.cn.*
- *Q. Ling is with the Department of Automation, University of Science and Technology of China, Hefei, Anhui 230027, China. E-mail: qingling@mail.ustc.edu.cn.*
- *Z. Tian is with the Department of Electrical and Computer Engineering, Michigan Technological University, Hougton, MI 49931. E-mail: ztian@mtu.edu.*

Manuscript received 25 Nov. 2013; revised 29 July 2014; accepted 7 Aug. 2014. Date of publication 20 Aug. 2014; date of current version 6 July 2015. Recommended for acceptance by W.-Z. Song.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2014.2350474

Centralized techniques for solving (1), in which a fusion center collects all sensory measurements and estimates positions and magnitudes of the events, have been studied extensively. However, the centralized approach incurs a high communication cost due to extensive data transmission from the distributed sensors to the fusion center. Besides, breakdown of the fusion center or some critical relaying sensors (e.g., those close to the fusion center) may result in loss of data and even failure of the event monitoring task. The disadvantages of the centralized approach have motivated recent research interest in decentralized data processing [12], [13], [14], [15], [16], [17], [18]. The decentralized approach requires the sensors to solve the optimization problem in an autonomous way based on their local measurements, allowing them to collaborate only with their neighbors at a low communication cost. The sensors no longer need to transmit data to the fusion center as in the centralized approaches; hence the communication cost is scalable to the network size and the algorithm is robust to the dynamic network topology.

When the original centralized optimization problem is separable, it is possible to guarantee optimality of decentralized data processing by properly allocating the optimization task to individual sensors. One powerful tool to help formulate such a separable optimization problem is known as *consensus optimization*, where each sensor holds a local copy of the decision variable and the local copies of neighboring sensors are enforced to consent to the same value (see e.g., [15], [16], [17]). Thus, the consensus optimization problem is equivalent to the original one given that the WSN is connected. Many decentralized iterative algorithms have been proposed to solve the consensus optimization problem, such as the alternating direction method of multipliers (ADMM) [19], the distributed subgradient method [20], and the distributed dual averaging algorithm [21]. In each iteration of these algorithms the sensors need to exchange their current local copies of the decision variable. Therefore, the communication cost is proportional to the size of the decision variable. We call this scheme as *full consensus*. For the event monitoring problem (1), the size of the decision variable $\mathbf{c}$ is equal to the number of candidate positions that can be very large. Hence the full consensus scheme, which requires the sensors to exchange their current local copies of $\mathbf{c}$, is not communication-efficient.

To reduce the communication cost of decentralized event monitoring algorithms, one of the key issues is to reduce the amount of information exchanged per iteration. Note that the full consensus scheme implies that the entire decision variable $\mathbf{c}$ (i.e., positions and magnitudes of all the events) is relevant to all the sensors. However, in many event monitoring tasks, an individual sensor is only influenced by a portion of the events. Motivated by this fact, this paper first proposes a *partial consensus* scheme in which neighboring sensors only consent on their relevant events; this way, the dimensionality of information exchanged is largely reduced compared with the full consensus scheme. Obviously we can see that the full consensus scheme is a special case of partial consensus scheme.

In some event monitoring applications, each sensor is only responsible of recovering one of its relevant events. For example, if we choose positions of the sensors as candidate positions of the events, one sensor may only care about the event occurring at its own position. In this case the consensus schemes are no longer needed because no more than one sensor is required to recover the same event. We develop a decentralized Jacobi algorithm in which each sensor only needs to transmit one scalar, which represents its estimate on the magnitude of the event at its own position, to its neighbors. Apart from the low communication cost per iteration, the decentralized Jacobi approach can be further accelerated by the Nesterov acceleration technique and converges much faster than those using the consensus schemes. Therefore, overall communication cost is significantly reduced.

The rest of this paper is organized as follows. In Section 2, the event monitoring task is formulated as a sparse signal recovery problem with the form of ℓ_1 regularized nonnegative least squares. Section 3 introduces a full consensus algorithm and proposes a partial consensus algorithm. Considering the case that each sensor recovers its own corresponding event, Section 4 develops a decentralized Jacobi algorithm and analyzes its convergence properties. Performance of the proposed algorithms is shown in Section 5. Section 6 summarizes the paper.

2 Problem Formulation

Let us consider a wireless sensor network that is deployed in a two-dimensional area. The network has a set of L sensors, denoted as $\mathcal{L} = \{v_i\}_{i=1}^{L}$. Sensors have a common communication range r_C, which means any two sensors whose distance is within r_C can communicate directly. Suppose that d_{ij} is the distance from sensor v_i to sensor v_j. We define $\mathcal{N}_i^C = \{v_j: d_{ij} \le r_C,\ j \ne i\}$ that is the one-hop neighbor set of sensor v_i.

At each sampling time, multiple events may occur in the sensing field. To establish a tractable mathematical model for event monitoring we confine the sources of events to sensor points; that is, one event occurs only at a sensor point. For example, in the structural health monitoring problem [22], a WSN detects damages of a steel-frame structure. The sensors are deployed at the joints of the frame, and it is reasonable to assume that the damages also occur at the joints. If the source of one event coincides with the position of sensor v_i, we denote the magnitude of the event by a positive scalar c_i. If no event occurs at v_i, then $c_i = 0$. Therefore, we can formulate the problem as recovering the signal $\mathbf{c} = [c_1, \ldots, c_L]^T$ where $\cdot^T$ is the transposition operator. Although events could occur anywhere in the sensing field, it is a viable practice to confine event sources to sensor points, which is adequate to guarantee satisfactory detection accuracy when the sensors are densely or appropriately deployed.

Suppose that the influence of a unit-magnitude event at sensor point v_j on the sensor point v_i is h_{ij}. We consider such monitoring tasks where the measurement of one sensor can be represented as the superposition of the influences of all events plus random noise. For example, the measurement of sensor v_i is $b_i = \sum_{v_j \in \mathcal{L}} h_{ij} c_j + e_i$, in which e_i is random noise.

Now we are ready to adopt a least squares formulation to recover $\mathbf{c}$:

$$\min_{\{c_i\}} \quad \sum_{i=1}^{L}\left(b_i - \sum_{j=1}^{L} h_{ij}c_j\right)^2,$$
$$\text{s.t.} \quad c_i \geq 0, \quad i = 1, 2, \ldots, L,$$

or equivalently in a matrix form:

$$\min_{\mathbf{c}} \quad ||\mathbf{Hc} - \mathbf{b}||_2^2,$$
$$\text{s.t.} \quad \mathbf{c} \geq 0.$$

Here the measurement vector $\mathbf{b} = [b_1, \ldots, b_L]^T$ and the ith row of the measurement matrix $\mathbf{H}$ is $\mathbf{h}_i^T = [h_{i1}, \ldots, h_{iL}]$.

The least squares formulation ignores the sparsity of the vector $\mathbf{c}$. Note that in a large-scale network, the number of events is generally much smaller than the number of sensors; hence the vector $\mathbf{c}$ has a large amount of zero elements. Without considering this prior knowledge, the least squares formulation will lead to a non-sparse solution, which means a non-negligible number of false alarms. Motivated by this fact, we formulate the ℓ_1 regularized nonnegative least squares problem in (1):

$$\min_{\mathbf{c}} \quad \frac{\lambda}{2}||\mathbf{Hc} - \mathbf{b}||_2^2 + ||\mathbf{c}||_1,$$
$$\text{s.t.} \quad \mathbf{c} \geq 0.$$

Note that (1) shares similarity with the basis pursuit denoising (BPDN) model [23] and the least absolute shrinkage and selection operator (LASSO) model [24], but the measurement matrix $\mathbf{H}$ and the decision variable $\mathbf{c}$ are confined to be nonnegative. Such a formulation arises in WSN applications, e.g., detecting footsteps and vehicles using seismic sensors [25] and localizing shooters using acoustic sensors [26].

Our goal is to develop communication-efficient decentralized algorithms to solve the event monitoring model (1). With the consensus technique, the decentralized algorithms introduced in Section 1 (such as the ADMM, the distributed subgradient method, the distributed dual averaging algorithm) can be applied to this problem at the cost of considerable communication cost. Nevertheless, the following observation from many real-world applications enables us to reduce the communication cost and improve the energy efficiency.

Observation. An event has *partial influence*.

Recall that the influence of an event at sensor point v_j on the sensor point v_i is h_{ij}. In many applications we observe that there exists a constant r_E, which denotes the influence range of an event, such that $h_{ij} = 0$ if $d_{ij} > r_E$. Therefore, for an event occurring at sensor point v_i we define its influence set $\mathcal{N}_i^E = \{v_j : d_{ij} \leq r_E\}$ that contains all sensors whose measurements are influenced by the event occurring at sensor point v_i. Note that the one-hop neighbor set $\mathcal{N}_i^C$ is different from the influence set $\mathcal{N}_i^E$. If we adjust the communication range r_C such that $r_C = r_E$, then $\mathcal{N}_i^E = \mathcal{N}_i^C \cup v_i$.

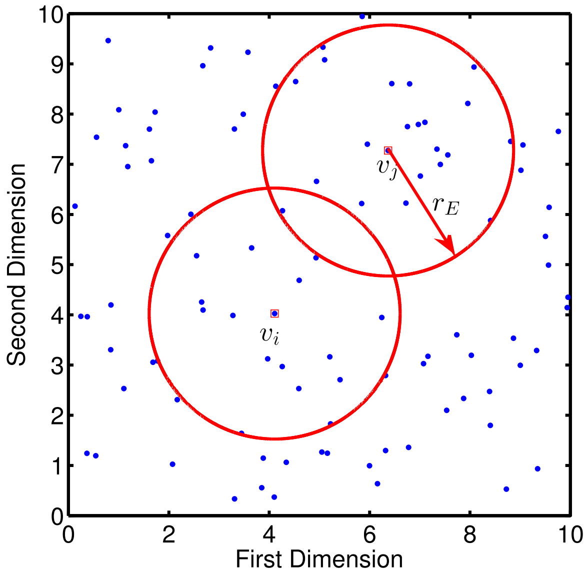


Fig. 1. The points indicate 100 temperature sensors randomly deployed in a two-dimensional area; the squares indicate two fire sources occurring at sensor positions v_i and v_j; the circles indicate the influence ranges r_E of the fire sources.

The phenomenon of partial influence can be observed in many applications, e.g., footstep and vehicle detection and acoustic source monitoring. More examples include fire source monitoring, target tracking, and nuclear radioactive detection, in which the influence of a source often decreases polynomially as the distance increases. Take the fire source monitoring application as an example. Suppose that 100 temperature sensors are randomly deployed in the sensing field as illustrated in Fig. 1. Two fire sources occur at v_i and v_j and they have the same influence ranges r_E. Therefore, only the sensors within the ranges can measure the high temperatures caused by the fire sources, while the sensors outside of the ranges are not influenced.

3 The Partial Consensus Algorithm Based on the ADMM

In this section, we present a partial consensus algorithm to solve the event monitoring problem (1) based the ADMM. We first introduce the full consensus algorithm that requires the sensors to exchange the entire decision variables. Motivated by the *partial influence* phenomenon introduced in Section 2 and detailed in Section 3.2 below, we improve the full consensus model and propose a partial consensus model for event monitoring. We develop a partial consensus algorithm based on the ADMM and analyze its convergence. Through reducing the amount of information exchanged per iteration, the partial consensus algorithm outperforms the full consensus algorithm in terms of communication efficiency.

Notations. Curlicue letters denote index sets. Given a column vector $\mathbf{a}$, $\mathbf{a}_{\mathcal{I}}$ denotes its projection onto the index set $\mathcal{I}$, i.e., stacking its elements $\{a_i\colon i \in \mathcal{I}\}$ to form $\mathbf{a}_{\mathcal{I}}$. Given a matrix $\mathbf{A}$, $\mathbf{A}_{(\mathcal{J},\mathcal{K})}$ denotes its projection onto the index set of rows $\mathcal{J}$ and the index set of columns $\mathcal{K}$, i.e., stacking its elements $\{a_{jk}\colon j \in \mathcal{J}, k \in \mathcal{K}\}$ to form $\mathbf{A}_{(\mathcal{J},\mathcal{K})}$.

3.1 Full Consensus Algorithm Based on ADMM

We first rewrite (1) in its unconstrained form

$$\min_{\mathbf{c}} \quad \frac{\lambda}{2}||\mathbf{Hc}-\mathbf{b}||_2^2 + ||\mathbf{c}||_1 + L\mathbb{I}_+(\mathbf{c}), \tag{2}$$

where L is the number of sensors and $\mathbb{I}_+(\mathbf{c})$ is an indicator function that equals to 0 when $\mathbf{c} \geq \mathbf{0}$ and $+\infty$ otherwise. Further separating the least squares term, (2) is equivalent to

$$\min_{\mathbf{c}} \quad \sum_{i=1}^{L}\left[\frac{\lambda}{2}\left(\mathbf{h}_i^T\mathbf{c}-b_i\right)_2^2 + c_i + \mathbb{I}_+(\mathbf{c})\right], \tag{3}$$

where $\mathbf{h}_i^T$ is the ith row of $\mathbf{H}$.

Letting $f_i(\mathbf{c}) = \frac{\lambda}{2}(\mathbf{h}_i^T\mathbf{c}-b_i)_2^2 + c_i + \mathbb{I}_+(\mathbf{c})$, we have the optimization problem

$$\min_{\mathbf{c}} \quad \sum_{i=1}^{L} f_i(\mathbf{c}), \tag{4}$$

whose objective function is separable with regard to the individual sensors and the decision variable is common. The full consensus technique introduces local copies of $\mathbf{c}$ at the sensors, by imposing consensus constraints on neighboring local copies (c.f., [15], [16], [17]), and reformulate (4) as follows:

$$\begin{aligned} \min_{\{\mathbf{c}^{(i)}\}} \quad & \sum_{i=1}^{L} f_i(\mathbf{c}^{(i)}), \\ \text{s.t.} \quad & \mathbf{c}^{(i)} = \mathbf{c}^{(j)}, \quad \forall v_i \in \mathcal{L},\ \forall v_j \in \mathcal{N}_i^C. \end{aligned} \tag{5}$$

Here $\mathbf{c}^{(i)}$ denotes the local copy of $\mathbf{c}$ at sensor v_i. The consensus constraint $\mathbf{c}^{(i)} = \mathbf{c}^{(j)}$ forces v_i and v_j, if they are one-hop neighbors, to consent on the value of their local copies. Apparently, (5) is equivalent to (4) if the WSN is connected.

We omit derivation of the full consensus algorithm based on the ADMM. Readers are referred to [17], or Section 3.2 that derives the ADMM for a more general case of partial consensus. For $f_i(\mathbf{c}) = \frac{\lambda}{2}(\mathbf{h}_i^T\mathbf{c}-b_i)_2^2 + c_i + \mathbb{I}_+(\mathbf{c})$, we have the following recursion at sensor v_i:

$$\begin{aligned} \mathbf{c}^{(i)}(t+1) = \arg\min_{\mathbf{c}^{(i)}\geq \mathbf{0}} \Bigg\{ & \frac{1}{2}\ \mathbf{c}^{(i)T}\left[\lambda\left(\mathbf{h}_i\mathbf{h}_i^T\right) + 2p\,|\mathcal{N}_i^C\,|\,\mathbf{I}\right]\mathbf{c}^{(i)} \\ & + \Bigg[\mathbf{e}_i - \lambda b_i\mathbf{h}_i + \boldsymbol{\alpha}_i(t) - p\,|\mathcal{N}_i^C\,|\,\mathbf{c}^{(i)}(t) \\ & - p\sum_{j\in\mathcal{N}_i^C}\mathbf{c}^{(j)}(t)\Bigg]^T\mathbf{c}^{(i)}\Bigg\}, \end{aligned} \tag{6}$$

$$\boldsymbol{\alpha}_i(t+1) = \boldsymbol{\alpha}_i(t) + p\,|\mathcal{N}_i^C|\mathbf{c}^{(i)}(t+1) - p\sum_{j\in\mathcal{N}_i^C}\mathbf{c}^{(j)}(t+1). \tag{7}$$

Here $\boldsymbol{\alpha}_i$ is an $L \times 1$ vector held by sensor v_i, $\mathbf{e}_i$ is the ith column of an $L \times L$ identity matrix, p is a positive constant, and $|\cdot|$ denotes the cardinality.

The full consensus algorithm based on the ADMM for event monitoring is shown in Algorithm 1. The algorithm guarantees convergence to the optimal solution of (5) according to the convergence analysis of the ADMM [19]. As shown in Algorithm 1, at each iteration sensor v_i transmits $\mathbf{c}^{(i)}$ to its neighbors and hence the communication cost per iteration is L. Therefore, the communication cost per iteration per sensor of the WSN is $L\sum_{i=1}^{L}|\mathcal{N}_i^C|$.

Algorithm 1. The Full Consensus Algorithm Based on the ADMM at Sensor v_i

Require: One-hop neighbor set $\mathcal{N}_i^C$, local data $\mathbf{h}_i$ and b_i.
1: Initialize $\mathbf{c}^{(i)}$ and $\boldsymbol{\alpha}_i$ as $\mathbf{0}$;
2: **for** $t = 0, 1, 2, \ldots$ **do**
3: Compute $\mathbf{c}^{(i)}(t+1)$ according to (6);
4: Transmit $\mathbf{c}^{(i)}(t+1)$ to and receive $\mathbf{c}^{(j)}(t+1)$ from $\mathcal{N}_i^C$;
5: Compute $\boldsymbol{\alpha}_i(t+1)$ according to (7);
6: **end for**
7: Return $\mathbf{c}^{(i)}(t+1)$.

Remark 1. The full consensus formulation (5) implies that the entire vector $\mathbf{c}$ is needed by all sensors to fulfill the monitoring task and/or the entire $\mathbf{c}$ is necessary in the optimization process; we call this phenomenon as *full influence*. To further illustrate this phenomenon, we can see that (5) is equivalent to

$$\begin{aligned} \min_{\{\mathbf{c}^{(i)}\}} \quad & \sum_{i=1}^{L} f_i(\mathbf{c}^{(i)}), \\ \text{s.t.} \quad & c_k^{(i)} = c_k^{(j)}, \quad \forall v_i \in \mathcal{L}, \forall v_j \in \mathcal{N}_i^C, \forall k, \end{aligned} \tag{8}$$

which indicates that to estimate c_k, the kth element of $\mathbf{c}$, any two neighboring sensors v_i and v_j need to consent on the value of its local copies $c_k^{(i)}$ and $c_k^{(j)}$. If in practice the events only have *partial influence*, i.e., an event only influences a portion of sensors, the full consensus algorithm is not communication-efficient. This fact motivates us to develop the partial consensus model and algorithm.

3.2 The Partial Consensus Model

The partial influence phenomenon indicates that for an event occurring at sensor point v_i, only a subset of sensors, denoted by $\mathcal{N}_i^E$, are influenced. All the sensors in $\mathcal{N}_i^E$ contribute relevant information to event estimation, but the sensors not in $\mathcal{N}_i^E$ are not necessary to participate. Therefore, we can let neighboring sensors, which are influenced by a common event, to consent on its value. This way, we are able to avoid the communication cost brought by consenting on the entire decision vector.

Define $\mathcal{J}_i = \{k\colon v_i \in \mathcal{N}_k^E, \forall k = 1, \ldots, L\}$ as the set of events that, if occur, will influence sensor v_i. For any possible event k (an event that might occur at the sensor point v_k), if $k \in \mathcal{J}_i$ then sensor v_i generates a local copy of c_k, denoted by $c_k^{(i)}$. Further, for any sensor v_j that is a one-hop neighbor of sensor v_i and influenced by the event k as well (i.e., $k \in \mathcal{J}_j$), sensors v_i and v_j must consent on the value of c_k. Therefore, the full consensus model (5) can be modified to the following partial consensus model:

$$\begin{aligned} \min_{\{\mathbf{c}_{\mathcal{J}_i}^{(i)}\}} \quad & \sum_{i=1}^{L} \tilde{f}_i(\mathbf{c}_{\mathcal{J}_i}^{(i)}), \\ \text{s.t.} \quad & c_k^{(i)} = c_k^{(j)}, \forall v_i \in \mathcal{N}_k^E, \quad \forall v_j \in \mathcal{N}_k^E \cap \mathcal{N}_i^C, \forall k. \end{aligned} \tag{9}$$

Here $\mathbf{c}^{(i)}_{\mathcal{J}_i}$ stacks the local copy of c_k at sensor v_i for all $k \in \mathcal{J}_i$ and $\tilde{f}_i(\mathbf{c}_{\mathcal{J}_i}) = \frac{\lambda}{2}\left((\mathbf{h}_i)^T_{\mathcal{J}_i}\mathbf{c}_{\mathcal{J}_i} - b_i\right)^2 + c_i + \mathbb{I}_+(\mathbf{c}_{\mathcal{J}_i})$; with a slight abuse of notation, $\mathbb{I}_+(\mathbf{c}_{\mathcal{J}_i})$ is an indicator function that equals to 0 when $\mathbf{c}_{\mathcal{J}_i} \geq \mathbf{0}$ and $+\infty$ otherwise. The following proposition shows that under certain conditions the partial consensus model (9) is equivalent to the centralized one (4).

Proposition 1. *Suppose that the partial influence phenomenon holds, i.e., $h_{ik} = 0$ if $v_i \notin \mathcal{N}^E_k$. Then the partial consensus model (9) is equivalent to the centralized one (4) in the sense that $c^{(i)}_k = c_k$ when $i \in \mathcal{N}^E_k$, if the subnetwork consisting of all sensors in $\mathcal{N}^E_k$ is connected for any k.*

Proof. Since for any k the subnetwork consisting of all sensors in $\mathcal{N}^E_k$ is connected, the consensus constraints in (9) force all $c^{(i)}_k$ to be equal if $v_i \in \mathcal{N}^E_k$. On the other hand, the function $f_i(\mathbf{c}^{(i)})$ defined in (4) is irrelevant with $c^{(i)}_k$ if $v_i \notin \mathcal{N}^E_k$ because $h_{ik} = 0$ in this case; therefore, $\tilde{f}_i(\mathbf{c}^{(i)}_{\mathcal{J}_i})$ defined in (9) is equal to $f_i(\mathbf{c}^{(i)})$. These two facts guarantee equivalence of (4) and (9). □

It is obvious that the full influence model (5) is a special case of the partial influence model (9). When it holds $\mathcal{N}^E_k = \mathcal{L}$ for any k, (9) degenerates to (5).

3.3 Decentralized Partial Consensus Algorithm

Through applying the ADMM to solve (9), the recursion at sensor v_i is

$$\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1) = \arg\min_{\mathbf{c}^{(i)}_{\mathcal{J}_i}} \Bigg\{ \tilde{f}_i(\mathbf{c}^{(i)}_{\mathcal{J}_i}) + \sum_{k\in\mathcal{J}_i}\Bigg(p\,|\mathcal{N}^E_k \cap \mathcal{N}^C_i|\,(c^{(i)}_k)^2 + \Bigg(\alpha_{ik}(t) - p\sum_{v_j\in\mathcal{N}^E_k\cap\mathcal{N}^C_i}\left(c^{(i)}_k(t) + c^{(j)}_k(t)\right)\Bigg)c^{(i)}_k\Bigg)\Bigg\}, \tag{10}$$

$$\alpha_{ik}(t+1) = \alpha_{ik}(t) + p\Bigg(\left|\mathcal{N}^E_k \cap \mathcal{N}_i\right| c^{(i)}_k(t+1) - \sum_{v_j\in\mathcal{N}^E_k\cap\mathcal{N}^C_i} c^{(j)}_k(t+1)\Bigg), \forall k \in \mathcal{J}_i. \tag{11}$$

Derivation of (10) and (11) can be found in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2350474.

Recall that $\tilde{f}_i(\mathbf{c}_{\mathcal{J}_i}) = \frac{\lambda}{2}\left((\mathbf{h}_i)^T_{\mathcal{J}_i}\mathbf{c}_{\mathcal{J}_i} - b_i\right)^2 + c_i + \mathbb{I}_+(\mathbf{c}_{\mathcal{J}_i})$ in the event monitoring application. Define $\mathbf{D}_i$ as a diagonal matrix whose kth diagonal element is $|\mathcal{N}^E_k \cap \mathcal{N}^C_i|$ and $\mathbf{g}_i(t)$ as a column vector whose kth element is $\sum_{j\in\mathcal{N}^E_k\cap\mathcal{N}^C_i} c^{(j)}_k(t)$. Substituting $\tilde{f}_i(\mathbf{c})$, $\mathbf{D}_i$ and $\mathbf{g}_i$ into (10), we get

$$\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1) = \arg\min_{\mathbf{c}^{(i)}_{\mathcal{J}_i}\geq\mathbf{0}} \Bigg\{ \frac{1}{2}\,\mathbf{c}^{(i)T}_{\mathcal{J}_i}\left[\lambda(\mathbf{H}_i\mathbf{h}^T_i)_{(\mathcal{J}_i,\mathcal{J}_i)} + 2p(\mathbf{D}_i)_{(\mathcal{J}_i,\mathcal{J}_i)}\right]\mathbf{c}^{(i)}_{\mathcal{J}_i} + \left[(\mathbf{e}_i)_{\mathcal{J}_i} - \lambda b_i(\mathbf{H}_i)_{\mathcal{J}_i} + \boldsymbol{\alpha}_{\mathcal{J}_i}(t) - p(\mathbf{D}_i)_{(\mathcal{J}_i,\mathcal{J}_i)}\mathbf{c}^{(i)}_{\mathcal{J}_i}(t) - p(\mathbf{g}_i(t))_{\mathcal{J}_i}\right]^T\mathbf{c}^{(i)}_{\mathcal{J}_i}\Bigg\}, \tag{12}$$

where $\mathbf{e}_i$ is the ith column of an $L \times L$ identity matrix. And $\boldsymbol{\alpha}_{\mathcal{J}_i}$ is updated through:

$$\boldsymbol{\alpha}_{\mathcal{J}_i}(t+1) = \boldsymbol{\alpha}_{\mathcal{J}_i}(t) + p(\mathbf{D}_i)_{(\mathcal{J}_i,\mathcal{J}_i)}\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1) - p(\mathbf{g}_i(t+1))_{\mathcal{J}_i}, \tag{13}$$

where $\boldsymbol{\alpha}_{\mathcal{J}_i}$ is the vector catenating all $\alpha_{ik},\ \forall k \in \mathcal{J}_i$.

The Hessian matrix of the objective function in (12) equals to $\lambda(\mathbf{H}_i\mathbf{h}^T_i)_{(\mathcal{J}_i,\mathcal{J}_i)} + 2p(\mathbf{D}_i)_{(\mathcal{J}_i,\mathcal{J}_i)}$, which is positive definite because $\mathbf{D}_i$'s diagonal elements are positive. The Hessian matrix is of size $|\mathcal{J}_i| \times |\mathcal{J}_i|$, which is far less than that in the full consensus case (i.e., $L \times L$). This property largely reduces the computation cost on each sensor. The full consensus algorithms (6) and (7) is a special case of the partial consensus algorithms (12) and (13) when $\mathcal{N}^E_i = \mathcal{L}$.

The partial consensus algorithm based on the ADMM is outlined in Algorithm 2. Now we consider its implementation. In the beginning, each sensor v_i broadcasts **HELLO** to all the sensors. Say sensor v_j is a one-hop neighbor of v_i. When v_j receives **HELLO**, it feedbacks **ECHO**. After sensor v_i receives **ECHO** from v_j, it recognizes v_j as a one-hop neighbor and puts v_j into the one-hop neighbor set $\mathcal{N}^C_i$. To know $\mathcal{N}^E_i$ usually we need to estimate the influence range r_E through experiments. Sensor v_j with $d_{ij} \leq r_E$ belongs to the influence set $\mathcal{N}^E_i$. The distance between two sensors can be measured via various methods, such as time of arrival (TOA), time difference of arrival (TDOA), or received signal strength indicator (RSSI) [27].

Algorithm 2. The Partial Consensus Algorithm Based on the ADMM at Sensor v_i

Require: One-hop neighbor set $\mathcal{N}^C_i$, influence set $\mathcal{N}^E_i$ and index set $\mathcal{J}_i$, local data $\mathbf{h}_i$ and b_i.

1: Initialize $\mathbf{c}^{(i)}$, $\mathbf{g}_i$, and $\boldsymbol{\alpha}_{\mathcal{J}_i}$ as $\mathbf{0}$;
2: **for** $t = 0, 1, 2, \ldots$, sensor v_i **do**
3: Compute $\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1)$ according to (12);
4: Transmit $c^{(i)}_k(t+1)$ to, and receive $c^{(j)}_k(t+1)$ from $v_j \in \mathcal{N}^E_k \cap \mathcal{N}^C_i$, $\forall k \in \mathcal{J}_i$;
5: Construct $\mathbf{g}_i(t+1)$ and compute $\boldsymbol{\alpha}_i(t+1)$ according to (13);
6: **end for**
7: Return $\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1)$.

In the partial consensus algorithm, sensor v_i needs to collect $\sum_{k\in\mathcal{J}_i}|\mathcal{N}^E_k \cap \mathcal{N}^C_i|$ scalar values to update $\mathbf{c}^{(i)}_{\mathcal{J}_i}(t+1)$, and hence the total communication cost per iteration is $\sum_{v_i\in\mathcal{L}}\sum_{k\in\mathcal{J}_i}|\mathcal{N}^E_k \cap \mathcal{N}^C_i|$. In contrast, in the full consensus algorithm the overall communication cost per iteration is $L\sum_{i=1}^{L}|\mathcal{N}^C_i|$. Obviously, the size of estimated variables needed to be exchanged among neighbors per iteration partial consensus is much less than that using full consensus. The advantage of using partial consensus to reduce communication cost is also discussed in [28], which focuses on the application of model predictive control. Note that [28] requires the network to be bipartite and this paper considers an arbitrary connected network.

4 THE JACOBI APPROACH

The partial consensus algorithm considerably reduces the communication cost per iteration compared to the full consensus algorithm. When the event influence range r_E becomes smaller, each sensor recovers a smaller number of events and the resulting communication cost is lighter. However, there still exists redundant communication since each sensor is responsible of recovering events occurring at nearby sensor points and neighboring sensors need to consent on relevant events.

We observe that when the events are localized such that the influence range is no larger than the communication range (i.e., $r_E \le r_C$), each sensor can only recover the event occurring at its own point, not others. This way, consensus is no longer necessary. Under this condition, this section first proposes a decentralized Jacobi approach to solve (1) and then develops its accelerated version.

4.1 The Projected Jacobi (PJ) Approach

Since $\mathbf{c} \ge 0$, (1) can be rewritten as

$$\begin{aligned} \min_{\mathbf{c}} \quad & \frac{1}{2}\mathbf{c}^T\mathbf{P}\mathbf{c} + \mathbf{r}^T\mathbf{c}, \\ \text{s.t.} \quad & \mathbf{c} \ge 0, \end{aligned} \tag{14}$$

where $\mathbf{P} = \lambda\mathbf{H}^T\mathbf{H}$ and $\mathbf{r} = \mathbf{1} - \lambda\mathbf{H}\mathbf{b}$ with $\mathbf{1} = [1; \cdots; 1]$ being an $L \times 1$ vector.

We solve (14) through an iterative projected Jacobi approach:

$$\mathbf{c}(t+1) = [\mathbf{c}(t) - \gamma\mathbf{M}^{-1}(\mathbf{P}\mathbf{c}(t) + \mathbf{r})]^+, \tag{15}$$

where $\mathbf{M}$ is a diagonal matrix whose diagonal elements equal to the corresponding diagonal elements of $\mathbf{P}$ and γ is a positive stepsize. Note that $\mathbf{P}\mathbf{c}(t) + \mathbf{r}$ is the gradient of the objective function of (14) at $\mathbf{c} = \mathbf{c}(t)$. Therefore, (15) can be viewed as the projected gradient descent method where the gradient is scaled by $\mathbf{M}^{-1}$. The following proposition provides a sufficient condition for the convergence of the projected Jacobi approach.

Proposition 2. *The projected Jacobi approach with the recursion (15) converges to the optimal solution of (14) if* $\gamma \in (0, 2/L)$.

Proof. Since $\mathbf{P} = \lambda\mathbf{H}^T\mathbf{H}$, $\mathbf{P}$ is positive semidefinite. Under such a condition, the projected Jacobi approach with the recursion (15) converges to the optimal solution of (14) (see page 261 in [19]). □

Proposition 2 indicates that a small stepsize γ assures convergence. However, $\gamma \in (0, 2/L)$ might be too conservative and could lead to slow convergence. Since Proposition 2 only gives a sufficient condition, we often tune γ to be a larger value in practice.

Next we show that when the partial influence phenomenon holds and the influence range r_E is no larger than the communication range r_C, the recursion (15) can be implemented in a decentralized manner. To this end, we define $\mathbf{u}(t) = \mathbf{H}\mathbf{c}(t)$, and $\mathbf{v}(t) = \mathbf{M}^{-1}(\lambda\mathbf{H}^T\mathbf{u}(t) + \mathbf{r})$. Since $h_{ij} = 0$ if $j \notin \mathcal{N}_i^E$, the recursion (15) is equivalent to

$$u_i(t+1) = \mathbf{h}_i^T\mathbf{c}(t) = \sum_{j\in\mathcal{N}_i^E} h_{ij}c_j(t), \tag{16a}$$

$$v_i(t+1) = \frac{1 - \lambda\mathbf{h}_i^T\mathbf{b} + \lambda\mathbf{h}_i^T\mathbf{u}(t+1)}{\lambda\mathbf{h}_i^T\mathbf{h}_i} \tag{16b}$$

$$= \frac{r_i + \lambda\sum_{j\in\mathcal{N}_i^E} h_{ij}u_j(t+1)}{\lambda m_i}, \tag{16c}$$

$$c_i(t+1) = [c_i(t) - \gamma v_i(t+1)]^+. \tag{16d}$$

Here u_i, v_i, r_i are the ith elements of $\mathbf{u}$, $\mathbf{v}$, $\mathbf{r}$, respectively; $m_i = \mathbf{h}_i^T\mathbf{h}_i$ is the ith diagonal element of $\mathbf{M}$. Note that by definition $r_i = 1 - \lambda\sum_{j\in\mathcal{N}_i^E} h_{ij}b_j$.

Note that $c_j(t)$ and $u_j(t)$ can be collected by v_i if $j \in \mathcal{N}_i^E$ since $r_E \le r_C$. This way, the recursion (16) is naturally decentralized as we outlined in Algorithm 3. For the update at time $t+1$, sensor v_i needs to collect $\{c_j(t), j \in \mathcal{N}_i^E\}$ and $\{u_j(t+1), j \in \mathcal{N}_i^E\}$. Therefore, its communication cost per iteration is $2|\mathcal{N}_i^E|$ and the total communication cost per iteration of the WSN is $2\sum_{v_i\in\mathcal{L}} |\mathcal{N}_i^E|$. Recall that for the partial consensus algorithm (see Section 3), the overall communication cost per iteration is $\sum_{v_i\in\mathcal{L}}\sum_{k\in\mathcal{J}_i} |\mathcal{N}_k^E \cap \mathcal{N}_i^C|$. For dense networks we have $2|\mathcal{N}_i^E| \le \sum_{k\in\mathcal{J}_i} |\mathcal{N}_k^E \cap \mathcal{N}_i^C|$. Furthermore, the projected Jacobi approach often converges faster than the partial consensus algorithm. Therefore, the projected Jacobi approach is more communication-efficient in each iteration.

Algorithm 3. The Projected Jacobi Approach at sensor v_i

Require: Influence set $\mathcal{N}_i^E$ and index set $\mathcal{J}_i$, local data $\mathbf{h}_i$ and b_i.

1: Transmit b_i to and receive b_j from $v_j \in \mathcal{N}_i^E$. Calculate $r_i = 1 - \lambda\sum_{j\in\mathcal{N}_i^E} h_{ij}b_j$ and $m_i = \mathbf{h}_i^T\mathbf{h}_i$;
2: Initialize c_i as 0;
3: **for** $t = 0, 1, 2, \ldots$, sensor v_i **do**
4: Compute $u_i(t+1)$ according to (16a). Transmit $u_i(t+1)$ to and receive $u_j(t+1)$ from $j \in \mathcal{N}_i^E$;
5: Compute $v_i(t+1)$ according to (16c);
6: Compute $c_i(t+1)$ according to (16d). Transmit $c_i(t+1)$ to and receive $c_j(t+1)$ from $j \in \mathcal{N}_i^E$;
7: **end for**
8: Return $c_i(t+1)$.

4.2 The Projected Jacobi Approach with Acceleration (PJA)

As we have discussed in the above section, the projected Jacobi approach is essentially a projected gradient descent method, and hence can be accelerated. Here we consider the Nesterov acceleration technique [29], [30], which greatly reduces the iteration complexity without incurring extra communication cost. Instead of directly using gradient descent in 16, we apply Nesterov acceleration technique to update the recursion:

$$y_i(t+1) = c_i(t) - \gamma v_i(t+1), \tag{17a}$$

$$c_i(t+1) = [y_i(t+1) + \delta(t)(y_i(t+1) - y_i(t))]^+, \tag{17b}$$

where the scalar $\delta(t)$ is a time-varying weight parameter. We propose to update $\delta(t)$ as

$$\theta(t+1) = \theta(t)\frac{\sqrt{\theta(t)^2 + 4} - \theta(t)}{2}, \tag{18a}$$

$$\delta(t+1) = (1 - \theta(t+1))\frac{\sqrt{\theta(t+1)^2 + 4} - \theta(t+1)}{2}, \tag{18b}$$

and $\theta(0)$ is initialized as 1.

Algorithm 4. The Projected Jacobi Approach with Acceleration at sensor v_i

Require: Influence set $\mathcal{N}_i^E$ and index set $\mathcal{J}_i$, local data $\mathbf{h}_i$ and b_i.

Transmit b_i to and receive b_j from $v_j \in \mathcal{N}_i^E$. Calculate $r_i = 1 - \lambda \sum_{j \in \mathcal{N}_i^E} h_{ij} b_j$ and $m_i = \mathbf{h}_i^T \mathbf{h}_i$;
Initialize θ as 1, y_i and c_i as 0;
for $t = 0, 1, 2, \ldots,$ sensor v_i **do**
  if $\mathrm{mod}\,(t+1, T) = 0$ **then**
    Set $\theta(t+1) = 1$;
  else
    Update $\theta(t+1)$ according to (18a);
  end if
  Compute $u_i(t+1)$ according to (16a). Transmit $u_i(t+1)$ to and receive $u_j(t+1)$ from $j \in \mathcal{N}_i^E$;
  Compute $v_i(t+1)$, $y_i(t+1)$, and $\delta(t+1)$ according to (16c), (17a), and (18b) respectively;
  Compute $c_i(t+1)$ according to (16d). Transmit $c_i(t+1)$ to and receive $c_j(t+1)$ from $j \in \mathcal{N}_i^E$;
end for
Return $c_i(t+1)$.

The Nesterov acceleration technique is a momentum method in which the current iteration depends on the pervious iterations, and the momentum grows from one iteration to the next [31]. When the momentum accumulates too much, the current iteration will deviate, and hence ripples and bumps will be observed if one traces the objective value. Therefore we can restart the acceleration process in order to alleviate the accumulation of momentum. For simplicity, here we use fixed restart which reset θ to its initial value 1 after every T iterations. The projected Jacobi approach with acceleration is outlined in Algorithm 4. Compared to the one without acceleration, the communication cost remains the same.

5 Simulation Experiments

In this section, we provide simulation experiments to demonstrate the effectiveness of the proposed decentralized algorithms and the effect of the parameters λ and r_E. Specifically, we show convergence of the algorithms to the optimal solution of (1) as well as how the convergence rate varies with respect to the regularization parameter λ. We also show the effect of the influence range r_E on the convergence rate and the estimation accuracy.

Throughout the simulation experiments, $L = 200$ sensors are uniformly randomly deployed in a 10×10 square sensing field. There are five events occurring at random sensor points and their magnitudes are uniformly randomly chosen from $[0, 1]$. We assume that the measurement coefficients $h_{ij} = \exp(-d_{ij}^2/\sigma^2)$ where σ^2 is a known parameter. Since this exponentially decaying function of d_{ij} is always positive, we define a nominal influence range r_{E0} such as $\exp(-r_{E0}^2/\sigma^2) = 0.01$. Therefore, an event has negligible influence on a sensor beyond the nominal influence range r_{E0}.

Note that this setting comes from the application of structural health monitoring. A WSN is applied to detecting damages of a steel-frame structure and the sensors are deployed at the joints of the frame. Each sensor has a baseline model about its response to ambient vibrations given that the structure is well-conditioned. If damages occur at joints, sensors close to these positions observe abnormal responses that correspond to abnormal statistical models. Through comparing the identified statistical models and the baseline models the WSN can estimate the positions and severities of the damages, which boils down to an ℓ_1 regularized nonnegative least squares problem in the form of (1). The baseline models as well as how a damage influences the identified statistical models can be simulated and pre-acquired by finite-element programs such as OpenSees [22].

We compare performance of the four decentralized algorithms:

1) Full consensus algorithm based on ADMM (FC);
2) Partial consensus algorithm based on ADMM (PC);
3) Projected Jacobi approach;
4) Projected Jacobi approach with acceleration that is restarted after every $T = 20$ iterations.

Two performance metrics are used for comparison. The first one is relative error, which is defined as the normalized distance between the current solution to the optimal solution of (1); the second one is convergence time, which is defined as the number of iterations when the distance between the current solution and the optimal solution of (1) reaches the threshold 0.001.

5.1 Convergence of the Proposed Algorithms

First we compare convergence of the four algorithms in Fig. 2. The sensory measurements are polluted by zero-mean Gaussian noise with standard deviation 0.1. The parameter $\sigma^2 = 1$, which corresponds to a nominal influence range $r_{E0} = 2.14$. The influence range r_E and the communication range r_C are both equal to the nominal influence range r_{E0}. For fair comparison, the parameters in the four algorithms are tuned to the best. As depicted in Fig. 2, the partial consensus algorithm converges much faster than the full consensus algorithm, while the convergence rates of the partial consensus algorithm and the projected Jacobi approach are similar. The Nesterov acceleration technique further improves the projected Jacobi approach at the cost of little extra computation burden.

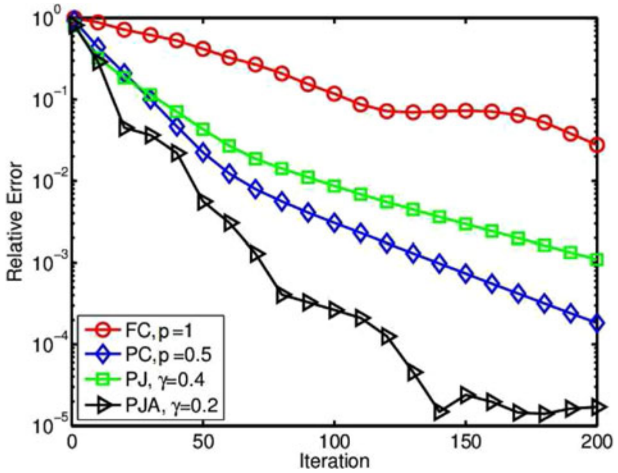


Fig. 2. Convergence of the algorithms.

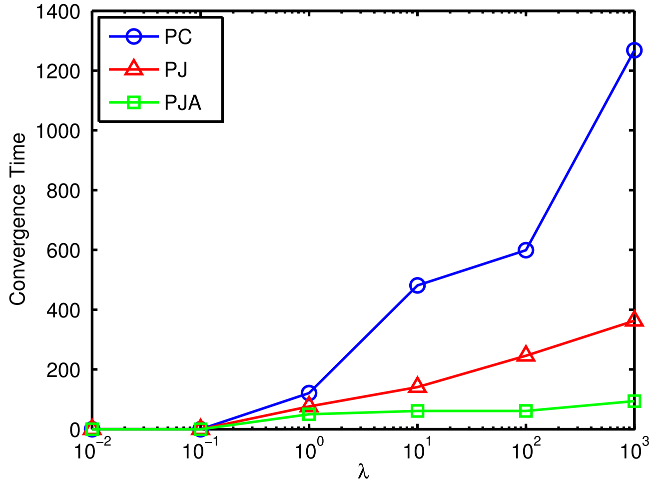


Fig. 4. Convergence time of the algorithms versus varying λ.

This observation indicates that reaching full consensus on the entire optimization variable not only results in high communication cost per iteration, but also incurs slow convergence. Reaching partial consensus helps reduce convergence time, while imposing no consensus constraint is the most advantageous. Hence properly modelling the problem is critical to designing communication-efficient decentralized algorithms.

The parameter σ^2, which shows how the influence of an event decays with distance, affects both the convergence time and the communication cost per iteration of the decentralized algorithms. Fig. 3 varies σ^2 such that the nominal influence range r_{E0} also varies. The influence range r_E and the communication range r_C are both equal to the nominal influence range r_{E0}. The sensory measurements are polluted by zero-mean Gaussian noise with standard deviation 0.1. When σ^2 becomes smaller, the nominal influence range r_{E0} also becomes smaller and both the partial consensus algorithm and the Jacobi approach converge faster. Furthermore, the communication cost per iteration is lower because the influence range and the communication range are also smaller. Recall that a small σ^2 means that the influence of the events is local, which appears in engineering applications such as structural health monitoring [22]. Particularly, we observe that if σ^2 is small enough such as the measurement matrix $\mathbf{H}$ is diagonal dominant, the projected Jacobi approach and the projected Jacobi approach with acceleration converges to the optimal solution within a dozen of iterations.

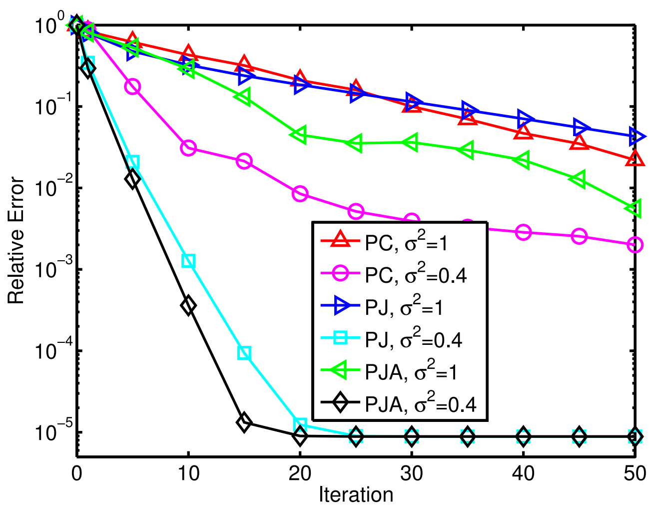


Fig. 3. Convergence of the algorithms with different σ^2.

5.2 The Effect of λ

The regularization parameter λ affects the optimal solution of the event detection model (1); this issue has been extensively discussed in the compressive sensing literature, e.g., [32]. Here we numerically check the effect of λ on the convergence rates of the proposed algorithms. Fig. 4 shows that the convergence rates of the partial consensus algorithm, the projected Jacobi approach, and the projected Jacobi approach with acceleration all become slower as λ increases. Considering both estimation accuracy and convergence rate, λ should be chosen as a medium value. For this concrete example $\lambda \in [20, 100]$ is proper for the partial consensus algorithm and $\lambda \in [20, 1000]$ is proper for the Jacobi approach.

5.3 The Effect of r_E

The influence range r_E is important to both the estimation accuracy and the communication cost of the decentralized algorithms. If r_E is smaller than the nominal influence range r_{E0}, the solutions of the decentralized algorithms are biased since the model is no longer accurate. Denote the optimal solution of (1) as $\mathbf{c}^*$. Given an influence range r_E and setting the measurement coefficients h_{ij} to be 0 if $d_{ij} \geq r_E$, the optimal solution of (1) becomes $\mathbf{c}_E$. Fig. 5 demonstrates how the normalized distance between $\mathbf{c}_E$ and $\mathbf{c}^*$ varies with the choice of r_E. Here the sensory measurements are polluted by zero-mean Gaussian noise with standard deviation 0.1; the parameter $\sigma^2 = 1$ and hence the nominal influence range $r_{E0} = 2.14$, and the communication range r_C is equal to the influence range r_E that varies. When r_E is close to r_{E0}, the model mismatch is neglectable. When $r_{E0}/r_E > 1.5$, the estimation accuracy significantly decreases.

The influence range r_E also affects the communication cost of the decentralized algorithms with respect to both convergence time and communication cost per iteration. In Figs. 6, 7, and 8, we show convergence of the the partial

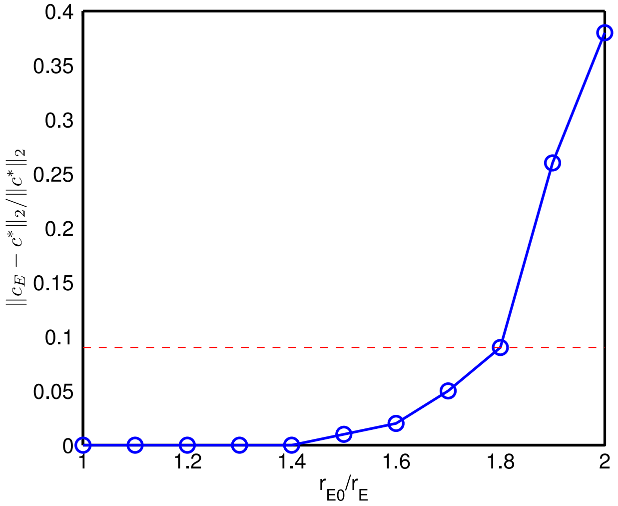


Fig. 5. $\|\mathbf{c}_E - \mathbf{c}^*\|_2/\|\mathbf{c}^*\|_2$ versus varying r_E.

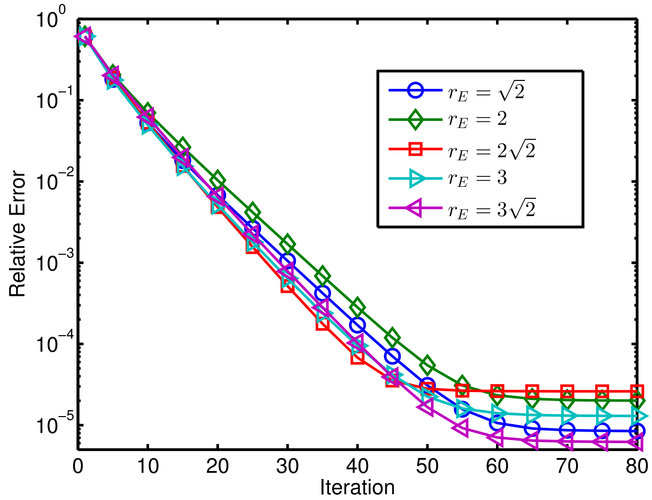


Fig. 7. Convergence of the projected Jacobi approach for different r_E.

consensus algorithm, the projected Jacobi approach, and the projected Jacobi approach with acceleration for different r_E. The sensory measurements are polluted by zero-mean Gaussian noise with standard deviation 0.1 and the parameter $\sigma^2 = 0.4$. Convergence rate of the partial consensus algorithm is highly dependent on the choice of r_E since r_E determines the number of consensus constraints. The projected Jacobi approach and the projected Jacobi approach with acceleration are insensitive to the choice of r_E because they do not impose any consensus constraints. However, since we choose $r_C = r_E$, their communication cost per iteration also varies with r_E.

6 Conclusion

This paper considers monitoring multiple events in a sensing field using a large-scale WSN. Exploiting the sparse nature of the events, the problem is formulated as ℓ_1 regularized nonnegative least squares where the optimization variable is a sparse event vector representing the locations and magnitudes of events. Several communication-efficient algorithms have been developed that are scalable to large networks. Motivated by the observation that an event occurring in the sensing field usually has limited influence range, we suggest to avoid the traditional full consensus technique that requires each sensor to recover the entire event vector and hence leads to high communication cost. Alternatively, we develop two decentralized algorithms, one is the partial consensus algorithm and another is the Jacobi approach. In the partial consensus algorithm based on the ADMM, each sensor is responsible of recovering those events relevant to itself. This strategy greatly reduces the amount of information exchanged among the sensors. The Jacobi approach addresses the case that each sensor only cares about the event occurring at its own position. The communication cost per iteration is hence minimal and the convergence rate is much faster than those based on the ADMM. Simulation results validate the effectiveness of the proposed algorithms and demonstrate the importance of proper modelling in designing communication-efficient decentralized algorithms for the event monitoring application.

Acknowledgments

The work of K. Yuan and Q. Ling was supported by NSFC grant 61004137.

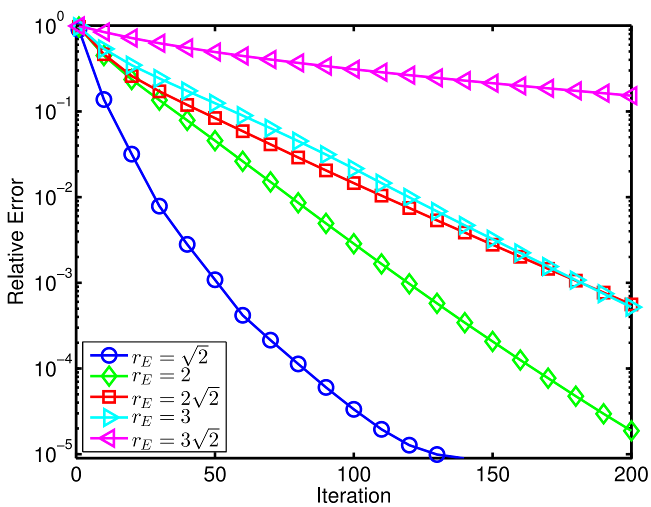


Fig. 6. Convergence of the partial consensus algorithm for different r_E.

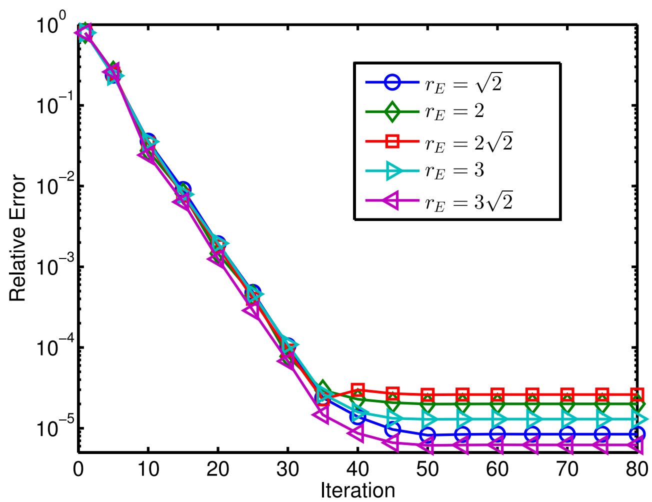


Fig. 8. Convergence of the projected Jacobi approach with acceleration for different r_E.

REFERENCES

[1] V. Cevher, M. Duarte, and R. Baraniuk, "Distributed target localization via spartial sparsity," in *Proc. Eur. Signal Process. Conf.*, 2008, pp. 25–29.

[2] J. Lynch, "An overview of wireless structural health monitoring for civil strucutres," *Philosophical Trans. Royal Soc. A*, vol. 365, pp. 345–372, 2007.

[3] A. Schmidt and J. Moura, "Distributed field reconstruction with model-robust basis pursuit," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2012, pp. 2673–2676.

[4] J. Meng, W. Yin, H. Li, E. Hossain, and Z. Han, "Collaborative spectrum sensing from sparse observations in cognitive radio networks," *IEEE J. Sel. Topics Commun.*, vol. 29, no. 2, pp. 327–337, Feb. 2011.

[5] A. Sundaresan, P. Varshney, and N. Rao, "Distributed detection of a nuclear radioactive source using fusion of correlated decisions," in *Proc. Int. Conf. Inf. Fusion*, 2007, pp. 1–7.

[6] R. Huang, W. Song, M. Xu, N. Peterson, B. Shirazi, and R. LaHusen, "Real-world sensor network for long-term volcano monitoring: design and findings," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 2, pp. 321–329, Feb. 2012.

[7] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J. Stankovic, T. Abdelzaher, and B. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments," in *Proc. ACM 3rd Int. Conf. Embedded Netw. Sens. Syst.*, 2005, pp. 205–217.

[8] G. Xing, J. Wang, Z. Yuan, R. Tan, L. Sun, Q. Huang, X. Jia, and H. So, "Mobile scheduling for spatiotemporal detection in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 12, pp. 1851–1866, Dec. 2010.

[9] S. Pereira, R. Lopez-Valcarce, and A. Pages-Zamora, "A diffusion-based EM algorithm for distributed estimation in unreliable sensor networks," *IEEE Signal Process. Lett.*, vol. 20, no. 6, pp. 595–598, Jun. 2013.

[10] J. Meng, H. Li, and Z. Han, "Sparse event detection in wireless sensor networks using compressive sensing," in *Proc. 43rd Annu. Conf. Inf. Sci. Syst.*, 2009, pp. 181–185.

[11] W. Najy, H. Zeineldin, A. Alaboudy, and W. Woon, "A Bayesian passive islanding detection method for inverter-based distributed generation using ESPRIT," *IEEE Trans. Power Delivery*, vol. 26, no. 4, pp. 2687–2696, Oct. 2011.

[12] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sen. Netw.*, 2004, pp. 20–27.

[13] M. Cetin, L. Chen, J. Fisher III, A. Ihler, R. Moss, M. Wainwright, and A. Willsky, "Distributed fusion in sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 42–55, Jul. 2006.

[14] J. Predd, S. Kulkarni, and H. Poor, "A collaborative training algorithm for distributed learning," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1856–1871, Apr. 2009.

[15] Q. Ling, and Z. Tian, "Decentralized sparse signal recovery for compressive sleeping wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3816–3827, Jul. 2010.

[16] J. Bazerque, and G. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Trans. Signal Proces.*, vol. 58, no. 3, pp. 1847–1862, Mar. 2010.

[17] G. Mateos, J. Bazerque, and G. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5262–5276, Oct. 2010.

[18] Q. Ling, Z. Wen, and W. Yin, "Decentralized jointly sparse optimization by reweighted ℓ_q minimization," *IEEE Trans. Signal Proces.*, vol. 61, no. 5, pp. 1165–1170, Mar. 2013.

[19] D. Bertsekas, and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, 2nd ed. Belmont, MA, USA: Athena Scientific, 1997.

[20] A. Nedic, and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[21] J. Duchi, A. Agarwal, and M. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 592–606, Mar. 2012.

[22] Q. Ling, Z. Tian, Y. Yin, and Y. Li, "Localized structural health monitoring using energy-efficient wireless sensor networks," *IEEE Sen. J.*, vol. 9, no. 11, pp. 1596–1604, Nov. 2009.

[23] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, pp. 33–61, 1998.

[24] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Royal Statist. Soc. B*, vol. 58, pp. 267–288, 1996.

[25] G. Koc, and K. Yegin, "Footstep and vehicle detection using seismic sensors in wireless sensor network: Field tests," *Int. J. Distrib. Sen. Netw.*, vol. 2013, pp. 1–8, 2013.

[26] D. Lindgren, O. Wilsson, F. Gustafsson, and H. Habberstad, "Shooter localization in wireless sensor networks," in *Proc. Int. Conf. Inf. Fusion*, 2009, pp. 404–411.

[27] A. Oka and L. Lampe, "Distributed target tracking using signal strength measurements by a wireless sensor network," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 7, pp. 1006–1015, Sep. 2010.

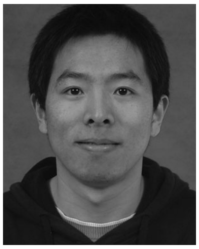
[28] J. Mota, J. Xavier, P. Aguiar, and M. Puschel, "Distributed ADMM for model predictive control and congestion control," in *Proc. 51st IEEE Annu. Conf. Decision Control*, 2012, pp. 5110–5115.

[29] Y. Nesterov, "Gradient methods for minimizing composite functions," *Math. Programming*, vol. 140, no. 1, pp. 125–161, 2013.

[30] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Norwell, MA, USA: Kluwer, 2004.

[31] B. O'Donoghue, and E. Candes, "Adaptive restart for accelerated gradient schemes," *Found. Comput. Math.*, pp. 1–18, 2012.

[32] M. Kolar, L. Song, A. Ahmed, and E. Xing, "Estimating time-varying networks," *The Ann. Appl. Statist.*, vol. 4, pp. 94–123, 2010.



Kun Yuan received the BE degree in telecommunication engineering from Xidian University in 2011. He is currently working toward the MS degree in control theory and control engineering at the Department of Automation, University of Science and Technology of China. His current research focuses on decentralized optimization of networked multi-agent systems.

Qing Ling received the BE degree in automation and the PhD degree in control theory and control engineering from the University of Science and Technology of China in 2001 and 2006, respectively. From 2006 to 2009, he was a postdoctoral research fellow with the Department of Electrical and Computer Engineering, Michigan Technological University. Since September 2009, he has been an associate professor with the Department of Automation, University of Science and Technology of China. His current research focuses on decentralized optimization of networked multi-agent systems.

Zhi Tian received the BE degree in automation from the University of Science and Technology of China in 1994, the MS and PhD degrees in electrical engineering from George Mason University in 1998 and 2000, respectively. Since August 2000, she has been with the Department of Electrical and Computer Engineering, Michigan Technological University, where she is currently a professor. Her research interests are in the areas of signal processing for wireless communications, estimation and detection theory. She served as an associate editors for *IEEE Transactions on Wireless Communications* and *IEEE Transactions on Signal Processing*. She received a CAREER award in 2003 from the US National Science Foundation. She is a fellow of the IEEE.