Decentralized Jointly Sparse Optimization by Reweighted ℓ_q Minimization

Qing Ling, Zaiwen Wen, and Wotao Yin

***Abstract*—A set of vectors (or signals) are jointly sparse if all their nonzero entries are found on a small number of rows (or columns). Consider a network of agents $\{i\}$ that collaboratively recover a set of jointly sparse vectors $\{\mathbf{x}^{(i)}\}$ from their linear measurements $\{\mathbf{y}^{(i)}\}$. Assume that every agent i collects its own measurement $\mathbf{y}^{(i)}$ and aims to recover its own vector $\mathbf{x}^{(i)}$ taking advantages of the joint sparsity structure. This paper proposes novel decentralized algorithms to recover these vectors in a way that every agent runs a recovery algorithm and exchanges with its neighbors only the estimated joint support of the vectors. The agents will obtain their solutions through collaboration while keeping their vectors' values and measurements private. As such, the proposed approach finds applications in distributed human action recognition, cooperative spectrum sensing, decentralized event detection, as well as collaborative data mining. We use a non-convex minimization model and propose algorithms that alternate between support consensus and vector update. The latter step is based on reweighted ℓ_q iterations, where q can be 1 or 2. We numerically compare the proposed decentralized algorithms with existing centralized and decentralized algorithms. Simulation results demonstrate that the proposed decentralized approaches have strong recovery performance and converge reasonably fast.**

***Index Terms*—Decentralized algorithm, jointly sparse optimization, non-convex model.**

I. Introduction

In this correspondence, we develop *decentralized algorithms* for recovering a set of *jointly sparse* vectors (or signals) from their distributed measurements. Suppose that there are L agents constituting a bidirectionally connected network, and each agent i in the network wants to recover its own signal $\mathbf{x}^{(i)} \in \mathcal{R}^N$ from its measurement $\mathbf{y}^{(i)} = \mathbf{A}^{(i)}\mathbf{x}^{(i)} + \mathbf{e}^{(i)} \in \mathcal{R}^M$ where $\mathbf{A}^{(i)} \in \mathcal{R}^{M\times N}$ is a given matrix and $\mathbf{e}^{(i)} \in \mathcal{R}^M$ is noise. Our goal is to recover $\{\mathbf{x}^{(i)}\}_{i=1}^L$ under the assumption that they are jointly sparse, namely, every $x^{(i)}$ has a small number of nonzero entries and the nonzero entries of $\{x^{(i)}\}_{i=1}^L$ appear on a small number of common rows of $\mathbf{X} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(L)}]$ [1]. Further, the recovery algorithms should be decentralized; no fusion center is in charge of collecting the distributed data and recovering the jointly sparse signals. Each agent recovers its own signal, and the computation relies on its local data and limited information exchange with its one-hop neighbors (those agents within its communication range).

Manuscript received July 31, 2011; revised February 19, 2012, June 26, 2012, and November 21, 2012; accepted December 04, 2012. Date of publication December 28, 2012; date of current version February 12, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Namrata Vaswani. The work of Q. Ling is supported in part by NSFC 61004137 and Fundamental Research Funds for the Central Universities. The work of Z. Wen is supported in part by NSFC 11101274 and Doctoral Fund of Ministry of Education of China 20110073120069. The work of W. Yin is supported in part by NSF DMS-07-48839 and ECCS-1028782, ONR N00014-08-1-1101, U.S. ARL and ARO W911NF-09-1-0383.

Q. Ling is with the Department of Automation, University of Science and Technology of China, Hefei, Anhui 230026, China (e-mail: qingling@mail.ustc.edu.cn).

Z. Wen is with the Department of Mathematics, MOE-LSC and Institute of Natural Sciences, Shanghai Jiaotong University, Shanghai 200030, China (e-mail: zw2109@sjtu.edu.cn).

W. Yin is with the Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005 USA (e-mail: wotao.yin@rice.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2012.2236830

Most existing jointly sparse optimization algorithms (e.g., [2]–[5]) are centralized. Consider jointly sparse optimization in a multi-agent network. With centralized computation, all the agents must send their data $\{\mathbf{y}^{(i)}\}_{i=1}^L$ to a fusion center, where a jointly sparse optimization algorithm then tries to obtain the solutions. Such a centralized approach brings high communication burden when the volume of data is large. Contrarily, decentralized algorithms eliminate the need of extensive data transmission, hence simplify traffic routing, balance communication load, and provide better network scalability. These attractive features of decentralized algorithms have been validated in the recent research on decentralized signal estimation [6] and decentralized sparse optimization [7].

A notable algorithm for decentralized jointly sparse optimization is [8], which considers the case where all agents measure the same signal with a group sparse structure. The algorithm requires the agents to exchange their estimates of the signal. In contrast, our work considers different agents measuring different signals with a jointly sparse structure, and our agents do not exchange their signals.

A. Applications

Decentralized jointly sparse optimization is suitable for a group of related agents to quickly obtain solutions and reach decisions by themselves, instead of spending time and bandwidth on sending all the data to a distant fusion center for processing. Let us briefly describe some examples:

Distributed human action recognition: A set of L wireless motion sensors are placed on a person to recognize N actions such as sitting, running, walking, and climbing upstairs/downstairs [9]. Associated with each sensor i is an $N\times 1$ vector $\mathbf{x}^{(i)}$ sparsely representing the current action, or a combination of the current actions. In the training stage, the network collects M measurements for each of the N actions; sensor i has the training data $\mathbf{A}^{(i)} \in \mathcal{R}^{M\times N}$. In the testing stage, sensor i collects $\mathbf{y}^{(i)} \in \mathcal{R}^M$, $i = 1, \ldots, L$, from which the action vectors $\{\mathbf{x}^{(i)}\}_{i=1}^L$ are recovered based on the joint sparsity assumption[1].

Cooperative spectrum sensing: Suppose that L cognitive radios are sensing sparse wideband spectra[2], where the ith cognitive radio takes the time-domain measurement $\mathbf{y}^{(i)}$. Each $\mathbf{y}^{(i)}$ can be sparsely represented under the Fourier basis, namely, $\mathbf{y}^{(i)} = \mathbf{F}^{-1}\mathbf{x}^{(i)}$ where $\mathbf{F}$ is the Fourier transform matrix and $\mathbf{x}^{(i)}$ is a sparse vector. Due to channel fading, $\{\mathbf{x}^{(i)}\}_{i=1}^L$ have different values, yet they are jointly sparse. The task of cooperative spectrum sensing is to recover $\{\mathbf{x}^{(i)}\}_{i=1}^L$ from $\{\mathbf{y}^{(i)}\}_{i=1}^L$ via the cooperation of the distributed cognitive radios [10].

In [11], a set of distributed cognitive radios, each equipped with a frequency selective filter, record (dimension-reduced) linear combinations of sparse wideband spectra. The ith cognitive radio records $\mathbf{y}^{(i)} = \mathbf{A}^{(i)}\mathbf{x}^{(i)}$. For the same reason, $\{\mathbf{x}^{(i)}\}_{i=1}^L$ are subject to jointly sparse recovery.

Decentralized event detection: In [12], footstep data were collected by two sets of nine sensors, each set consisting of four acoustic, three seismic, one passive infrared and one ultrasonic sensors. Jointly sparse optimization recovers a set of coefficient vectors, which are then used to classify the subjects and extract their features. Stronger performance than support vector machine (SVM) and kernel SVM is obtained.

Collaborative data mining: Suppose that a set of data centers perform learning tasks, which take advantages of the jointly sparse struc-

[1]Paper [9] assumes that $\{\mathbf{x}^{(i)}\}_{i=1}^L$ are identical and sparse, but we argue that they are jointly sparse. They are not always identical because the same action can vary in speed and size and thus leads to different readings across different sensors.

[2]The number of actives channels is assumed to be small compared to the total number of channels.

ture. If the data contain precious or private information belonging to the individual agents (e.g., medical data, user rating data, or any data whose sharing are prohibited by law), or if the amount of data is too large to send over the network, collecting all the data centrally is infeasible. However, if the data centers are willing to collaborate by exchanging their estimates of the joint support, properly designed decentralized algorithms become a viable option.

B. Models

To achieve measurement privacy, this paper uses decentralized algorithms based on agents exchanging their estimates of the joint support, instead of exchanging their solutions $\mathbf{x}^{(i)}$ or data $\mathbf{y}^{(i)}$. The algorithms are developed from the following non-convex model, which extends [13] from sparse optimization to jointly sparse optimization:

$$\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} f\left(\{\mathbf{x}^{(i)}\}_{i=1}^{L}\right) := \sum_{i=1}^{L} g^{(i)}(\mathbf{x}^{(i)},\mathbf{y}^{(i)}) + \lambda_q L \sum_{k=1}^{N} \log\left(\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}|^q + \epsilon_q\right), \quad (1)$$

where $g^{(i)}$'s are convex loss functions, $q = 1$ or 2, $\epsilon_q > 0$ is a smoothing parameter, and $\lambda_q > 0$ is a penalty parameter. In (1), the sum-log term is non-convex and promotes the joint sparsity of $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$. The sum $\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}|^q$ bundles the kth elements from all the L vectors. Hence, minimizing the entire sum-log term of (1) becomes minimizing the logarithmic sum of the N bundles, which tends to make all the bundles zero except for a few.

In model (1), applying quadratic loss to linear measurements (or linear regression), we obtain:

$$\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} f\left(\{\mathbf{x}^{(i)}\}_{i=1}^{L}\right) := \frac{1}{2}\sum_{i=1}^{L}\|\mathbf{A}^{(i)}\mathbf{x}^{(i)} - \mathbf{y}^{(i)}\|_2^2 + \lambda_q L \sum_{k=1}^{N} \log\left(\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}|^q + \epsilon_q\right). \quad (2)$$

Corresponding to $q = 1$ and $q = 2$, iteratively reweighted ℓ_1 and ℓ_2 algorithms can be used to solve (1) and (2); see Section II. Due to non-convexity, no guarantees are known for finding the global optimum, even for (2) with $L = 1$. However, like several algorithms for sparse signal recovery such as the reweighted ℓ_1 and ℓ_2 algorithms [13], [14], our reweighted ℓ_q algorithms empirically return solutions of equal or better quality than convex models including the one based on $\ell_{2,1}$-norm minimization.

For joint sparsity under a basis or dictionary $\mathbf{D}$, one can use the loss function:

$$\frac{1}{2}\sum_{i=1}^{L}\|\mathbf{A}^{(i)}\mathbf{D}\mathbf{x}^{(i)} - \mathbf{y}^{(i)}\|_2^2. \quad (3)$$

Other examples include the absolute (i.e., ℓ_1), logistic, and hinge loss functions, which are commonly used in statistical machine learning.

For joint sparsity under transform Ψ (namely, $\{\Psi(\mathbf{x}^{(i)})\}_{i=1}^{L}$ is jointly sparse), the term $|x_k^{(i)}|^q$ in the regularization function (the sum-log term of (1)) should be replaced by $|(\Psi(\mathbf{x}^{(i)}))_k|^q$.

Furthermore, instead of one signal for each agent, each agent can be associated with a group of vectors or signals. If the groups for different agents do not overlap, model (1) still applies. If some groups overlap, additional constraints in the form of $\mathbf{x}^{(i)} = \mathbf{x}^{(j)}$, where $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are copies of one common vector in different groups, must be introduced and will lead to computational overhead in decentralized algorithms.

The rest of this paper is organized as follows. We describe in Section II the centralized algorithm for (1). The corresponding decentralized algorithms with exact and inexact consensus are introduced in Section III. Numerical comparison between the centralized and decentralized algorithms are presented in Section IV. Finally, we conclude this paper with a few remarks on future work in Section V. The convergence proof of the centralized algorithm is given in Appendix.

II. Centralized Reweighted ℓ_q Minimization

This section explains how reweighted ℓ_q iterations, $q = 1, 2$, solve models (1) (and (2)) and also describes a centralized approach based on these iterations.

A. Majorization Minimization

In the iterative majorization minimization algorithm [15], given $\{\mathbf{x}^{(i)}(t)\}_{i=1}^{L}$ at iteration (or time) t, the sum-log term of (1) is linearized with respect to $\{\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}|^q\}_{k=1}^{N}$ as:

$$\begin{aligned}\lambda_q L \sum_{k=1}^{N} \log\left(\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}|^q + \epsilon_q\right) &\simeq \lambda_q L \sum_{k=1}^{N} \log\left(\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}(t)|^q + \epsilon_q\right) \\ &\quad + \lambda_q \sum_{k=1}^{N} \frac{\sum_{i=1}^{L}\left(|x_k^{(i)}|^q - |x_k^{(i)}(t)|^q\right)}{\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}(t)|^q + \epsilon_q}, \end{aligned} \quad (4)$$

where the first term is a constant and the second term is a function of the unknowns $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$. The surrogate objective function is thus the first term in (1) plus the linearization term in (4):

$$\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} f_s(\{\mathbf{x}^{(i)}\}_{i=1}^{L}) := \sum_{i=1}^{L} g^{(i)}(\mathbf{x}^{(i)},\mathbf{y}^{(i)}) + \lambda_q \sum_{k=1}^{N} \frac{\sum_{i=1}^{L}|x_k^{(i)}|^q}{\frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}(t)|^q + \epsilon_q}. \quad (5)$$

This surrogate objective function is convex in $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$ for $q \geq 1$ and separable with respect to $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$; before introducing its decentralized algorithms, we first describe a centralized approach based on reweighted ℓ_q minimization iterations.

B. Centralized Reweighted ℓ_q Minimization

Introduce a weight vector $\mathbf{u} = [u_1,\ldots,u_N]^T \in \mathcal{R}^N$. To minimize (5), the centralized approach performs the following two steps at iteration t:

Step 1: Updating the Weight Vector. Calculate $\mathbf{u}$ from $\{\mathbf{x}^{(i)}(t)\}_{i=1}^{L}$:

$$u_k(t+1) \leftarrow \frac{1}{L}\sum_{i=1}^{L}|x_k^{(i)}(t)|^q + \epsilon_q, \quad k = 1,\ldots,N. \quad (6)$$

Step 2: Updating the Estimates. Minimize (5) with respect to $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$:

$$\mathbf{x}^{(i)}(t+1) \leftarrow \arg\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} g^{(i)}(\mathbf{x}^{(i)},\mathbf{y}^{(i)}) + \lambda_q \sum_{k=1}^{N} \frac{|x_k^{(i)}|^q}{u_k(t+1)}, \quad i = 1,\ldots,L. \quad (7)$$

If $L = 1$, steps (6) and (7) reduce to the building block of the reweighted ℓ_1 and ℓ_2 algorithms [13], [14] for $q = 1$ and 2, respectively. One can see that $u_k^{-1}(t+1)$, $k = 1, \ldots, N$, are the weight variables. If $\mathbf{x}^{(i)}(t) \approx \mathbf{x}^{(i)}(t+1)$, $i = 1, 2, \ldots, L$ and ϵ_q is relatively small compared to $\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t)|^q$, then $\sum_{i=1}^{L} \frac{|x_k^{(i)}|^q}{u_k(t+1)} \approx 1$ whenever $\sum_{i=1}^{L} |x_k^{(i)}|^q \neq 0$; hence, the last term in (7), summing up over $i = 1, \ldots, L$, becomes approximately the number of nonzero rows in $\mathbf{X}$.

The following theorem elaborates on the convergence of the above iteration for solving (1).

Theorem 2.1: The sequence $\left\{\{\mathbf{x}^{(i)}(t+1)\}_{i=1}^{L}\right\}_{t=0,1,\ldots}$ generated by (6) and (7) for $q = 1$ or 2 has a convergent subsequence which converges to a stationary point of (1).

A proof of the theorem can be found in Appendix.

III. Decentralized Reweighted ℓ_q Minimization

In the centralized approach, (7) is carried out by each individual agent i, so it is naturally decentralized. It remains to decentralize the computation of $\mathbf{u}(t+1)$ in (6). In this section, we describe an exact and an inexact decentralized approaches, both of which reduce to average consensus subproblems. Among several average consensus algorithms, we use the alternating direction method (ADM) [16].

A. Consensus Optimization

Observe that in (6), $\mathbf{u}(t+1)$ is set as the average of $|x_k^{(i)}(t)|^q$, $i = 1, \ldots, L$, plus ϵ_q, and this is the well-known *average consensus problem* . It has stochastic decentralized solutions, such as the randomized gossip algorithm in [17]. Here we solve this *average consensus problem* in a deterministic decentralized manner, first formulating it as a *consensus optimization problem* and then solving it with the ADM.

Consensus optimization is a powerful tool in designing decentralized networked multi-agent systems. To obtain an optimization variable common to the agents, we let each agent hold a local copy of the optimization variable and impose consensus constraints on the local copies. Such a technique has been successfully applied in decentralized signal recovery [7], [18]. To update $\mathbf{u}$ according to (6), we introduce the consensus optimization problem:

$$\min_{\mathbf{u}^{(1)},\ldots,\mathbf{u}^{(L)}} \quad \frac{1}{2}\sum_{i=1}^{L}\sum_{k=1}^{N}\left(u_k^{(i)} - |x_k^{(i)}(t)|^q - \epsilon_q\right)^2, \quad \text{s.t.} \quad \mathbf{u}^{(i)} = \mathbf{u}^{(j)}, \quad \forall j \in \mathcal{N}_i,\ \forall i, \tag{8}$$

where $\mathbf{u}^{(i)}$ denotes the local copy of $\mathbf{u}$ held by agent i, $u_k^{(i)}$ is its kth element, and $\mathcal{N}_i$ denotes the neighbors of agent i. Two agents are neighbors if they are connected by a bidirectional edge. Solving (8) gives a copy $\mathbf{u}^{(i)}(t+1)$ of $\mathbf{u}(t+1)$ for each agent i, and they all obey (6); we refer the interested reader to the proof in [7].

B. Decentralized Consensus Optimization

Now we apply the ADM [16] to problem (8) in a decentralized manner for each t. Suppose that time t is divided into S slots. Given $\{\mathbf{x}^{(i)}(t)\}_{i=1}^{L}$, a two-step inner loop is used to update $\{\mathbf{u}^{(i)}(t+1)\}_{i=1}^{L}$:

Step 1.1: Optimizing the Weight Vector. At time t slot s, agent i updates $\mathbf{u}^{(i)}(t + \frac{s+1}{S})$ as:

$$u_k^{(i)}\left(t+\tfrac{s+1}{S}\right) \leftarrow \frac{|x_k^{(i)}(t)|^q - \alpha_k^{(i)}(t+\frac{s}{S}) + c|\mathcal{N}_i| u_k^{(i)}(t+\frac{s}{S})}{1+2c|\mathcal{N}_i|} + \frac{c\sum_{j\in\mathcal{N}_i} u_k^{(j)}(t+\frac{s}{S})}{1+2c|\mathcal{N}_i|} + \epsilon_q, \quad k = 1, \ldots, N, \tag{9}$$

where $\alpha_k^{(i)}$ is the kth element of the Lagrange multiplier $\boldsymbol{\alpha}^{(i)} \in \mathcal{R}^N$, $|\mathcal{N}_i|$ denotes the cardinality of the set $\mathcal{N}_i$, and c is either fixed to a positive constant or varying during the optimization process.

Step 1.2: Updating the Lagrange Multipliers. At time t slot s, agent i updates $\boldsymbol{\alpha}^{(i)}(t + \frac{s+1}{S})$ as:

$$\boldsymbol{\alpha}^{(i)}(t+\tfrac{s+1}{S}) \leftarrow \boldsymbol{\alpha}^{(i)}(t+\tfrac{s}{S}) + c\left(|\mathcal{N}_i|\mathbf{u}^{(i)}(t+\tfrac{s+1}{S}) - \sum_{j\in\mathcal{N}_i}\mathbf{u}^{(j)}(t+\tfrac{s+1}{S})\right). \tag{10}$$

As S grows, this loop converges to the centralized solution (6). See [18] for more details.

C. Inexact Decentralized Consensus Optimization

Running many loops of (9) and (10) is not entirely necessary. We found that an inexact update of $\mathbf{u}$ will still allow $\{\mathbf{x}^{(i)}\}_{i=1}^{L}$ to improve quickly. Therefore, we compute (6) inexactly before letting agent i update its $\mathbf{x}^{(i)}$ in (7), and this will translate to significantly reduced agent-agent communication. To compute (6) inexactly, we propose to execute (9) and (10) only once. This leads to the following 3–step decentralized iteration at t, which replaces the 2–step centralized iteration (6)–(7):

Step 1: Updating the Weight Vector. The ith agent collects $\sum_{j\in\mathcal{N}_i} u_k^{(j)}(t)$ for all k (already done in step 2 at last iteration) and sets:

$$u_k^{(i)}(t+1) \leftarrow \frac{|x_k^{(i)}(t)|^q - \alpha_k^{(i)}(t) + c|\mathcal{N}_i| u_k^{(i)}(t) + c\sum_{j\in\mathcal{N}_i} u_k^{(j)}(t)}{1+2c|\mathcal{N}_i|} + \epsilon_q, \quad k = 1, \ldots, N. \tag{11}$$

Step 2: Updating the Lagrange Multipliers. The ith agent collects $\sum_{j\in\mathcal{N}_i}\mathbf{u}^{(j)}(t+1)$ and sets:

$$\boldsymbol{\alpha}^{(i)}(t+1) \leftarrow \boldsymbol{\alpha}^{(i)}(t) + c\left(|\mathcal{N}_i|\mathbf{u}^{(i)}(t+1) - \sum_{j\in\mathcal{N}_i}\mathbf{u}^{(j)}(t+1)\right). \tag{12}$$

Step 3: Updating the Signals. The ith agent computes:

$$\mathbf{x}^{(i)}(t+1) \leftarrow \arg\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} g^{(i)}(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})^2 + \lambda_q \sum_{k=1}^{N} \frac{|x_k^{(i)}|^q}{u_k(t+1)}, \quad i = 1, \ldots, L. \tag{13}$$

Steps 1–3 are iterated until a certain stopping criterion is met.

The decentralized reweighted ℓ_q minimization algorithm works as follows. At the initial time 0, each agent i takes its measurement vector $\mathbf{y}^{(i)}$ with the measurement matrix $\mathbf{A}^{(i)}$, coordinates with its neighbors to decide the value of $|\mathcal{N}_i|$, and sets the initial values of $\mathbf{x}^{(i)}$, $\mathbf{u}^{(i)}$, and $\boldsymbol{\alpha}^{(i)}$ as zero. The settings of λ_q, c, and ϵ_q can be common to all agents. At time t, the agents update ϵ_q as we will discuss below, and broadcast $\{\mathbf{u}^{(i)}(t)\}_{i=1}^{L}$ to their neighbors. After collecting $\{\mathbf{u}^{(j)}(t)\}_{j\in\mathcal{N}_i}$ from its neighbors, each agent i updates the weight vector $\mathbf{u}^{(i)}(t+1)$ with (11), updates the Lagrange multipliers $\boldsymbol{\alpha}^{(i)}(t)$ with (12), and updates the local copies $\mathbf{x}^{(i)}(t+1)$ with (13). The algorithm is fully decentralized, requiring only local information exchange among one-hop neighbor agents.

We have not been able to prove convergence for the proposed decentralized reweighted ℓ_q minimization algorithms though we numerically demonstrate their quick convergence in the next section. We compare all algorithms under the same network topology, and we do not study

TABLE I
COMPARED ALGORITHMS

Name		Algorithm
CRLq	q=1,2	*centralized* reweighted ℓ_q algorithms; Section II
DRLq	q=1,2	*decentralized* reweighted ℓ_q algorithms; Section III
IRLq	q=1,2	*independent* reweighted ℓ_q algorithms at each agent; *no joint sparsity*
CL21		*centralized* $\ell_{2,1}$-norm minimization

how the network topology affects the convergence speed, which is beyond the scope of this paper[3]. Further, it is theoretically interesting to understand why inexact consensus at each t is sufficient and leads to an overall effective algorithm.

D. Parameter Settings

There are three parameters λ_q, c, and ϵ_q in the proposed decentralized algorithms. λ_q balances data fidelity and signal sparsity, and it should be chosen based on the prior knowledge on the data and unknown signals, or determined by cross validation. c can be set as a constant in the algorithms, though our preliminary simulations show that decreasing c gradually may improve the convergence speed.

Regarding the parameter ϵ_q, $q = 1, 2$, [13] suggests that ϵ_1 be set slightly smaller than the expected nonzero magnitudes of the signal vector in their reweighted ℓ_1 algorithm. Though this adaptation strategy is successful in recovering a single sparse signal, it is difficult to be implemented in recovering jointly sparse signals. If each agent updates ϵ_1 based on its own estimate of the signal vector, different agents may have different values of ϵ_1. Due to this reason, we simply set $\epsilon_1(t) = \max\left(\epsilon_1(0)d_1^t, 10^{-4}\right)$, where $d_1 \leq 1$ is a positive constant. For $q = 2$, we adopt the strategy in the reweighted ℓ_2 algorithm [14]: $\epsilon_2(t) = \max\left(\epsilon_2(0)d_2^t, 10^{-8}\right)$, where $d_2 \leq 1$ is a positive constant. According to our preliminary numerical experiments, this setting $\epsilon_1(t)$ and $\epsilon_1(t)$ works well for different problem dimensions.

IV. NUMERICAL EXPERIMENTS

This section presents the numerical results of solving (2) by the centralized and decentralized algorithms listed in Table I. The presentation focuses on their recovery performances, as well as convergence rates. Due to space limit, we present only one set of results[4].

A. Simulation Settings

Our numerical simulations consider $L = 50$ agents uniformly randomly deployed in a 100×100 area. Pair-wise agents within a communication radius of $r_C = 30$ are bidirectionally connected. A total of L signal vectors of dimension $N = 20$ are jointly K-sparse, where $K = 2$. Each signal is measured by $M = 10$ linear projections, which are corrupted by i.i.d. zero-mean Gaussian noise with standard deviation 0.05. As $M < N$, the measurements give rise to under-determined linear systems. The nonzero elements in the signals and those in the measurement matrices are sampled independently from the standard Gaussian distribution.

In reweighted ℓ_q algorithms CRLq, DRLq, and IRLq, parameters c and $\epsilon_q(0)$ are set as 1, and d_q as 0.8, which is a moderate value. CL21 centrally optimizes the convex model:

$$\min_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(L)}} \quad \frac{1}{2}\sum_{i=1}^{L} \|\mathbf{A}^{(i)}\mathbf{x}^{(i)} - \mathbf{y}^{(i)}\|_2^2 + \rho \sum_{k=1}^{N} \sqrt{\sum_{i=1}^{L} (x_k^{(i)})^2}. \quad (14)$$

[3]For the average consensus problem, the impact of network topology on convergence speed is discussed in, e.g., [17].

[4]The demo is available on http://www.caam.rice.edu/optimization/L1/decentral/

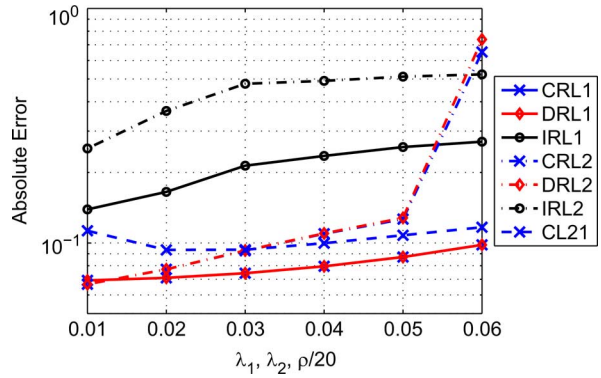

Fig. 1. Comparison of recovery quality. λ_1 and λ_2 are the penalty parameters in (2) for the reweighted ℓ_1 and ℓ_2 algorithms, respectively, and ρ is the penalty parameter in (14) for the $\ell_{2,1}$-norm minimization algorithm.

Performances of the algorithms are measured in terms of the recovered absolute error, which is defined as the ℓ_2-norm difference between the true and obtained signals over the L agents.

B. Recovery Performance and Convergence Rate

Fig. 1 depicts the recovery performances of the algorithms in Table I. Different regularization functions need different parameters, and our purpose is to evaluate different algorithms over a variety of parameter values so we can compare them at their better values.

Firstly, whenever on the same parameter, CRL1 (centralized) and DRL1 (decentralized) give almost the same absolute error. So do CRL2 (centralized) and DRL2 (decentralized). These strongly suggest that decentralizing the reweighted ℓ_1 and ℓ_2 algorithms (through carrying out the average consensus steps incompletely) does not affect their solution quality.

Secondly, comparing CRL1 (non-convex) and CL21 (convex), we find that the former consistently gives better solutions than the latter over a set of different parameters. On the other hand, the performance of CRL2 (non-convex) is more sensitive to the choice of its penalty parameter λ_2. With a small λ_2, CRL1 and CRL2, both decentralized and non-convex, have similar recovery performance and are better than CL21 (convex and centralized), yet that of CRL2 starts to drop as λ_2 is increased. Therefore, we suggest using conservative values of λ_2 when applying CRL2.

Finally, no matter convex or not, the joint sparse algorithms, CL21, CRL1/DRL1, and CRL2/DRL2, have much better performance than algorithms IRL1 and IRL2, which neglect joint sparsity.

The convergence speed decides how many iterations a decentralized algorithm requires to take in order to attain a satisfactory solution and thus directly affects the amount of communication needed. As shown in Fig. 2, the decentralized algorithms, DRL1 with $\lambda_1 = 0.02$ and DRL2 with $\lambda_2 = 0.02$, have reasonably fast convergence. We tested other problem sizes and found that they converged in 100–200 iterations.

V. CONCLUSIONS

(Jointly) sparse optimization has been an active research area during the last several years, and it is used widely in compressive sensing, signal/image processing, and machine learning. There is a practical class of problems where signal measurements or data are located on distributed agents of a network. To recover jointly sparse signals from these measurements or to learn jointly sparse features from the data,

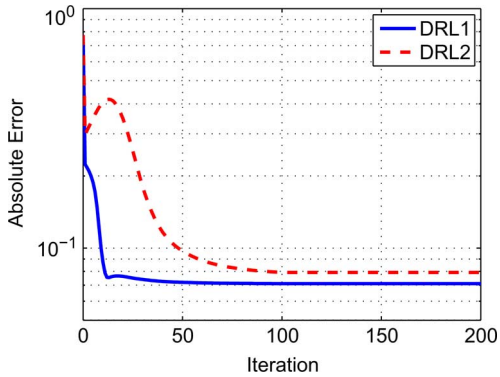

Fig. 2. Convergence rates of the algorithms for a single instance. Here $\lambda_1 = 0.02$ for DRL1 and $\lambda_2 = 0.02$ for DRL2.

the majority of the existing approaches rely on centralized computation at a fusion center, which must have access to all the measurements or data. There are very few approaches to decentralize the computation to the distributed agents. This work addresses the recovery of jointly sparse vectors by decentralized computation in which the agents do *not* exchange the data or solutions but just exchange their guesses of a common signal supports. The update of support guesstimates is based on consensus averaging, and this update is alternated with each agent update of its own vector. The proposed decentralized algorithms compare favorably with various existing centralized and decentralized algorithms.

APPENDIX

Now we prove Theorem 2.1. Our analysis uses the techniques from [19]; we made changes since [19] only considers single sparse signal recovery.

Proof: Since $\mathbf{x}^{(i)}(t+1)$ is the solution of (7), there exist subgradients $c_k^{(i)} \in \partial|x_k^{(i)}(t+1)|^q,\ k = 1, \ldots, N$ such that:

$$\lambda_q \left(\frac{c_k^{(i)}}{\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t)|^q + \epsilon_q} \right)_{1 \le k \le N} + (\mathbf{A}^{(i)})^T (\mathbf{A}^{(i)} \mathbf{x}^{(i)}(t+1) - \mathbf{y}^{(i)}) = 0, \quad i = 1, \ldots, L. \quad (15)$$

Hence, we obtain:

$$\begin{aligned} & f(\{\mathbf{x}^{(i)}(t)\}_{i=1}^L) - f(\{\mathbf{x}^{(i)}(t+1)\}_{i=1}^L) \\ & \quad - \frac{1}{2}\sum_{i=1}^{L} \|\mathbf{A}^{(i)}\mathbf{x}^{(i)}(t+1) - \mathbf{A}^{(i)}\mathbf{x}^{(i)}(t)\|_2^2 \\ & \ge \lambda_q L \sum_{k=1}^{N} \log\left(\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t)|^q + \epsilon_q \right) \\ & \quad - \log\left(\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t+1)|^q + \epsilon_q \right) \\ & \quad + \lambda_q L \sum_{k=1}^{N} \frac{\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t+1)|^q - \frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t)|^q}{\frac{1}{L}\sum_{i=1}^{L} |x_k^{(i)}(t)|^q + \epsilon_q} \\ & = \delta(t+1) \ge 0, \end{aligned} \quad (16)$$

where we have used (15), and the inequality $\log(p) + \frac{1}{p} - 1 \ge 0$ for any positive number p.

Hence, the sequence $\{f(\{\mathbf{x}^{(i)}(t)\}_{i=1}^L)\}$ is monotonically decreasing and converges. It is clear that the sequence $\{\{\mathbf{x}^{(i)}(t+1)\}_{i=1}^L\}$ is contained in a level set

$$\mathcal{L} := \left\{ \{\mathbf{x}^{(i)}\}_{i=1}^L \mid f(\{\mathbf{x}^{(i)}\}_{i=1}^L) \le f(\{\mathbf{x}^{(i)}(0)\}_{i=1}^L) \right\}.$$

Obviously, $\mathcal{L}$ is bounded. Hence, there exists a convergent subsequence $\{\{\mathbf{x}^{(i)}(t_c)\}_{i=1}^L\}$ whose limit is $\{\tilde{\mathbf{x}}^{(i)}\}_{i=1}^L$. It follows from (16) that $\lim_{t\to\infty} \delta(t+1) = 0$, which yields $\lim_{t\to\infty} |\mathbf{x}^{(i)}(t+1)|^q - \lim_{t\to\infty} |\mathbf{x}^{(i)}(t)|^q = 0,\ i = 1, \ldots, L$. This together with (16) implies that: $\lim_{t\to\infty} c_k^{(i)}(x_k^{(i)}(t+1) - x_k^{(i)}(t)) = 0,\ i = 1, \ldots, L;\ k = 1, \ldots, N$.

Using $c_k^{(i)} \in \partial|x_k^{(i)}(t+1)|^q,\ i = 1, \ldots, L;\ k = 1, \ldots, N$ and $\lim_{t\to\infty} |\mathbf{x}^{(i)}(t+1)|^q - \lim_{t\to\infty} |\mathbf{x}^{(i)}(t)|^q = 0$, we obtain: $\lim_{t\to\infty} \mathbf{x}^{(i)}(t+1) - \lim_{t\to\infty} \mathbf{x}^{(i)}(t) = 0,\ i = 1, \ldots, L$. Hence, the subsequence $\{\{\mathbf{x}^{(i)}(t_c+1)\}_{i=1}^L\}$ also converges to $\{\tilde{\mathbf{x}}^{(i)}\}_{i=1}^L$. Combining these fact together and using the upper semi-continuous property of the subgradient, we obtain:

$$\lambda_q \left(\frac{\tilde{c}_k^{(i)}}{\frac{1}{L}\sum_{i=1}^{L} |\tilde{x}_k^{(i)}|^q + \epsilon_q} \right)_{1 \le k \le N} + (\mathbf{A}^{(i)})^T (\mathbf{A}^{(i)} \tilde{\mathbf{x}}^{(i)} - \mathbf{y}^{(i)}) = 0, \quad i = 1, \ldots, L, \quad (17)$$

where $\tilde{c}_k^{(i)} \in \partial|\tilde{x}_k^{(i)}|^q,\ i = 1, \ldots, L;\ k = 1, \ldots, N$. Hence, $\{\tilde{\mathbf{x}}^{(i)}\}_{i=1}^L$ is a stationary point [20] of (2). ■

REFERENCES

[1] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Roy. Statist. Soc. B*, vol. 68, pp. 49–67, 2007.

[2] Y. Eldar, P. Kuppinger, and H. Böcskei, "Block-sparse signals: Uncertainty relations and efficient recovery," *IEEE Trans. Signal Process.*, vol. 58, no. 6, pp. 3042–3054, 2010.

[3] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 317–329, 2010.

[4] A. Rakotomamonjy, "Surveying and comparing simultaneous sparse approximation (or group-Lasso) algorithms," *Signal Process.*, vol. 91, pp. 1505–1526, 2011.

[5] W. Deng, W. Yin, and Y. Zhang, "Group sparse optimization by alternating direction method," [Online]. Available: http://www.caam.rice.edu/zhang/reports/tr1106.pdf

[6] I. Schizas, A. Ribeiro, and G. Giannakis, "Consensus in ad hoc WSNs with noisy links—Part I: Distributed estimation of deterministic signals," *IEEE Trans. Signal Process.*, vol. 56, no. 1, pp. 350–364, 2008.

[7] 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, 2010.

[8] J. Bazerque, G. Mateos, and G. Giannakis, "Group-Lasso on splines for spectrum cartograph," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4648–4663, 2011.

[9] A. Yang, M. Gastpar, R. Bajcsy, and S. Sastry, "Distributed sensor perception via sparse representation," *Proc. IEEE*, vol. 98, pp. 1077–1088, 2010.

[10] F. Zeng, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multi-hop wideband cognitive networks," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 1, pp. 37–48, 2011.

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

[12] N. Nguyen, N. Nasrabadi, and T. Tran, "Robust multi-sensor classification via joint sparse representation," 2011 [Online]. Available: http://thanglong.ece.jhu.edu/Tran/Pub/multisensor_classification_J.pdf

[13] E. Candès, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, pp. 877–905, 2008.

[14] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Las Vegas, NV, USA, Mar. 30–Apr. 4, 2008, pp. 3879–3882.

[15] P. Stoica and Y. Selén, "Cyclic minimizers, majorization techniques, and the expectation-maximization algorithm: A refresher," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 112–114, 2004.
[16] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, 2nd ed. Singapore: Athena Scientific, 1997.
[17] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.
[18] G. Mateos, J. Bazerque, and G. Giannakis, "Distributed sparse linear regression," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5262–5276, 2010.
[19] X. Chen and W. Zhou, "Convergence of reweighted ℓ_1 minimization algorithms and unique solution of truncated ℓ_p minimization," 2010 [Online]. Available: http://www.polyu.edu.hk/ama/staff/xjchen/IRL1-4-8.pdf
[20] J. Nocedal and S. Wright, *Numerical Optimization*. New York, NY, USA: Springer, 2006.