Optimization Methods and Software
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/goms20

A new class of distributed optimization algorithms: application to regression of distributed data

S. Sundhar Ram [a], A. Nedić [b] & V. V. Veeravalli [c]

[a] Electrical and Computer Engineering Department, University of Illinois, Urbana, IL, 61801, USA

[b] Industrial and Enterprise Systems Engineering Department, University of Illinois, Urbana, IL, 61801, USA

[c] Electrical and Computer Engineering Department, University of Illinois, Urbana, IL, 61801, USA

Available online: 24 Jun 2011

To cite this article: S. Sundhar Ram, A. Nedić & V. V. Veeravalli (2012): A new class of distributed optimization algorithms: application to regression of distributed data, Optimization Methods and Software, 27:1, 71-88

To link to this article: http://dx.doi.org/10.1080/10556788.2010.511669

Optimization Methods & Software
Vol. 27, No. 1, February 2012, 71–88

Taylor & Francis
Taylor & Francis Group

A new class of distributed optimization algorithms: application to regression of distributed data

S. Sundhar Ram[a], A. Nedić[b]* and V.V. Veeravalli[c]

[a]*Electrical and Computer Engineering Department, University of Illinois, Urbana, IL 61801, USA;* [b]*Industrial and Enterprise Systems Engineering Department, University of Illinois, Urbana, IL 61801, USA;* [c]*Electrical and Computer Engineering Department, University of Illinois, Urbana, IL 61801, USA*

(*Received 29 December 2009; final version received 20 July 2010*)

In a distributed optimization problem, the complete problem information is not available at a single location but is rather distributed among different agents in a multi-agent system. In the problems studied in the literature, each agent has an objective function and the network goal is to minimize the sum of the agents' objective functions over a constraint set that is globally known. In this paper, we study a generalization of the above distributed optimization problem. In particular, the network objective is to minimize a *function* of the sum of the individual objective functions over the constraint set. The 'outer' function and the constraint set are known to all the agents. We discuss an algorithm and prove its convergence, and then discuss extensions to more general and complex distributed optimization problems. We provide a motivation for our algorithms through the example of distributed regression of distributed data.

Keywords: distributed optimization; convex optimization; distributed regression

AMS Subject Classification: 90C25; 90C30

1. Introduction

This paper deals with a distributed optimization problem, where the complete objective function is not available at a single location, but is rather distributed among different agents who are connected through a network. The focus is on solving the distributed optimization problem in large time-synchronous multi-agent systems. In multi-agent systems, each agent only knows the identity of its immediate neighbours and has no information about the global network topology. The large size of the network and the lack of global network topology information makes it infeasible to collect the problem data from the agents at a single location and then use standard centralized optimization techniques. Instead, algorithms that are *distributed* and *local* are appropriate. In a distributed algorithm, different parts of the algorithm are executed by different agents, possibly simultaneously. The algorithm is additionally local when each agent uses only information locally available to it and other information it can obtain from its immediate neighbours.

*Corresponding author. Email: angelia@illinois.edu

ISSN 1055-6788 print/ISSN 1029-4937 online

http://dx.doi.org/10.1080/10556788.2010.511669
http://www.tandfonline.com

Prior work has addressed different versions of the distributed sum optimization (DSO) problem. In this problem, each agent has a unique objective function and the network goal is to minimize the sum of the individual objective functions over a constraint set. The constraint set is known to all the agents. See [14] for an overview of literature related to the DSO problem. In this paper, we study a generalization of the DSO problem, where the network objective is to minimize a nonlinear function of the sum of the individual agent's objective functions over the constraint set. The 'outer' nonlinear function and the constraint set are known to all the agents. To solve this problem, we propose a distributed, local and iterative algorithm. Each agent maintains an estimate of the optimal point and a summary statistic that is updated in each iteration. The agent receives the estimate and the summary statistic from its immediate neighbours, and then evaluates a weighted average. The weighted average is then updated using locally available information, i.e. the agent's own objective function, the outer function and the constraint set. The network connectivity assumptions are as in [14]. We discuss the proof of convergence for the above problem, and then discuss extensions to more general and complex distributed optimization problems.

Our contributions are twofold. First, we contribute to the literature on distributed optimization by introducing a new class of distributed problems, and an algorithm for solving these problems. The novelty of the algorithm is in the use of a 'tracking'-like step in combination with a distributed gradient-based update. Second, we apply the algorithm to address the problem of vertically and horizontally distributed regression in large peer to peer systems.

The rest of the paper is organized as follows. In Section 2, we introduce the optimization problem and discuss the algorithm. In Section 3, we discuss the assumptions that we make, and in Section 4, we discuss the necessary background. The convergence of the algorithm is proved in Section 5. We then discuss some extensions of the problem in Section 6. We address the distributed regression problem in Section 7 and show that this is a special case of the problems solved in this paper. We conclude with some comments in Section 8.

2. Problem and algorithm

We consider a network consisting of m agents that are indexed by $V = \{1, \ldots, m\}$. The network objective is to solve the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad f(x) := g\left(\sum_{i=1}^{m} h_i(x)\right), \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{1}$$

where $g : \Re \to \Re$, $X \subseteq \Re^p$, $h_i : X \to \Re$ for all $i \in V$. The function h_i is known only to agent i. The function g and the set X are globally known, i.e. to every agent. Further, the network size m is also known to all the agents. The optimal value and the optimal set of the problem are denoted as follows:

$$f^* = \min_{x \in X} f(x), \quad X^* = \{x \in X : f(x) = f^*\}. \tag{2}$$

To solve problem (1), we propose the following distributed, local and iterative algorithm. Time is slotted and one iteration of the algorithm is performed in one time slot. At any time k, every agent i has two statistics; $x_{i,k}$ and $s_{i,k}$. The statistic $x_{i,k}$ is agent i's estimate of an optimal point and $s_{i,k}$ is agent i's estimate of the average value $(1/m)\sum_{i=1}^{m} h_i(x_{i,k})$. These statistics are updated

as follows:

$$\begin{bmatrix} \bar{x}_{i,k} \\ \bar{s}_{i,k} \end{bmatrix} = \sum_{j \in N_i(k+1)} a_{i,j}(k+1) \begin{bmatrix} x_{j,k} \\ s_{j,k} \end{bmatrix},$$

$$x_{i,k+1} = P_X[\bar{x}_{i,k} - \alpha_{k+1} g'(m\bar{s}_{i,k}) \nabla h_i(\bar{x}_{i,k})],$$

$$s_{i,k+1} = \bar{s}_{i,k} + h_i(x_{i,k+1}) - h_i(x_{i,k}). \tag{3}$$

Here, $N_i(k+1)$ is the index set of agents that agent i can communicate with at time $k+1$, and this set also includes agent i. Further, $a_{i,j}(k+1)$ are positive weights, α_{k+1} is the stepsize, g' is the derivative of g, ∇h_i is the gradient of h_i, and P_X denotes the Euclidean projection onto the set X. The algorithm is initialized with

$$x_{i,0} \in X, \quad s_{i,0} = h_i(x_{i,0}) \quad \text{for all } i \in V. \tag{4}$$

The algorithm is distributed and local. Agent i receives $x_{j,k}$ and $s_{j,k}$ from its current neighbours and calculates the weighted averages $\bar{x}_{i,k}$ and $\bar{s}_{i,k}$ using the weights $a_{i,j}(k+1)$. The weighted average is then updated using locally available information (functions g and h_i, number of agents m and the set X) to generate $x_{i,k+1}$ and $s_{i,k+1}$. We postpone the discussion of the algorithm to Section 5, as we need some background material that is introduced in the subsequent sections.

3. Assumptions

We here discuss our assumptions on the problem including the function g, the agents' functions h_i, $i \in V$, the set X, as well as the connectivity structure of the underlying communication network for the agents.

ASSUMPTION 3.1 *The following conditions hold.*

(a) *The set X is convex and closed.*
(b) *The set X is bounded, i.e. there exists a scalar $D > 0$ such that* $\sup_{x \in X} \|x\| \leq D$.
(c) *The function f is convex over an open set that contains the set X.*
(d) *The functions g and h_i are differentiable. Further, g' and ∇h_i are Lipschitz continuous with constant L.*

Assumption 3.1(a) and (c) imply that the problem is a convex optimization problem. From Assumptions 3.1(b) and (d) we can conclude that the norms of the gradients, i.e. $|g'|$ and $\|\nabla h_i\|$, are uniformly bounded over the set X. We denote this bound by C, i.e.

$$\left| g' \left(\sum_{i=1}^{m} h_i(x) \right) \right| \leq C, \quad \|\nabla h_i(x)\| \leq C \quad \text{for all } x \in X \text{ and } i \in V. \tag{5}$$

We make the following standard assumption [14] on the network. At each time k, the topology of the network is represented by a directed graph $G_k = (V, E_k)$, with $(i, j) \in E_k$ if and only if agent i and agent j can communicate at time k.

ASSUMPTION 3.2 *There exists a positive integer Q such that the graph $(V, \cup_{\ell=1,\ldots,Q} E_{k+\ell})$ is strongly connected for all k.*

Note that we do not require the network to be connected at each time. Further, the global constant Q need not be known to any of the agents. It is only important that such a constant exists. The weights $a_{i,j}(k)$ are chosen to satisfy the following assumption.

ASSUMPTION 3.3 *For all k, we have*

(a) $a_{i,j}(k) \geq 0$ *for all* $i, j \in V$, *and* $a_{i,j}(k) = 0$ *for all* $i \in V$ *and* $j \notin N_i(k)$;
(b) $\sum_{j=1}^{m} a_{i,j}(k) = 1$ *for all* $i \in V$;
(c) *There exists a scalar* $\eta \in (0, 1)$ *such that* $a_{i,j}(k) \geq \eta$ *whenever* $a_{i,j}(k) > 0$;
(d) $\sum_{i=1}^{m} a_{i,j}(k) = 1$ *for all* $j \in V$.

Assumption 3.3(a) states that the weights $a_{i,j}(k)$ are non-negative and are equal to zero when agent j is not a neighbour of i at a given time. Assumption 3.3(b) requires that the sum of all weights $a_{i,1}(k), \ldots, a_{i,m}(k)$ is 1 for each agent i. Assumptions 3.3(a) and (b) imply that each agent computes a weighted average of all the iterates it receives from its neighbours. Assumption 3.3(c) ensures that each agent gives a sufficient weight to its current iterate and all the iterates it receives. The agents need not be aware of the common bound η.

Assumption 3.3(d) requires that all the weights $a_{1,j}(k), \ldots, a_{m,j}(k)$ sum to 1 for every agent j. This assumption together with Assumption 3.2, ensures that all the agents are equally influential in the long run. To satisfy Assumption 3.3(d), the agents need to coordinate their weights. Some coordination schemes are discussed in [16].

4. Preliminaries

In our analysis, we will use the following results. For a non-empty closed convex set $X \subseteq \Re^p$, the Euclidean projection on X is non-expansive,

$$\|P_X[x] - P_X[y]\| \leq \|x - y\| \quad \text{for all } x, y \in \Re^p. \tag{6}$$

We will also use the following theorem.

THEOREM 4.1 *Let $\{b_k\}$, $\{d_k\}$, and $\{c_k\}$ be non-negative sequences. Suppose that $\sum_k c_k < \infty$ and*

$$b_{k+1} \leq b_k - d_k + c_k \quad \textit{for all } k, \tag{7}$$

then the sequence $\{b_k\}$ converges and $\sum_k d_k < \infty$.

This is a deterministic version of the theorem by Robbins and Seigmund [10, Lemma 11, Chapter 2.2].

4.1 *Distributed averaging*

We briefly review the distributed averaging algorithm. See [7] for a recent survey. In the distributed averaging problem, agent i has the value θ_i. The goal in distributed averaging is for the agents to learn $\hat{\theta} = (1/m) \sum_{i=0}^{m} \theta_i$ in a distributed and local manner. We will refer to $\hat{\theta}$ as the *target* and θ_i as agent i's *start value*.

Distributed averaging is usually achieved iteratively through a sequence of *consensus steps*. In each step, each agent evaluates the new iterate as a weighted average of its current iterate and

the current iterates of its neighbours. The initial value of the iterate at agent i is its start value θ_i. Formally, if $\{\theta_{i,k}\}$ denotes the sequence of estimates for the target generated by agent i then

$$\theta_{i,k+1} = \sum_{j \in N_i(k+1)} a_{i,j}(k+1)\theta_{j,k}, \quad \theta_{i,0} = \theta_i. \tag{8}$$

Under Assumptions 3.2 and 3.3, it can be shown that $\lim_{k\to\infty} \theta_{i,k} = \hat{\theta}$ for all $i \in V$ (see [8, Corollary 1]). Further, it is intuitively clear that the closer the agents' start values are to the target value the fewer number of iterations are required to obtain a good estimate of the target by each agent.

We state a result that captures the effect of deterministic errors in the averaging algorithm. The result guarantees that the agents achieve consensus when the errors diminish. We state the result here and provide its proof in the appendix.

THEOREM 4.2 *Let Assumptions* 3.2 *and* 3.3 *hold. Consider the iterates generated by*

$$\theta_{i,k+1} = \sum_{j=1}^{m} a_{i,j}(k+1)\theta_{j,k} + \epsilon_{i,k+1} \quad \textit{for all } i \in V.$$

Suppose there exists a non-negative non-increasing scalar sequence $\{\alpha_k\}$ *such that*

$$\sum_{k=1}^{\infty} \alpha_k \|\epsilon_{i,k}\| < \infty \quad \textit{for all } i \in V.$$

Then, for all $i, j \in V$,

$$\sum_{k=1}^{\infty} \alpha_k \|\theta_{i,k} - \theta_{j,k}\| < \infty.$$

5. Algorithm convergence

Since $a_{i,j}(k+1) = 0$ for $j \notin N_i(k+1)$, we can rewrite (3) as follows:

$$\begin{aligned} \begin{bmatrix} \bar{x}_{i,k} \\ \bar{s}_{i,k} \end{bmatrix} &= \sum_{j=1}^{m} a_{i,j}(k+1) \begin{bmatrix} x_{j,k} \\ s_{j,k} \end{bmatrix}, \\ x_{i,k+1} &= P_X[\bar{x}_{i,k} - \alpha_{k+1} g'(m\bar{s}_{i,k}) \nabla h_i(\bar{x}_{i,k})], \\ s_{i,k+1} &= \bar{s}_{i,k} + h_i(x_{i,k+1}) - h_i(x_{i,k}). \end{aligned} \tag{9}$$

We next provide some intuition for the algorithm. The standard gradient projection algorithm for solving the problem in (1) has the following form:

$$x_{k+1} = P_X \left[x_k - \alpha_{k+1} g' \left(\sum_{j=1}^{m} h_j(x_k) \right) \sum_{j=1}^{m} \nabla h_j(x_k) \right].$$

To replicate the standard gradient projection algorithm in our distributed setting, the computations of $\sum_{j=1}^{m} \nabla h_j(x_k)$ and $\sum_{j=1}^{m} h_j(x_k)$ have to be distributed and local. When the function g is the identity function then (9) is identical to the distributed subgradient algorithm in [14]. As in [14],

the combined effect of agent i using $\nabla h_i(\bar{x}_{i,k})$ to update the iterate and then evaluating a weighted average approximates $\sum_{j=1}^m \nabla h_j(\bar{x}_{i,k})$ with decreasing error as time k increases.

The term $\bar{s}_{i,k}$ is essentially an approximation for the average value $(1/m)\sum_{j=1}^m h_j(x_{i,k})$. Therefore, agent i uses $g'(m\bar{s}_{i,k})$ to approximate $g'(\sum_{j=1}^m h_j(x_k))$. In iteration $k+1$, each agent in the network is interested in determining $(1/m)\sum_{j=1}^m h_j(x_{j,k+1})$ while each term $h_i(x_{i,k+1})$ is available only to agent i. If agent i were to use $h_i(x_{i,k+1})$ as the start value, then a large number of consensus steps would be required for agent i to obtain a good approximation to $(1/m)\sum_{j=1}^m h_j(x_{j,k+1})$. As an alternative, we consider a more efficient procedure that uses $\bar{s}_{i,k} + h_i(x_{i,k+1}) - h_i(x_{i,k})$ as the start value to estimate $(1/m)\sum_{j=1}^m h_j(x_{j,k+1})$. In this way, a single consensus step is enough to obtain a sufficiently good approximation. When the difference between $x_{i,k+1}$ and $x_{i,k}$ is small, the difference between $\sum_{j=1}^m h_j(x_{j,k+1})$ and $\sum_{j=1}^m h_j(x_{j,k})$ is also small. Thus, the value $\bar{s}_{i,k} + h_i(x_{i,k+1}) - h_i(x_{i,k})$ is closer to the target value than just $h_i(x_{i,k+1})$, assuming $\bar{s}_{i,k}$ is a good approximation to $(1/m)\sum_{j=1}^m h_j(x_{j,k})$. This approach to tracking the network wide average of a changing statistic is reminiscent of the consensus filters than have been proposed in literature [9].

We now formally establish the convergence of the algorithm. We first characterize the rate of consensus.

LEMMA 5.1 *Let Assumptions* 3.1, 3.2, *and* 3.3 *hold. If* $\{\alpha_k\}$ *is a non-negative non-increasing sequence such that* $\sum_{k=1}^\infty \alpha_k^2 < \infty$, *then*

$$\sum_{k=1}^{\infty} \alpha_k \|x_{i,k} - x_{j,k}\| < \infty \quad \textit{for all } i, j \in V.$$

Proof Note from (9) that

$$x_{i,k+1} = \sum_{j=1}^{m} a_{i,j}(k+1)x_{j,k} + p_{i,k+1}, \tag{10}$$

where $p_{i,k+1} = x_{i,k+1} - \bar{x}_{i,k}$. From (9), the Euclidean projection property in (6) and the boundedness of the gradients in (5), we obtain

$$\|p_{i,k+1}\| \leq \alpha_{k+1}\|g'(m\bar{s}_{i,k})\nabla h_i(x_{i,k})\| \leq \alpha_{k+1}C^2. \tag{11}$$

Since $\sum_k \alpha_k^2 < \infty$ we conclude that $\sum_k \alpha_k \|p_{i,k}\| < \infty$ for each $i \in V$. Therefore, by (10) and (11), the iterates $\{x_{i,k}\}$ satisfy the conditions of Theorem 4.2 and the result follows. ■

An immediate consequence of relation (11) is that the agents achieve consensus asymptotically, i.e. $\lim_{k\to\infty} \|x_{i,k} - x_{j,k}\| = 0$ for all $i, j \in V$.

Define

$$\hat{x}_k = \frac{1}{m}\sum_{j=1}^{m} x_{j,k}, \quad \hat{s}_k = \frac{1}{m}\sum_{j=1}^{m} h_j(\hat{x}_k). \tag{12}$$

We next characterize the rate of consensus for $\{s_{i,k}\}$.

LEMMA 5.2 *Let Assumptions* 3.1, 3.2 *and* 3.3 *hold. If* $\{\alpha_k\}$ *is a non-negative non-increasing sequence such that* $\sum_{k=1}^\infty \alpha_k^2 < \infty$, *then* $\sum_{k=1}^\infty \alpha_k \|s_{i,k} - \hat{s}_k\| < \infty$ *for all* $i \in V$.

Proof Using the triangle inequality, we obtain

$$\|s_{i,k} - \hat{s}_k\| \leq \left\| s_{i,k} - \frac{1}{m}\sum_{j=1}^{m} s_{j,k} \right\| + \left\| \frac{1}{m}\sum_{j=1}^{m} s_{j,k} - \hat{s}_k \right\|. \tag{13}$$

We consider the last term and show that

$$\sum_{i=1}^{m} s_{i,k} = \sum_{i=1}^{m} h_i(x_{i,k}). \tag{14}$$

We will use the induction. For $k = 0$, from (4) we have $s_{i,0} = h_i(x_{i,0})$ and, hence, the hypothesis is true for $k = 0$. Now, assume that the hypothesis is true for $k - 1$. Observe that from Assumption 3.3(d) we have

$$\sum_{i=1}^{m} \bar{s}_{i,k-1} = \sum_{i=1}^{m}\sum_{j=1}^{m} a_{i,j}(k)\bar{s}_{j,k-1} = \sum_{j=1}^{m} s_{j,k-1} = \sum_{i=1}^{m} h_j(x_{j,k-1}),$$

where the last equality follows by the hypothesis. By the definition of $s_{i,k}$ in (9) and the preceding relation, we conclude that

$$\sum_{j=1}^{m} s_{j,k} = \sum_{j=1}^{m} \bar{s}_{j,k-1} + \sum_{j=1}^{m} h_j(x_{j,k}) - \sum_{j=1}^{m} h_j(x_{j,k-1}) = \sum_{j=1}^{m} h_j(x_{j,k}).$$

This proves the induction hypothesis for k and hence (14) follows. Using (14) in (13) and substituting for $\hat{s}_k$, we obtain

$$\|s_{i,k} - \hat{s}_k\| \leq \left\| s_{i,k} - \frac{1}{m}\sum_{j=1}^{m} s_{j,k} \right\| + \left\| \frac{1}{m}\sum_{j=1}^{m} h_j(x_{j,k}) - \frac{1}{m}\sum_{j=1}^{m} h_j(\hat{x}_k) \right\|. \tag{15}$$

Now, we deal with the first term on the right-hand side of (15). Note that we can rewrite

$$s_{i,k+1} = \sum_{j=1}^{m} a_{i,j}(k+1)s_{j,k} + w_{i,k+1}, \tag{16}$$

where $w_{i,k+1} = h_i(x_{i,k+1}) - h_i(x_{i,k})$. Next, we show that $\sum_k \alpha_{k+1}\|w_{i,k+1}\| < \infty$. Since the gradient of h_i is bounded by C, the function h_i is Lipschitz continuous with C. Using this and the definition of $x_{i,k+1}$ in (9), we get

$$\|w_{i,k+1}\| \leq C\|x_{i,k+1} - x_{i,k}\| \leq C\|P_X[\bar{x}_{i,k} - \alpha_{k+1}g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k})] - x_{i,k}\|.$$

By using the triangle inequality and the Euclidean projection property in (6), we have

$$\begin{aligned}\|w_{i,k+1}\| &\leq C\|\bar{x}_{i,k} - \alpha_{k+1}g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}) - x_{i,k}\| \\ &\leq C(\|\bar{x}_{i,k} - x_{i,k}\| + \alpha_{k+1}\|g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k})\|) \\ &\leq C(\|\bar{x}_{i,k} - x_{i,k}\| + \alpha_{k+1}C^2),\end{aligned} \tag{17}$$

where the last inequality follows from the boundedness of the gradients in (5). Next note that the conditions of Lemma 5.1 are satisfied. Therefore for all $i, j \in V$,

$$\sum_{k=1}^{\infty} \alpha_k \|x_{i,k} - x_{j,k}\| < \infty,$$

and hence $\sum_{k=1}^{\infty} \alpha_k \|x_{i,k} - \bar{x}_{i,k}\| < \infty$ for all $i \in V$. Since $\{\alpha_k\}$ is a non-increasing sequence this implies

$$\sum_{k=1}^{\infty} \alpha_{k+1} \|x_{i,k} - \bar{x}_{i,k}\| < \infty \quad \text{for all } i \in V.$$

Using the preceding inequality in (17) and the fact that $\sum_k \alpha_k^2 < \infty$, we can conclude that

$$\sum_{k=1}^{\infty} \alpha_{k+1} \|w_{i,k+1}\| < \infty \quad \text{for all } i \in V.$$

Thus (16) satisfies the conditions of Theorem 4.2 and we can conclude that for all $i, j \in V$,

$$\sum_{k+1}^{\infty} \alpha_k \|s_{i,k} - s_{j,k}\| < \infty,$$

which implies

$$\sum_{k=1}^{\infty} \alpha_k \left\| s_{i,k} - \frac{1}{m} \sum_{j=1}^{m} s_{j,k} \right\| < \infty \quad \text{for all } i \in V. \tag{18}$$

We now consider the term $\|(1/m) \sum_{j=1}^{m} h_j(x_{j,k}) - (1/m) \sum_{j=1}^{m} h_j(\hat{x}_k)\|$ on the right-hand side of (15). From the Lipschitz continuity of h_j, we have

$$\|h_j(x_{j,k}) - h_j(\hat{x}_k)\| \leq C \|x_{j,k} - \hat{x}_k\|.$$

By the definition of $\hat{x}_k$ in (12), we have $\hat{x}_k = (1/m) \sum_{j=1}^{m} x_{j,k}$, so that

$$\|h_j(x_{j,k}) - h_j(\hat{x}_k)\| \leq C \left\| x_{j,k} - \frac{1}{m} \sum_{i=1}^{m} x_{i,k} \right\| \leq \frac{C}{m} \sum_{i=1}^{m} \|x_{j,k} - x_{i,k}\|.$$

By Lemma 5.1, we obtain for all $i \in V$,

$$\sum_{k=1}^{\infty} \alpha_k \|h_j(\hat{x}_k) - h_j(x_{j,k})\| \leq \frac{C}{m} \sum_{k=1}^{\infty} \alpha_k \sum_{i=1}^{m} \|x_{j,k} - x_{i,k}\| < \infty.$$

Therefore, from the preceding inequality, (15) and (18) we get for all $i \in V$,

$$\sum_{k=1}^{\infty} \alpha_k \|s_{i,k} - \hat{s}_k\| < \infty.$$

■

We next use Lemmas 5.1 and 5.2 to prove convergence to an optimal point.

THEOREM 5.3 *Let Assumptions* 3.1, 3.2 *and* 3.3 *hold. If the stepsize sequence* $\{\alpha_k\}$ *is non-negative, non-increasing, and such that* $\sum_{k=1}^{\infty} \alpha_k = \infty$ *and* $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$, *then there exists a vector* $x^* \in X^*$ *such that* $\lim_{k\to\infty} \|x_{i,k} - x^*\| = 0$ *for all* $i \in V$.

Proof Note that the solution set X^* is non-empty since f is continuous over the set X and X is compact. Fix an arbitrary $x^* \in X^*$. By the Euclidean projection property in (6), we have

$$\begin{aligned}\|x_{i,k+1} - x^*\|^2 &= \|P_X[\bar{x}_{i,k} - \alpha_{k+1} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k})] - x^*\|^2 \\ &\leq \|\bar{x}_{i,k} - x^*\|^2 + \alpha_{k+1}^2 \|g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k})\|^2 \\ &\quad - 2\alpha_{k+1}(\bar{x}_{i,k} - x^*)^{\mathrm{T}} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}). \end{aligned} \tag{19}$$

Using the boundedness of the gradients in (5), we obtain

$$\|g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k})\| \leq C^2.$$

By Assumption 3.3 on the weights and the convexity of the squared Euclidean norm, we have

$$\begin{aligned}\sum_{i=1}^m \|\bar{x}_{i,k} - x^*\|^2 &= \sum_{i=1}^m \left\| \sum_{j=1}^m a_{i,j}(k+1)x_{j,k} - x^* \right\|^2 \\ &\leq \sum_{i=1}^m \sum_{j=1}^m a_{i,j}(k+1)\|x_{j,k} - x^*\|^2 \\ &\leq \sum_{j=1}^m \|x_{j,k} - x^*\|^2. \end{aligned}$$

Summing (19) over all i and using the preceding two relations, we obtain

$$\begin{aligned}\sum_{i=1}^m \|x_{i,k+1} - x^*\|^2 &\leq \sum_{i=1}^m \|x_{i,k} - x^*\|^2 + m\alpha_{k+1}^2 C^4 \\ &\quad - 2\alpha_{k+1} \sum_{i=1}^m (\bar{x}_{i,k} - x^*)^{\mathrm{T}} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}). \end{aligned} \tag{20}$$

By the definitions of $\hat{x}_k$ and $\hat{s}_k$ in (12), we have $\nabla f(\hat{x}_k) = \sum_{i=1}^m g'(m\hat{s}_k)\nabla h_i(\hat{x}_k)$. Using this we can write

$$\begin{aligned}\sum_{i=1}^m (\bar{x}_{i,k} - x^*)^{\mathrm{T}} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}) &= \sum_{i=1}^m (\bar{x}_{i,k} - \hat{x}_k)^{\mathrm{T}} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}) \\ &\quad + \sum_{i=1}^m (\hat{x}_k - x^*)^{\mathrm{T}} (g'(m\bar{s}_{i,k}) - g'(m\hat{s}_k))\nabla h_i(\bar{x}_{i,k}) \\ &\quad + \sum_{i=1}^m (\hat{x}_k - x^*)^{\mathrm{T}} g'(m\hat{s}_k)(\nabla h_i(\bar{x}_{i,k}) - \nabla h_i(\hat{x}_k)) \\ &\quad + (\hat{x}_k - x^*)^{\mathrm{T}} \nabla f(\hat{x}_k). \end{aligned}$$

By the boundedness of the gradients (cf. Eq. (5)), we have

$$\sum_{i=1}^m (\bar{x}_{i,k} - \hat{x}_k)^{\mathrm{T}} g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}) \geq -C^2 \sum_{i=1}^m \|\bar{x}_{i,k} - \hat{x}_k\|.$$

By the compactness of X (Assumption 3.1(b)), gradient boundedness and the Lipschitz gradient assumption (Assumption 3.1(d)), we also have

$$\sum_{i=1}^{m}(\hat{x}_k - x^*)^{\mathrm{T}}(g'(m\bar{s}_{i,k}) - g'(m\hat{s}_k))\nabla h_i(\bar{x}_{i,k}) \geq -DLCm\sum_{i=1}^{m}\|\bar{s}_{i,k} - \hat{s}_k\|,$$

$$\sum_{i=1}^{m}(\hat{x}_k - x^*)^{\mathrm{T}}g'(m\hat{s}_k)(\nabla h_i(\bar{x}_{i,k}) - \nabla h_i(\hat{x}_k)) \geq -DCL\sum_{i=1}^{m}\|\bar{x}_{i,k} - \hat{x}_k\|,$$

where D is the diameter of the set X. Finally, by the convexity of f (Assumption 3.1(c)), we have

$$(\hat{x}_k - x^*)^{\mathrm{T}}\nabla f(\hat{x}_k) \geq f(\hat{x}_k) - f(x^*).$$

Combining the preceding relations, we obtain

$$\sum_{i=1}^{m}(\bar{x}_{i,k} - x^*)^{\mathrm{T}}g'(m\bar{s}_{i,k})\nabla h_i(\bar{x}_{i,k}) \geq (f(\hat{x}_k) - f(x^*)) - C^2\sum_{i=1}^{m}\|\bar{x}_{i,k} - \hat{x}_k\| - DCL\sum_{i=1}^{m}(m\|\bar{s}_{i,k} - \hat{s}_k\| + \|\bar{x}_{i,k} - \hat{x}_k\|).$$

Using this relation in (20), we obtain

$$\sum_{i=1}^{m}\|x_{i,k+1} - x^*\|^2 \leq \sum_{i=1}^{m}\|x_{i,k} - x^*\|^2 - 2\alpha_{k+1}(f(\hat{x}_k) - f^*) + 2C^2\sum_{i=1}^{m}\alpha_{k+1}\|\bar{x}_{i,k} - \hat{x}_k\| + 2DCL\sum_{i=1}^{m}\alpha_{k+1}(m\|\bar{s}_{i,k} - \hat{s}_k\| + \|\bar{x}_{i,k} - \hat{x}_k\|) + m\alpha_{k+1}^2C^4.$$

From Lemma 5.1 and the fact that the sequence $\{\alpha_{k+1}\}$ is non-increasing, we have

$$\sum_{k=1}^{\infty}\alpha_{k+1}\|\bar{x}_{i,k} - \hat{x}_k\| < \infty \quad \text{for all } i \in V.$$

Using Lemma 5.2 and the fact that the sequence $\{\alpha_{k+1}\}$ is non-increasing, we obtain

$$\sum_{k=1}^{\infty}\alpha_{k+1}\|\bar{s}_{i,k} - \hat{s}_k\| < \infty \quad \text{for all } i \in V.$$

Thus, the conditions of Lemma 4.1 are satisfied and we can conclude that $\|x_{i,k+1} - x^*\|$ converges for every $x^* \in X^*$ and every $i \in V$, and

$$\sum_{k=1}^{\infty}\alpha_{k+1}\left(f(\hat{x}_k) - f^*\right) < \infty.$$

This and Lemma 5.1 imply that the sequences $\{x_{i,k}\}$, $i \in V$, must converge to a common point in the set X^*. ■

6. Extensions

We next discuss two extensions of the problem in (1) and generalize the algorithm in (9) to solve these extensions.

6.1 *Extension I*

Consider the following general distributed optimization problem

$$\begin{aligned} &\text{minimize} \quad \sum_{j=1}^{m} g_j \left(\sum_{i=1}^{m} h_{i,j}(x) \right), \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{21}$$

where $g_j : \Re \to \Re$, $X \subseteq \Re^p$, $h_{i,j} : X \to \Re$ for all i, $j \in V$. The functions $h_{i,j}$, $j \in V$, are known only to agent i. The functions g_j and the set X are globally known. Note that the index set for j can be other than V. We prefer to keep $j \in V$ for simplicity of notation.

We can modify the algorithm in (9) to solve problem (21) as follows. Let $s_{i,k}$ now denote agent i's estimate of the vector $[(1/m)\sum_{r=1}^{m} h_{r,1}(x_{i,k}) \ \ldots \ (1/m)\sum_{r=1}^{m} h_{r,m}(x_{i,k})]^{\mathrm{T}}$. Agent i recursively generates $x_{i,k+1}$ and $s_{i,k+1}$ according to the following rules:

$$\begin{aligned} \begin{bmatrix} \bar{x}_{i,k} \\ \bar{s}_{i,k} \end{bmatrix} &= \sum_{j=1}^{m} a_{i,j}(k+1) \begin{bmatrix} x_{j,k} \\ s_{j,k} \end{bmatrix}, \\ x_{i,k+1} &= P_X \left[\bar{x}_{i,k} - \alpha_{k+1} \sum_{j=1}^{m} g_j'(m[\bar{s}_{i,k}]_j) \nabla h_{i,j}(\bar{x}_{i,k}) \right], \\ s_{i,k+1} &= \bar{s}_{i,k} + \begin{bmatrix} h_{i,1}(\bar{x}_{i,k+1}) \\ \vdots \\ h_{i,m}(\bar{x}_{i,k+1}) \end{bmatrix} - \begin{bmatrix} h_{i,1}(\bar{x}_{i,k}) \\ \vdots \\ h_{i,m}(\bar{x}_{i,k}) \end{bmatrix}. \end{aligned} \tag{22}$$

Here $[\bar{s}_{i,k}]_j$ denotes the jth component of the vector $\bar{s}_{i,k}$. In the $(k+1)$th iteration, agent i receives $x_{j,k}$ and $s_{j,k}$ from its current immediate neighbours and computes a weighted averages $\bar{x}_{i,k}$ and $\bar{s}_{i,k}$. The weighted average is then updated using locally available information (functions g_j and $h_{i,j}$, and the set X) to generate $x_{i,k+1}$ and $s_{i,k+1}$. The algorithm is initialized with

$$x_{i,0} \in X, \quad s_{i,0} = \begin{bmatrix} h_{i,1}(x_{i,0}) \\ \vdots \\ h_{i,m}(x_{i,0}) \end{bmatrix} \quad \text{for all } i \in V.$$

6.2 *Extension II*

We next consider a generalization of problem (21), where the objective function has the same form but the knowledge about the functions is distributed differently. For this, we write problem (21) in the following form:

$$\begin{aligned} &\text{minimize} \quad g_1 \left(\sum_{\ell=1}^{m} h_\ell(x) \right) + g_2 \left(\sum_{\ell=m+1}^{2m} h_\ell(x) \right) + \cdots + g_m \left(\sum_{\ell=m(m-1)+1}^{m^2} h_\ell(x) \right), \\ &\text{subject to} \quad x \in X, \end{aligned} \tag{23}$$

where $g_j : \Re \to \Re$ for all j, $X \subseteq \Re^p$ and $h_\ell : X \to \Re$ for all ℓ.

Now, consider the case where each function h_ℓ is known to only one agent, denoted by ℓ. This would give rise to a network of m^2 agents indexed by ℓ, where $\ell \in W = \{1, \ldots, m^2\}$. In addition,

we let each agent ℓ know the function g_j that depends on its function h_ℓ. To formally describe this, we let $j(\ell) = \lceil \ell/m \rceil$, and note that in this notation, agent ℓ knows $g_{j(\ell)}$. Essentially, the function g_j is known to agents $\ell = m(j-1)+1, \ldots, mj$. As before, the set X is known to all the agents.

We modify the algorithm in (9) to solve problem (23) as follows. Let $x_{\ell,k} \in \Re^p$ denote agent ℓ estimate of the optimal point at time k. Let $s_{\ell,k} \in \Re^m$ denote agent ℓ estimate of the following vector

$$\left[\frac{1}{m} \sum_{r=1}^{m} h_r(x_{\ell,k}) \; \frac{1}{m} \sum_{r=m+1}^{2m} h_r(x_{\ell,k}) \; \ldots \; \frac{1}{m} \sum_{r=m(m-1)+1}^{m^2} h_r(x_{\ell,k}) \right]^{\mathrm{T}}.$$

For each $\ell \in W$, agent ℓ recursively generates vectors $x_{\ell,k+1}$ and $s_{\ell,k+1}$ as follows:

$$\begin{aligned}
\begin{bmatrix} \bar{x}_{\ell,k} \\ \bar{s}_{\ell,k} \end{bmatrix} &= \sum_{r=1}^{m^2} a_{\ell,r}(k+1) \begin{bmatrix} x_{r,k} \\ s_{r,k} \end{bmatrix}, \\
x_{\ell,k+1} &= P_X[\bar{x}_{\ell,k} - \alpha_{k+1} g'_{j(\ell)}(m[\bar{s}_{\ell,k}]_{j(\ell)}) \nabla h_\ell(\bar{x}_{\ell,k})], \\
[s_{\ell,k+1}]_j &= \begin{cases} [\bar{s}_{\ell,k}]_j & \text{for } j \neq j(\ell), \\ [\bar{s}_{\ell,k}]_j + h_\ell(x_{\ell,k+1}) - h_\ell(x_{\ell,k}) & \text{for } j = j(\ell). \end{cases}
\end{aligned} \tag{24}$$

In the $(k+1)$th iteration, agent ℓ receives $x_{j,k}$ and $s_{j,k}$ from its current immediate neighbours and calculates weighted averages $\bar{x}_{\ell,k}$ and $\bar{s}_{\ell,k}$. These averages are then updated using locally available information (functions $g_{j(\ell)}$ and h_ℓ, and the set X) to generate $x_{\ell,k+1}$ and $s_{\ell,k+1}$. Note that agent ℓ updates $j(\ell)$th coordinate of the vector $s_{\ell,k+1}$ differently from the other coordinates, where $j(\ell)$ is the index of the function g_j that depends on the function h_ℓ known to agent ℓ. The other coordinates of $s_{\ell,k+1}$ are updated based on the consensus-step only.

The algorithm is initialized with $x_{\ell,0} \in X$ and $s_{\ell,0} = h_\ell(x_{\ell,0}) e_{j(\ell)}$, where $e_{j(\ell)}$ is the unit vector with $j(\ell)$th component equal to 1.

7. Application to distributed regression

Our main motivation for studying this class of optimization problems is distributed regression. Regression involves modelling a response variable as a function of one or more predictor variables using samples of the response variable at different levels of the predictor variables (see, e.g. textbook [4]). The response variable is viewed as a random variable and its mean is modelled as a known function of the predictor variables and an unknown regression parameter. The goal in regression is to determine the unknown regression parameter value that best explains the observed data. This is usually done by defining a 'goodness of fit' cost criterion and choosing the parameter value that minimizes the criterion. Thus, regression involves solving an optimization problem in which the objective function is decided by the observed data and the optimization is with respect to the regression parameter.

Let us consider regression in large peer to peer systems like sensor networks. In such systems, different parts of the regression data are collected by different agents in the network. Data in the network is distributed *vertically* when each agent has access only to a subset of the predictor variables' values in each sample. Thus the optimization problem that specifies the optimal parameter value becomes a distributed optimization problem. This is in contrast to *horizontally* distributed data where the agents observe complete samples but no agent has access to all the samples [5]. This leads to a different distributed optimization problem. Finally, data could be both vertically

and horizontally distributed, which leads to a general distributed optimization problem. As we will see, when the regression function has an additive form, the resultant optimization problems are special cases of the general optimization problems discussed in this paper.

Distributed regression in networks with structured connectivity (such as ring structure) has been considered, for example, for sensor networks (see [2,3,11,12] and the literature therein). On the other hand, the literature on distributed regression in large networks with arbitrary connectivity structure is limited. The chapter [17] surveys sequential linear ordinary least square algorithms proposed for horizontally distributed data in large unstructured networks. In addition, a recent related paper is [1], where both linear regression and model monitoring algorithms are proposed for horizontally distributed data. For horizontal distributed regression in small structured networks, algorithms with a meta-learning approach are proposed in [19] and algorithms that emphasize privacy are proposed in [6]. We note that in [19], the data sets are heterogeneous but still horizontal. To the best of our knowledge, there is no literature on vertical distributed regression in unstructured networks. The papers [5,13,20] study vertical distributed regression in structured networks with a central fusion centre.

We mathematically describe the regression problem below. Let R be the response variable and $U^{(i)}$, $i \in V = \{1, \ldots, m\}$, denote the ith predictor variable. For convenience, we take $U^{(i)}$ and R to be scalar variables. The regression parameter is x, $x \in \Re^p$, and the regression function has the following additive form:

$$\mathsf{E}[R \mid U^{(1)}, \ldots, U^{(m)}] = \sum_{i=1}^{m} f_i(x, U^{(i)}) \quad \text{for } x \in X, \tag{25}$$

where $X \subseteq \Re^p$ and $f_i : X \times \Re \to \Re$. Note that the case $f_i(x, U^{(i)}) = x_i U^{(i)}$ corresponds to linear regression.

A total of m observation samples are available.[1] The jth response sample is denoted by r_j and it is measured at the value $u_j^{(i)}$ of the ith predictor variable. The optimal parameter x^* is chosen as follows:

$$x^* = \operatorname{Arg\,min}_{x \in X} \frac{1}{m} \sum_{j=1}^{m} q\left(r_j - \sum_{i=1}^{m} f_i(x, u_j^{(i)})\right). \tag{26}$$

In least square regression, the function $q(t)$ is given by t^2. Other functions such as Huber's function may be used for robust regression.

7.1 *Horizontal regression*

Consider a network of m agents indexed by j, $j \in V = \{1, \ldots, m\}$. When the data is horizontally distributed, only agent j has access to the jth sample, i.e. $r_j, u_j^{(1)}, \ldots, u_j^{(m)}$. This case is illustrated in the plot to the left in Figure 1. Observe now that problem (26) is a special case of the distributed optimization problem in (1) with

$$g(t) = t, \quad h_j(x) = q\left(r_j - \sum_{i=1}^{m} f_i(x, u_j^{(i)})\right).$$

7.2 *Vertical regression*

Consider a network of m agents indexed by i, $i \in V$. The data are vertically distributed. Therefore, only agent i has access to the samples of the ith predictor random variable, i.e. $\{u_j^{(i)}\}_{j \in V}$ is known

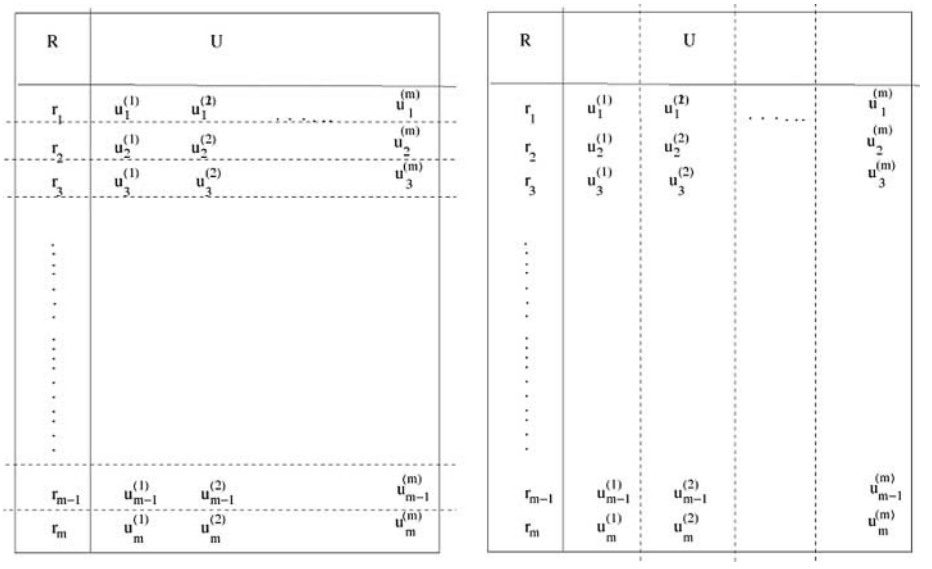

Figure 1. Horizontally/vertically distributed regression data: R is the response variable and r_j is the jth response sample. The variable U is the predictor, where the value $u_j^{(i)}$ is the ith predictor variable for sample r_j. For simplicity, it is assumed that the number of samples and the predictors is the same. The plot to the left shows the case when the data in each row is available to a single agent, i.e. the variables r_j and $u_j^{(1)}, \ldots, u_j^{(m)}$ are known to agent j. The plot to the left shows the case when the data in each column of the predictor (U-data) is available to a single agent and all agents have access to all the samples, i.e. the variables $r_1, \ldots, r_m$ and $u_1^{(i)}, \ldots, u_m^{(i)}$ are known to agent i.

only to agent i. The response variable samples are available to all the agents. This situation is illustrated in the plot to the right in Figure 1. Problem (26) can be seen to be a special case of problem (21) by letting

$$g_j = q, \quad h_{i,j}(x) = \frac{r_j}{m} - f_i\left(x, u_j^{(i)}\right).$$

This formulation can be relaxed to the case when there is an $(m+1)$th agent that collects the samples of the response variable.

7.3 *Vertical and horizontal regression*

We consider a network of m^2 agents. Each agent is indexed by ℓ, where $\ell \in W$. When the data are vertically and horizontally distributed, agent $i + m(j-1)$ has access to only $u_j^{(i)}$ and r_j. In this case, problem (26) is a special case of (23) with

$$g_j = q, \quad h_{m(j-1)+i} = \frac{r_j}{m} - f_i\left(x, u_j^{(i)}\right).$$

8. Discussion

In this paper, we have introduced a new class of distributed optimization problems. We have proposed and analysed a distributed and local algorithm for solving such problems in a network of agents with partial information about the problem data. We have established convergence of the algorithm to an optimal point. The algorithm combines the ideas of consensus-based gradient schemes [14] and consensus-based tracking schemes [9]. The algorithm has a potential use in other distributed contexts, such as power control [18].

Implicit in the development of (3) is an algorithm to track the network wide average of a statistic that 'slowly' changes with time. This is similar to the distributed filters proposed by [9] though the analysis perspective is different.

There are multiple directions for future work. First, it is important to understand the effect of gradient stochastic errors on the algorithm. This will help in extending the methods to sequential distribution regression as in [14,16]. Second, for practical implementations, it is important to obtain bounds on the performance of the algorithm as a function of number of iterations. Third, we have assumed that the objective function is smooth and convex. In order to handle ℓ_1-regularized problems, it is important to extend the results to non-smooth convex functions. Also, it is of interest to obtain convergence results (typically to a stationary point) when the functions are non-convex. Finally, the algorithm proposed in this paper is synchronous. Ideas similar to those used in [15] can be employed to accommodate an asynchronous implementation of the algorithm.

Acknowledgements

This work has been supported by NSF Career Grant CMMI 07-42538.

Note

1. There is no loss of generality in fixing the number of observations to be m.

References

[1] K. Bhaduri and H. Kargupta, *A scalable local algorithm for distributed multivariate regression*, Stat. Anal. Data Min. 1(3) (2008), pp. 177–194.
[2] V. Delouille, R. Neelamani, and R. Baraniuk, *Robust distributed estimation in sensor networks using the embedded polygons algorithm*, Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, Berkeley, CA, 2004, pp. 405–413.
[3] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, *Distributed regression: An efficient framework for modeling sensor network data*, Proceedings of Information Processing in Sensor Network, Berkeley, CA, 2004, pp. 1–10.
[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer-Verlag, New York, 2001.
[5] D. Hershberger and H. Kargupta, *Distributed multivariate regression using wavelet-based collective data mining*, J. Parallel Distrib. Comput. 61(3) (2001), pp. 372–400.
[6] A. Karr, X. Lin, A. Sanil, and J. Reiter, *Secure regression on distributed databases*, J. Comput. Graph. Stat. 14(2) (2005), 263–279.
[7] U. Khan, S. Kar, and J. Moura, *Handbook on Sensor and Array Processing*, S. Haykin and K.J. Ray Liu, eds. Distributed Algorithms in Sensor Networks, Wiley-Interscience, New York, 2009.
[8] A. Nedić, A. Olshevsky, A. Ozdaglar, and J.N. Tsitsiklis, *Distributed subgradient algorithms and quantization effects*, Proceedings of the 47th IEEE Conference on Decision and Control, Cancún, Mexico, 2008.
[9] R. Olfati-Saber and J. Shamma, *Consensus filters for sensor networks and distributed sensor fusion*, in *Proceedings of the IEEE Conference on Decision and Control*, Seville, Spain, Vol. 7, 2005, pp. 6698–6703.
[10] B.T. Polyak, *Introduction to Optimization*, Optimization Software Inc., New York, 1987.
[11] J.B. Predd, S.R. Kulkarni, and H.V. Poor, *Regression in sensor networks: Training distributively with alternating projections*, Proceedings of the SPIE Conference on Advanced Signal Processing Algorithms and Implementations XV, San Diego, CA, 2005, pp. 591006-1–591006-15.
[12] J.B. Predd, S.R. Kulkarni, and H.V. Poor, *A collaborative training algorithm for distributed learning*, IEEE Trans. Inform. Theory 55(4) (2009) 1856–1871.
[13] A. Sanil, A. Karr, and X. Lin, *Privacy preserving regression modelling via distributed computation*, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, 2004, pp. 677–682.
[14] S. Sundhar Ram, A. Nedić, and V.V. Veeravalli, *Distributed stochastic subgradient algorithm for convex optimization*, 2008. Available at http://arxiv.org/abs/0811.2595.
[15] S. Sundhar Ram, A. Nedić, and V.V. Veeravalli, *Asynchronous gossip algorithms for stochastic optimization*, Proceedings of the 48th Conference on Control, Decision and Control, Shanghai, China, 2009.

[16] S. Sundhar Ram, A. Nedić, and V.V. Veeravalli, *Incremental stochastic sub-gradient algorithms for convex optimization*, SIAM J. Optim. 20(2) (2009) 691–717.
[17] S. Sundhar Ram, V.V. Veeravalli, and A. Nedić, *Sensor Networks: When Theory Meets Practice*, G. Ferrari, ed., Distributed and Recursive Estimation, Springer, Berlin/Heidelberg, 2009.
[18] S. Sundhar Ram, V.V. Veeravalli, and A. Nedić, *Distributed and non-autonomous power control through distributed convex optimization*, The 28th IEEE Conference on Computer Communications INFOCOM, Rio de Janeiro, Brazil, 19–25 April 2009, pp. 3001–3005.
[19] Y. Xing, M. Madden, J. Duggan, and G. Lyons, *Advances in Intelligent Data Analysis V*, Vol. 2811/2003, Lecture Notes in Computer Science, Distributed Regression for Heterogeneous Datasets, Springer, Berlin/Heidelberg, 2003, pp. 544–553.
[20] H. Yu and E.-C. Chang, *Distributed multivariate regression based on influential observations*, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, 2003, pp. 679–684.

A. Appendix

Proof of Theorem 4.2 In view of Assumption 3.3 we can rewrite (8) as

$$\theta_{i,k+1} = \sum_{j=1}^{m} a_{i,j}(k+1)\theta_{j,k} + \epsilon_{i,k+1}, \quad \theta_{i,0} = \theta_i. \tag{A1}$$

Let $A(k)$ be the matrix with (i, j)th entry equal to $a_{i,j}(k)$. As a consequence of Assumptions 3.3(a), (b) and (d), the matrix $A(k)$ is doubly stochastic (its entries are non-negative, and the sum of its entries in every row and in every column is equal to 1). Define, for all k, s with $k \geq s$,

$$\Phi(k, s) = A(k)A(k-1)\cdots A(s+1). \tag{A2}$$

We first state a result from [8, Corollary 1] on the convergence properties of the matrix $\Phi(k, s)$. Let $[\Phi(k, s)]_{i,j}$ denote the (i, j)th entry of the matrix $\Phi(k, s)$, and let $e \in \Re^m$ be the column vector with all entries equal to 1. ■

LEMMA A.1 *Let Assumptions* 3.2 *and* 3.3 *hold. Then*

(1) $\lim_{k\to\infty} \Phi(k, s) = (1/m)ee^{\mathrm{T}}$ *for all* s.
(2) *Further, the convergence is geometric and the rate of convergence is given by*

$$\left| [\Phi(k, s)]_{i,j} - \frac{1}{m} \right| \leq \gamma\beta^{k-s},$$

where

$$\gamma = \left(1 - \frac{\eta}{4m^2}\right)^{-2}, \quad \beta = \left(1 - \frac{\eta}{4m^2}\right)^{1/Q}.$$

We will also use the following result from [14] (Lemma 3.1(b)).

LEMMA A.2 *Let* $\{\zeta_k\}$ *be a scalar sequence. If* $\zeta_k \geq 0$ *for all* k, $\sum_k \zeta_k < \infty$ *and* $0 < \beta < 1$, *then* $\sum_{k=0}^{\infty} \left(\sum_{\ell=0}^{k} \beta^{k-\ell}\zeta_\ell\right) < \infty$.

Using the matrices $\Phi(k, s)$ defined in (A2) we can write

$$\theta_{j,k+1} = \sum_{i=1}^{m} [\Phi(k+1, 0)]_{j,i}\theta_{i,0} + \epsilon_{j,k+1} + \sum_{\ell=1}^{k} \left(\sum_{i=1}^{m} [\Phi(k+1, \ell)]_{j,i}\epsilon_{i,\ell} \right). \tag{A3}$$

Define

$$\phi_k = \frac{1}{m} \sum_{i=1}^{m} \theta_{i,k}.$$

Using (A1), we can also rewrite ϕ_k as

$$\phi_{k+1} = \frac{1}{m}\left(\sum_{i=1}^{m}\sum_{j=1}^{m} a_{i,j}(k+1)\theta_{j,k} + \sum_{i=1}^{m}\epsilon_{i,k+1}\right)$$
$$= \frac{1}{m}\left(\sum_{j=1}^{m}\left(\sum_{i=1}^{m} a_{i,j}(k+1)\right)\theta_{j,k} + \sum_{i=1}^{m}\epsilon_{i,k+1}\right).$$

In the view of the doubly stochasticity of the weights, we have $\sum_{i=1}^{m} a_{i,j}(k+1) = 1$, implying that

$$\phi_{k+1} = \frac{1}{m}\left(\sum_{j=1}^{m}\theta_{j,k} + \sum_{i=1}^{m}\epsilon_{i,k+1}\right) = \phi_k + \frac{1}{m}\sum_{i=1}^{m}\epsilon_{i,k+1}.$$

Therefore

$$\phi_{k+1} = \phi_0 + \frac{1}{m}\sum_{\ell=1}^{k+1}\sum_{i=1}^{m}\epsilon_{i,\ell} = \frac{1}{m}\sum_{i=1}^{m}\theta_{i,0} + \frac{1}{m}\sum_{\ell=1}^{k+1}\sum_{i=1}^{m}\epsilon_{i,\ell}. \tag{A4}$$

Substituting for ϕ_{k+1} from (A4) and for $\theta_{j,k+1}$ from (A3), we obtain

$$\|\phi_{k+1} - \theta_{j,k+1}\| = \left\| \frac{1}{m}\sum_{i=1}^{m}\theta_{i,0} + \frac{1}{m}\sum_{\ell=1}^{k+1}\sum_{i=1}^{m}\epsilon_{i,\ell} - \left(\sum_{i=1}^{m}[\Phi(k+1,0)]_{j,i}\theta_{i,0} + \epsilon_{j,k+1} + \sum_{\ell=1}^{k}\sum_{i=1}^{m}[\Phi(k+1,\ell)]_{j,i}\epsilon_{i,\ell}\right)\right\|$$
$$= \left\| \sum_{i=1}^{m}\left(\frac{1}{m} - [\Phi(k+1,0)]_{j,i}\right)\theta_{i,0} + \sum_{\ell=1}^{k}\sum_{i=1}^{m}\left(\frac{1}{m} - [\Phi(k+1,\ell)]_{j,i}\right)\epsilon_{i,\ell} + \frac{1}{m}\sum_{i=1}^{m}\epsilon_{i,k+1} - \epsilon_{j,k+1}\right\|.$$

Therefore, for all $j \in V$ and all k,

$$\|\phi_{k+1} - \theta_{j,k+1}\| \le \sum_{i=1}^{m}\left|\frac{1}{m} - [\Phi(k+1,0)]_{j,i}\right| \|\theta_{i,0}\| + \sum_{\ell=1}^{k}\sum_{i=1}^{m}\left|\frac{1}{m} - [\Phi(k+1,\ell)]_{j,i}\right| \|\epsilon_{i,\ell}\| + \frac{1}{m}\sum_{i=1}^{m}\|\epsilon_{i,k+1}\| + \|\epsilon_{j,k+1}\|.$$

We can bound $\|\theta_{i,0}\| \le \max_{i\in V}\|\theta_{i,0}\|$. Further, we can use the rate of convergence result from Lemma A.1 to bound $|(1/m) - [\Phi(k,\ell)]_{j,i}|$. We obtain

$$\|\phi_{k+1} - \theta_{j,k+1}\| \le m\gamma\beta^{k+1}\max_{i\in V}\|\theta_{i,0}\| + m\gamma\sum_{\ell=1}^{k}\beta^{k+1-\ell}\max_{i\in V}\|\epsilon_{i,\ell}\| + \frac{1}{m}\sum_{i=1}^{m}\|\epsilon_{i,k+1}\| + \|\epsilon_{j,k+1}\|.$$

Multiplying both sides by α_{k+1} and using the fact that the sequence $\{\alpha_k\}$ is non-negative and non-increasing, we have

$$\begin{aligned}\sum_{k=1}^{\infty}\alpha_{k+1}\|\phi_{k+1}-\theta_{j,k+1}\| \leq & m\gamma \max_{i\in V}\|\theta_{i,0}\|\sum_{k=1}^{\infty}\alpha_{k+1}\beta^{k+1}\\ &+ m\gamma\sum_{k=1}^{\infty}\sum_{\ell=1}^{k}\beta^{k+1-\ell}\left(\alpha_\ell \max_{i\in V}\left\|\epsilon_{i,\ell}\right\|\right)\\ &+\sum_{k=1}^{\infty}\frac{\alpha_{k+1}}{m}\sum_{i=1}^{m}\|\epsilon_{i,k+1}\|+\sum_{k=1}^{\infty}\alpha_{k+1}\left\|\epsilon_{j,k+1}\right\|.\end{aligned}$$

Since the sequence $\{\alpha_{k+1}\}$ is bounded and $0<\beta<1$, the first term is finite. By letting $\zeta_k = \alpha_k \max_{i\in V}\|\epsilon_{i,k}\|$ in Lemma A.2 and using the assumption $\sum_{k=1}^{\infty}\alpha_{k+1}\|\epsilon_{i,k+1}\|<\infty$, we see that the second term is finite. Also, by $\sum_{k=1}^{\infty}\alpha_{k+1}\|\epsilon_{i,k+1}\|<\infty$, the last two terms are finite. Therefore, we conclude that

$$\sum_{k=1}^{\infty}\alpha_{k+1}\|\phi_{k+1}-\theta_{j,k+1}\|<\infty \quad \text{for all } i\in V.$$

Using the triangle inequality we can write for all $i, j \in V$,

$$\sum_{k=1}^{\infty}\alpha_{k+1}\|\theta_{i,k+1}-\theta_{j,k+1}\| \leq \sum_{k=1}^{\infty}\alpha_{k+1}\|\phi_{k+1}-\theta_{j,k+1}\|+\sum_{k=1}^{\infty}\alpha_{k+1}\|\phi_{k+1}-\theta_{i,k+1}\|<\infty.$$