INTERNATIONAL JOURNAL OF PARALLEL, EMERGENT AND DISTRIBUTED SYSTEMS, 2017
http://dx.doi.org/10.1080/17445760.2017.1294690

Taylor & Francis
Taylor & Francis Group

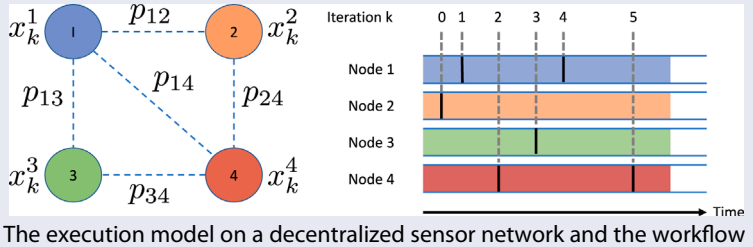
Asynchronous broadcast-based decentralized learning in sensor networks

Liang Zhao[a], Wen-Zhan Song[b], Xiaojing Ye[c] and Yujie Gu[d]

[a]Department of Computer Science, Georgia State University, Atlanta, GA, USA; [b]School of Electrical & Computer Engineering, University of Georgia, Athens, GA, USA; [c]Department of Mathematics & Statistics, Georgia State University, Atlanta, GA, USA; [d]Department of Electrical & Computer Engineering, Temple University, Philadelphia, PA, USA

ABSTRACT

In this paper, we study the problem of decentralized learning in sensor networks in which local learners estimate and reach consensus to the quantity of interest inferred globally while communicating only with their immediate neighbours. The main challenge lies in reducing the communication cost in the network, which involves inter-node synchronisation and data exchange. To address this issue, a novel asynchronous broadcast-based decentralized learning algorithm is proposed. Furthermore, we prove that the iterates generated by the developed decentralized method converge to a consensual optimal solution (model). Numerical results demonstrate that it is a promising approach for decentralized learning in sensor networks.



The execution model on a decentralized sensor network and the workflow of asynchronous computing.

ARTICLE HISTORY
Received 19 December 2016
Accepted 8 February 2017

KEYWORDS
Asynchronous algorithm; big data; decentralized learning; sensor network

1. Introduction

In recent years much attention has been paid to the fully distributed (decentralized) consensus optimization problem, especially in applications like distributed machine learning, multi-agent optimization, etc. When big data is generated in a distributed fashion over a sensor network, decentralized computing is considered to tackle the learning(estimation) problem since it will be very costly and sometimes infeasible to gather all the raw data in the sensor network due to bandwidth and energy limitation. In this scenario, data processing is highly preferred to be done in the network and close to where the data is generated. Decentralized optimization is seen as a scalable approach to solve the aforementioned problem of big data computing in sensor networks by leveraging the computational capacity of all the nodes. In addition, it can balance the communication load among the nodes and preserve the privacy of each node.

CONTACT Liang Zhao lzhao2@ggc.edu

The problem in this paper has a general form as follows. Consider an undirected connected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ denotes the node set and $\mathcal{E}$ is the edge set. The size of network is $m = |\mathcal{V}|$ (cardinality of the set $\mathcal{V}$) and two nodes i, j are called neighbours if $(i, j) \in \mathcal{E}$. Now each node (sensor or agent) i privately holds an objective $F_i : \mathbb{R}^n \to \mathbb{R}$, which describes the data and acquisition process at node i. The goal is to find the global consensus solution $x \in \mathbb{R}^n$ to minimise the optimization problem as follows.

$$\min_{x \in \mathbb{R}^n} \left\{ F(x) := \sum_{i=1}^{m} F_i(x) \right\}. \tag{1}$$

In general, solving (1) in (wireless) sensor networks requires careful algorithm design. Broadcasting is usually preferred in sensor networks since communication is more energy and time consuming than computation. In addition, broadcasting can improve the consensus speed by diffusing information to more nodes with the same amount of cost. However, huge amount of coordination is required for this dynamic network if a synchronous algorithm is used. In contrast, an asynchronous method such that each node can decide its action independently would be more appropriate in this situation.

1.1. Related work

A number of synchronous methods have been proposed in the past years, see, for example [1–11], and the references therein. If we want to apply the aforementioned algorithms into sensor networks, synchronisation among neighbouring nodes is inevitably required. Furthermore, for each node, it has to wait for its slowest neighbour in order to perform its own update. Distributed optimization methods for asynchronous models have been designed in [12–14]. Wei [12] and Iutzeler [13] leverage alternating direction method of multipliers for the computation part, and in each iteration, one node needs to randomly wake up one of its neighbours to exchange information. However, the communication schemes in these works are based on unicast, which is much less preferable than broadcast communication, especially in real-world wireless sensor network scenario. Work [14] proposes an asynchronous model for distributed optimization, while in its model each node maintains a partial vector of the global variable. It is different from our goal of decentralized consensus such that each node contains an estimate of the global common interest. The first broadcast-based distributed/decentralized consensus method was proposed in [15] for consensus average problem without objective. However, the generated consensus solution can not guarantee to be the true average. Inspired by the push-sum algorithm in [16], several works have been done recently to overcome the issue [17,18]. In [17], Iutzeler applies the push-sum algorithm into the broadcast gossip model in [15], and develops an algorithm that requires the node to broadcast a pair of variables (instead of only the solution variable) in the communication stage. This method enjoys not only the similar convergence speed as the broadcast gossip in [15], but also the guarantee of converging to the true average. While the broadcast-based works above are developed for consensus average problem, it is essential to investigate problems with 'real' objective functions. Nedic [19] first fills this gap by considering general decentralized convex optimization under the asynchronous broadcast setting. It adopted the asynchronous broadcast model in [15] and developed a gradient-based update rule for its computation.

1.2. Motivation and contribution

There are two important motivations and contributions of this paper. One is adopting the broadcast-based communication, which is a preferred advantage with wireless networks. One transmission can be received by all neighbours, hence the transmitter does not need to send to neighbours one after another. It saves the energy and speeds up the convergence (since more nodes would be performing update in one round). Another is asynchronous execution, which enables local nodes to take actions independent of other nodes. Each node then does not need to wait for its slowest neighbour for update, which is required in synchronous algorithms (the synchronisation is very costly

and even infeasible in some networks). These settings are realistic for a class of practical sensor network applications. To our best knowledge, we are among the first to propose them.

In this work, we modify Nedic's algorithm [19] in order to fit the nature of applications in sensor networks. We propose a suitable scheme, which is more communication efficient than Nedic's method. Our algorithm has higher complexity than Nedic's algorithm in the first several iterations but it is not the main concern here since the computation time in general is much less comparing to the communication time. We also analyse and prove the convergence of the proposed decentralized consensus algorithm in terms of objective function and disagreements between nodes. Simulation results demonstrate that the proposed algorithm converges faster than the benchmark in terms of number of communication rounds (total execution time).

1.3. Notations and Assumptions

Notation: *Let $x \in \mathbb{R}^n$ be a column vector in problem (1), and $x^i \in \mathbb{R}^n$ be the local copy held privately by node i for every $i \in \mathcal{V}$. Without further remark, vectors are all column vectors. Subscript k is outer iteration number, which is also the number of communication. $\mathbb{E}(\cdot)$ denotes the expectation operator.*

Each sensor node is assumed to have its local clock that ticks at a user-customised Poisson rate for unit time, which is independent of the clocks of the other nodes. Each node broadcasts its current estimate to its neighbours at each tick of its local clock. During broadcasting, each sensor receives neighbours' information subject to link failures. For example, when node *i* broadcasts, its neighbour *j* will receive *i*'s iterate with probability p_{ij}. It is equivalent to consider a virtual global clock existing in the network for the algorithm analysis. Since the Poisson clock of each node (suppose rate = 1) is independent of each other, it is same as a global clock with Poisson rate *m*. We can then analyse the problem given that in each global iteration only one node broadcasts its value. There are several additional assumptions adopted in this paper as follows.

Assumption 1: *The network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected. The link failure between node i and j is independent of other links and identically distributed. The probability of a successful connection between node i and j is denoted as p_{ij}.*

Assumption 2: *The solution set of (1) is nonempty. The private local objective function F_i, $i \in \mathcal{V}$ is differentiable and convex.*

Assumption 3: *The gradient of function F_i is bounded such that $\|\nabla F_i\| \leq G$, where $G > 0$ is some positive number.*

Note that the settings above are common for analysis in asynchronous models [15,19,20].

2. Algorithm design

2.1. Proposed algorithm

In this section, we first describe the asynchronous broadcast-based algorithm. Assume in iteration *k* (according to the virtual global Poisson clock), one node i_k is randomly chosen from $\mathcal{V}$ and broadcasts its most recent value $x_{k-1}^{i_k}$ to a subset J_k of its neighbours $N(i_k)$. Then each node $i \in J_k$ performs the following computation using its own F_i:

$$x_k^i = \arg\min_{x \in \mathbb{R}^n} \left\{ \frac{1}{2\alpha_{i,k}} \|x - (\theta x_{k-1}^{i_k} + (1-\theta) x_{k-1}^i)\|^2 + F_i(x) \right\} \tag{2}$$

where $\theta \in (0,1)$ (e.g. $\theta = \frac{1}{2}$) is the weight factor and F_i is the objective function at node *i*. For nodes $i \notin J_k$ and node i_k, there is no update performed. The step size $\alpha_{i,k}$ is set to $1/\Gamma_k(i)$, where $\Gamma_k(i)$ is the number of updates performed at node *i* up to iteration *k* (same as the one in [19]). Initially, we set $J_k = N(i_k)$, which indicates reliable broadcast communication. With iterations we construct J_k such that it consists only nodes *j* where the broadcast from i_k successfully arrives at *j* with probability p_{ij}.

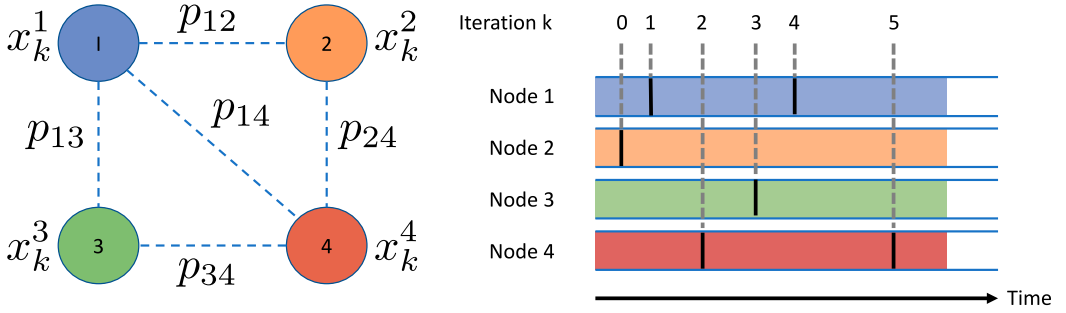


Figure 1. Network model and asynchronous computing. Left: The model of an example decentralized sensor network. Right: Asynchronous computing model.

An optimal solution of (1) should be consensual and optimised at the same time. Hence, two important measures must be considered for analysing our algorithm. The measures should be vs. iteration number k, which corresponds to the number of broadcast rounds in the entire network. The two metrics considered in this paper are described as follows.

- **Optimal value of objective function**:

$$\sum_{i=1}^{m} F_i(\bar{x}_k), \text{ where } \bar{x}_k = \frac{1}{m}\sum_{i=1}^{m} x_k^i.$$

  It determines whether the objective function of the averaged solution reaches the optimal objective value.
- **Disagreement among the nodes in the network**:

$$\sum_{i=1}^{m} \|x_k^i - \bar{x}_k\|^2.$$

  It measures how close every node in the network reaches the consensus solution.

Remark 1: We realise that a trade-off between communication cost and computation effort exists in the problem of decentralized learning. That is, for any decentralized algorithm, if we prefer to have less communication rounds to achieve the desirable result, the computational complexity in each update round should be higher. It is reasonable to see that if the update stage can be done more completely and then it should provide a better estimate, which can speed up the convergence of the network after broadcasting. The rationale of proposing communication-efficient algorithm for sensor networks is expressed as follows. First, the cost of communication is very high in practice (e.g. Ref. [21]). Especially in sensor networks, we are highly constrained by the physical bandwidth and energy consumption. It is known that the energy consumption of 1-bit radio communication is equal to that of 1 million computation operations. Moreover, the communication step is much more time-consuming than the computation step. Thus, reducing the total communication rounds is the key for speeding up decentralized learning in terms of total execution time.

To summarise, our proposed algorithm can be expressed in a formal way:

(1) **Initialization**: Each node is equipped with a local Poisson clock (rate 1), which is independent of other clocks. Set the weight factor θ.
(2) **Communication**: Assume the local clock of node i ticks (only one tick at the same time in the whole network). Node i broadcasts its estimate x^i to its neighbours. The neighbours receive the information with probability $p_{ij}, j \in N(i)$.

(3) **Update**: The neighbours who receive node i's broadcast will perform update as follows.

$$x^j \leftarrow \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2\alpha_j}\|x - (\theta x^i + (1-\theta)x^j)\|^2 + F_j(x) \right\};$$

α_j denotes the step size defined in previous. It is set to $1/\Gamma_k(i)$, where $\Gamma_k(i)$ is the number of updates performed at node i up to iteration k. Node i and the other nodes who do not receive node i's iterate keep unchanged.

(4) **Repeat**: The network stays silent until next local clock ticks and repeats steps 2–3.

Figure 1 illustrates the execution model on a decentralized sensor network and also the workflow of asynchronous computing. In this example, there are four nodes in the network and the dash line between a pair of two nodes implies that a communication link exists between the nodes (through broadcast communication) with connection probability p_{ij} (=p_{ji}) if (i,j) is a pair a neighbours. In the right figure, it shows that the local clock in node 2 first ticks and the global iteration number k increase by 1 to $k=1$. Node 2 will broadcast its current value to its neighbours and the neighbours who receive the value update their estimates according to the rule shown above. In the next, node 1 is activated (iteration number $k=2$) and do the same thing as node 2 in the previous round. The process will keep running in an asynchronous manner.

2.2. Algorithm interpretation

In below, we describe the differences and improvements of the proposed method over Nedic's [19]. Assuming nodes $i \in J_k$ receive the estimate $x_{k-1}^{i_k}$ (i_k is the index of the node selected at iteration k), Nedic's update rule can be expressed as follows.

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1-\theta)x_{k-1}^i \\ x_k^i &= y_k^i - \alpha_{i,k}\nabla F_i(y_k^i). \end{aligned} \tag{3}$$

To better analyse the algorithm, we equivalently convert (3) into its compact form:

$$\begin{aligned} x_k^i &= \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2\alpha_{i,k}}\|x - y_k^i\|^2 + \langle \nabla F_i(y_k^i), x\rangle \right\} \\ &= \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2\alpha_{i,k}}\|x - (\theta x_{k-1}^{i_k} + (1-\theta)x_{k-1}^i)\|^2 + \langle \nabla F_i(y_k^i), x\rangle \right\}. \end{aligned} \tag{4}$$

Comparing (4) with our update rule (2) in the proposed algorithm, we can see that the difference is on the second term within the argmin function. In fact, the inner product item in (4) is a linearization of $F_i(x)$ at point y_k^i. The effect is that in each round of update, Nedic's algorithm performs an approximation to the solution of the local minimization problem. Instead, our designed method solves the local optimization problem 'completely' in each update round. According to the trade-off (between computation complexity and communication cost) we observe (refer to Remark 1), our proposed algorithm is thus expected to outperform Nedic's in terms of communication cost and total execution time, which are considered to be the main metrics evaluating decentralized learning algorithms in sensor networks.

Remark 2: In local nodes, if gradient (sub-gradient)-based methods are used (like Nedic's), the convergence rate (in terms of communication rounds) would depend on the Lipschitz constant (while our proposed algorithm does not) which could be very large (larger Lipschitz constant in general implies slower convergence rate). Also in our proposed method, as the local estimate becomes more accurate with neighbours' information, the computation in each node would become much faster.

Remark 3: We claim that the proposed algorithm is suitable for the scenario that when the sensors may move randomly, in which case the neighbours of each sensor node may change randomly. In fact,

our algorithm is designed for sensor networks using broadcast communications. Each sensor node does not need to know its neighbour list. In each round, one node broadcasts its current estimate and the nodes who successfully receive the value will be considered as its neighbours. Thus, in the proposed algorithm and settings, the neighbours of each sensor node may change at each iteration and the network topology is indeed random along iterations.

3. Convergence analysis

In this section, we analyse the convergence behaviour of the proposed algorithm in terms of consensus and optimal solution measure. We follow the idea in [19] to derive the analysis of our developed scheme.

Before conducting the analysis, we can rewrite the proposed algorithm (2) in a form that similar to Nedic's method as follows.

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1-\theta) x_{k-1}^i \\ x_k^i &= y_k^i - \alpha_{i,k} \nabla F_i(x_k^i). \end{aligned} \tag{5}$$

Note that the second equation of (5) comes from the first-order optimality condition of Equation (2):

$$\frac{x_k^i - y_k^i}{\alpha_{i,k}} + \nabla F_i(x_k^i) = 0. \tag{6}$$

A compact form for all k and $i \in \mathcal{V}$ can be expressed as:

$$\begin{aligned} y_k^i &= \sum_{j=1}^{m} [W_k]_{i,j} x_{k-1}^j \\ x_k^i &= y_k^i - \left[y_k^j - \alpha_{i,k} \nabla F_i(x_k^i) \right] I(i \in J_k) - y_k^i I(i \in J_k) \end{aligned} \tag{7}$$

where $I(\cdot)$ is the indicator function and matrix W_k is defined as:

$$\begin{aligned} [W_k]_{i,i} &= 1-\theta \text{ for } i \in J_k, \ [W_k]_{i,i} = 1 \text{ otherwise} \\ [W_k]_{i,i_k} &= \theta \text{ for } i \in J_k, \ [W_k]_{i,j} = 0 \text{ otherwise.} \end{aligned} \tag{8}$$

In addition, three lemmas which would be used later are described here.

Lemma 3.1: *[[22]: Lemma 11] Assume σ_k, φ_k, ω_k, and ε_k are nonnegative random variables and assume the following hold*

$$\begin{aligned} &\mathbb{E}\left(\sigma_{k+1} | \Omega_k\right) \le \left(1+\omega_k\right)\sigma_k - \varphi_k + \varepsilon_k \ \text{almost surely}, \\ &\sum_{k=0}^{\infty} \omega_k < \infty \ \text{almost surely}, \ \sum_{k=0}^{\infty} \varepsilon_k < \infty \ \text{almost surely} \end{aligned}$$

where $\mathbb{E}\left(\sigma_{k+1} | \Omega_k\right)$ represents the conditional expectation given all the past history of σ_k, φ_k, ω_k, and ε_k up to iteration k. Then it concludes that

$$\sigma_k \to \sigma \ \text{almost surely}, \ \sum_{k=0}^{\infty} \varphi_k < \infty \ \text{almost surely}$$

where $\sigma \ge 0$ is some random variable.

Lemma 3.2: *[[19]: Lemma 2] The random matrix $W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k$ is independent and has identical distribution with each other.*

$$\mu := \mu\left(\mathbb{E}\left[\left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k\right)^T \left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k\right)\right]\right) < 1 \tag{9}$$

where $\mu(\mathbf{A})$ denotes the largest eigenvalue of a symmetric matrix $\mathbf{A}$.

Lemma 3.3: *[[19]] The upperbounds of step size $\alpha_{i,k}$ are obtained as follows when k is large enough ($k > \tilde{k}(m,q)$)*

$$\alpha_{i,k} \leq \frac{2}{k\delta_i}, \quad \alpha_{i,k}^2 \leq \frac{4m^2}{k^2 p_*^2}, \quad \left|\alpha_{i,k} - 1/k\delta_i\right| \leq \frac{2}{k^{\frac{3}{2}-q} p_*^2} \tag{10}$$

where δ_i is the total probability that node i updates. p_* denotes the minimum among all p_{ij}'s. $q \in \left(0, \frac{1}{2}\right)$ is some constant. $\tilde{k}(m,q)$ is an integer determined by the number of nodes m and q.

The main convergence results of our proposed decentralized learning algorithm are presented in the following two theorems. The proofs are provided in Appendices 1 and 2, respectively.

Theorem 3.4: *Let $\left\{x_k^i\right\}, \forall i \in \mathcal{V}, k \geq 0$ be the sequence generated by the algorithm in (2) and given that all the assumptions are satisfied. Then we can have:*

$$\sum_{k=1}^{\infty} \frac{1}{k} \|x_{k-1}^i - \bar{x}_{k-1}\| < \infty, \text{ and } \lim_{k\to\infty} \|x_k^i - \bar{x}_k\| = 0 \text{ almost surely}. \tag{11}$$

Note that Theorem 3.4 implies that the disagreement between each node's solution to the average of all the nodes' converges to zero almost surely. It indicates that all the nodes' solutions reach to consensus almost surely, which guarantees that all the nodes (learning individually) converge to a single model.

Theorem 3.5: *Let $\left\{x_k^i\right\}, \forall i \in \mathcal{V}, k \geq 0$ be the sequence generated by the algorithm in (2) and given that all the assumptions are satisfied. Then the sequences converge to a same optimal point almost surely for any node i.*

Theorem 3.5 shows the convergence of each node's solution x_k^i to the same optimal point. It implies that the consensual model learned by all the nodes is optimal with respect to problem in (1).

Notice that the two theorems above correspond to the measures of optimal value of objective function and node consensus defined in Section 2.1, respectively.

4. Experiment results

In this section, we investigate the performance of the proposed learning algorithm with two typical objective functions: regularized least-squares and logistic regression. Sensor networks are generated randomly with certain average node degrees (with 100 nodes in total). We exam the quantitative performance of the decentralized algorithms in terms of average objective value and node consensus described in section 2.1. Considering that Nedic's algorithm is the only work having the identical practical settings as ours, we compare our proposed method with Nedic's algorithm as a benchmark.

In decentralized regularized least-squares, the private objective function of node i is $F_i(x) = \frac{1}{2}\|\mathbf{A}_i x - \mathbf{b}_i\|_2^2 + \lambda_i \|x\|_2^2$, where the regularisation parameter λ_i is set to $1/100$, $\mathbf{A}_i$ and $\mathbf{b}_i$ (same dimension for each i) are data points available in node i. In this scenario, the size of $\mathbf{A}_i$ is $400 \times 32,768$ and the dimension of $\mathbf{b}_i$ is set accordingly.

In decentralized logistic regression, the local objective function (of node i) F_i is set to $F_i(x) = \sum_{j=1}^{p_i} \left(\log\left[1 + \exp\left(\left(d_i^j\right)^T x\right)\right] - b_i^j \left(d_i^j\right)^T x\right)$ where $p_i = 5, n = 200$, $\left(d_i^j\right)^T$ represents j-th row of $\mathbf{A}_i$ and b_i^j is the j-th entry of $\mathbf{b}_i$. We generate $\mathbf{A}_i \in \mathbb{R}^{p_i \times n}, \forall i$ randomly except the first columns are set to 1. Binary vector $\mathbf{b}_i \in \mathbb{R}^{p_i}$ is generated randomly.

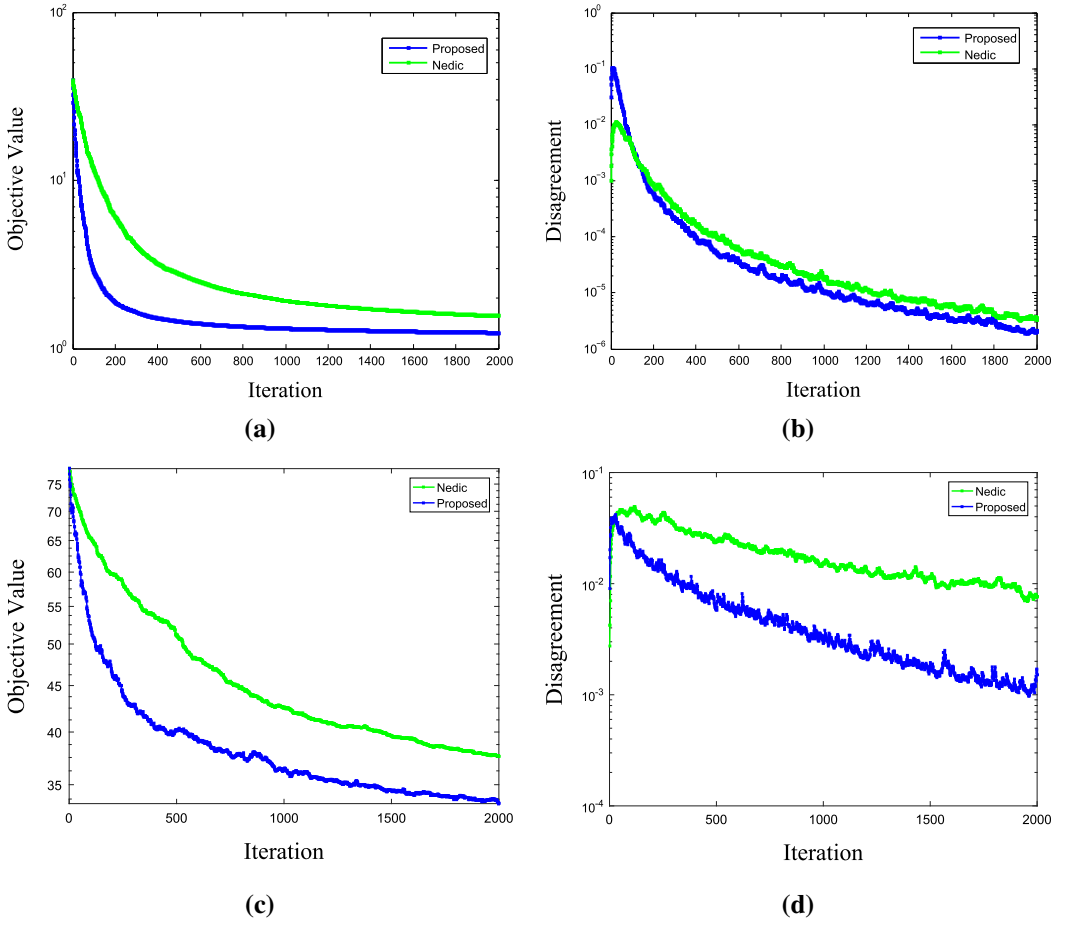


Figure 2. Comparison of convergence speed. Application in decentralized regularized least-squares (a-b). Decentralized logistic regression problem (c-d).

4.1. Comparison of convergence speed

We conduct the experiment using a randomly generated communication network, such that each senor is assumed to have 10 neighbours on average within its communication range. From Figure 2 we can see that the proposed algorithm (blue) converges faster than Nedic's (green) in terms of objective function value and consensus measure. This is consistent with our anlaysis before. We observe that the improvement of proposed algorithm over the benchmark is more significant in the plot of the average objective value. In fact, this effect is favored in system-level design since in realistic scenario, we might take the average of all the available solutions (from working nodes) rather than randomly select one node. Thus the performance in term of average objective value is more important than the consensus measure for real system. Notice that the *y*-axis in the figures are all in log-scale, hence the difference between the two methods is in fact much larger than the difference seen in normal linear scale.

4.2. Performance under unreliable links

In Figure 3, we study the performance of the proposed algorithm under link failures. Reliable link ($p_{ij} = 1$), less reliable link with connection probability $p_{ij} = 0.8$ and 0.6 are considered, respectively. Notice that, the convergence of the proposed algorithm is guaranteed. When the connection probability is lower, it is equivalent to the scenario that some nodes may have fewer number of neighbours at some

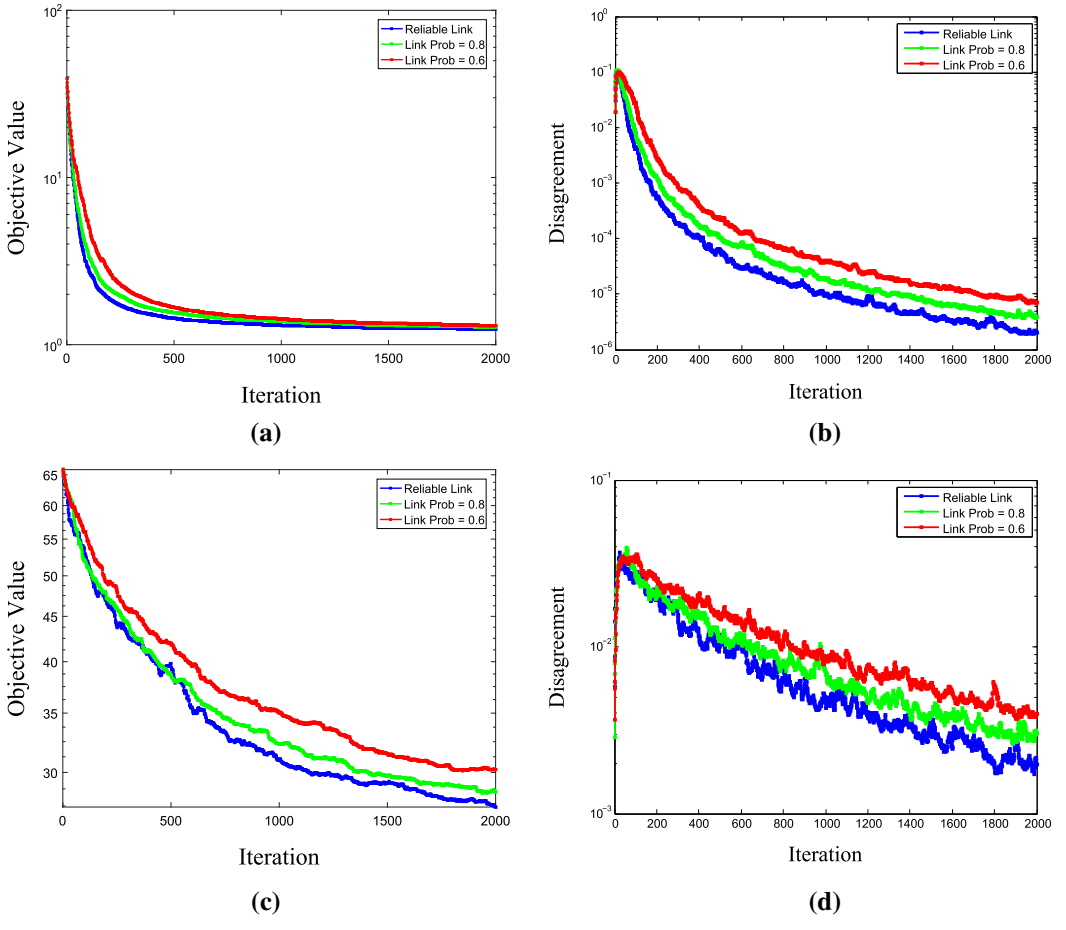


Figure 3. Effect of link failure in convergence behaviour. Application in decentralized regularized least-squares (a-b). Decentralized logistic regression problem (c-d).

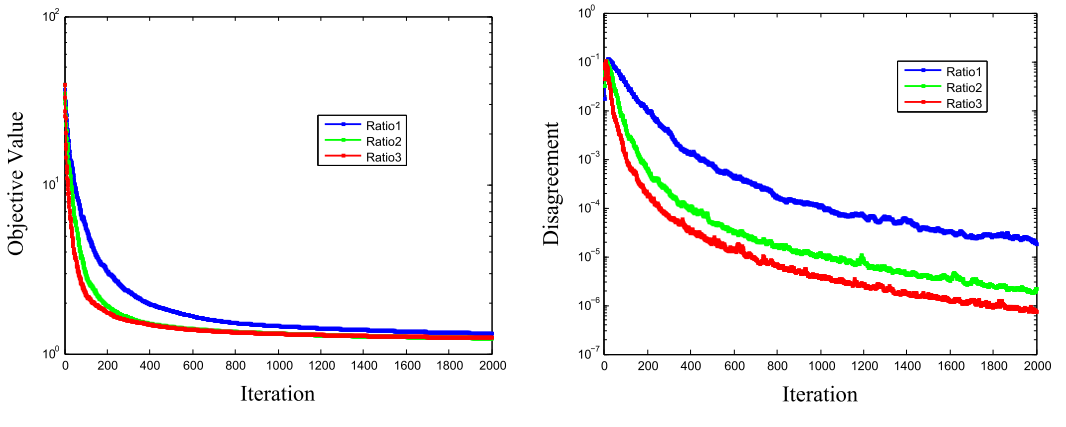


Figure 4. Influence of network connectivity ratio. Ratio1 (blue): average node degree = 5, Ratio2 (green): average node degree = 10, Ratio3 (red): average node degree = 15. Total number of nodes in the network is 100.

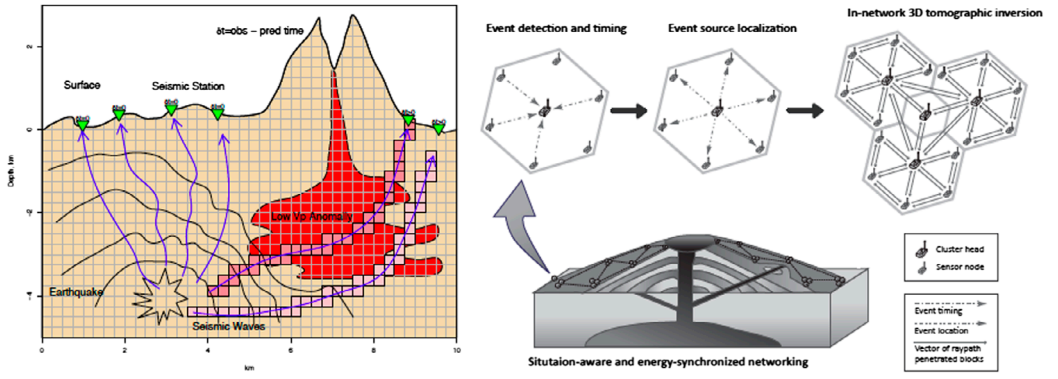


Figure 5. Illustration of seismic tomography and the new decentralized sensor network. Left: Principle of travel-time seismic tomography. Right: Real-time decentralized volcano tomography.

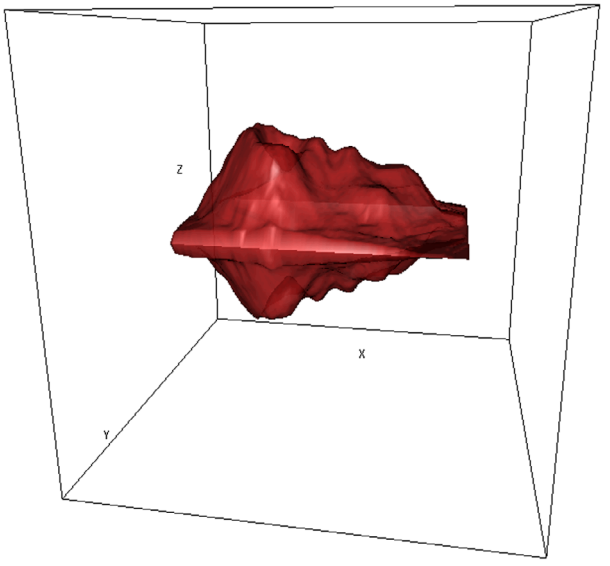
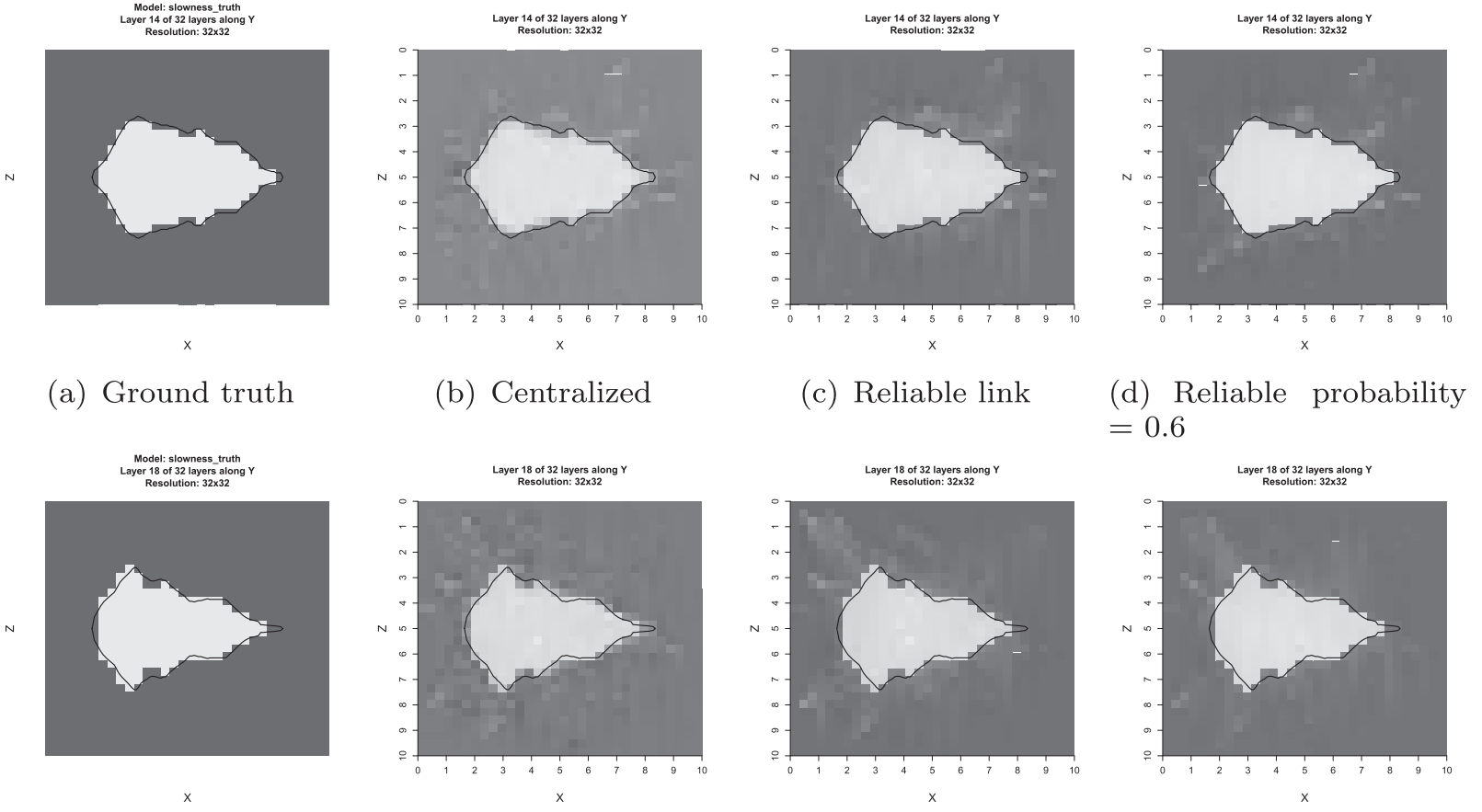


Figure 6. Ground truth of the magma model.

iterations. Hence, it can cause slower convergence since fewer number of nodes will get updated. As long as the network is connected and the connection probability is greater than zero (in Assumption 1), the information of each node will eventually be distributed to all the nodes in the network. Thus, all the nodes can reach consensus eventually and the convergence can be obtained. It is clear that the convergence is faster if the link is more reliable since in each communication round, more nodes receive the broadcast and perform updates. This speeds up the convergence of the network as a whole. More interestingly, the curves with link failures are kind of 'close' to the one with perfect link especially in the plot of the average objective value. This demonstrates the robustness of the developed method under unreliable links.

4.3. Influence of network connectivity ratio

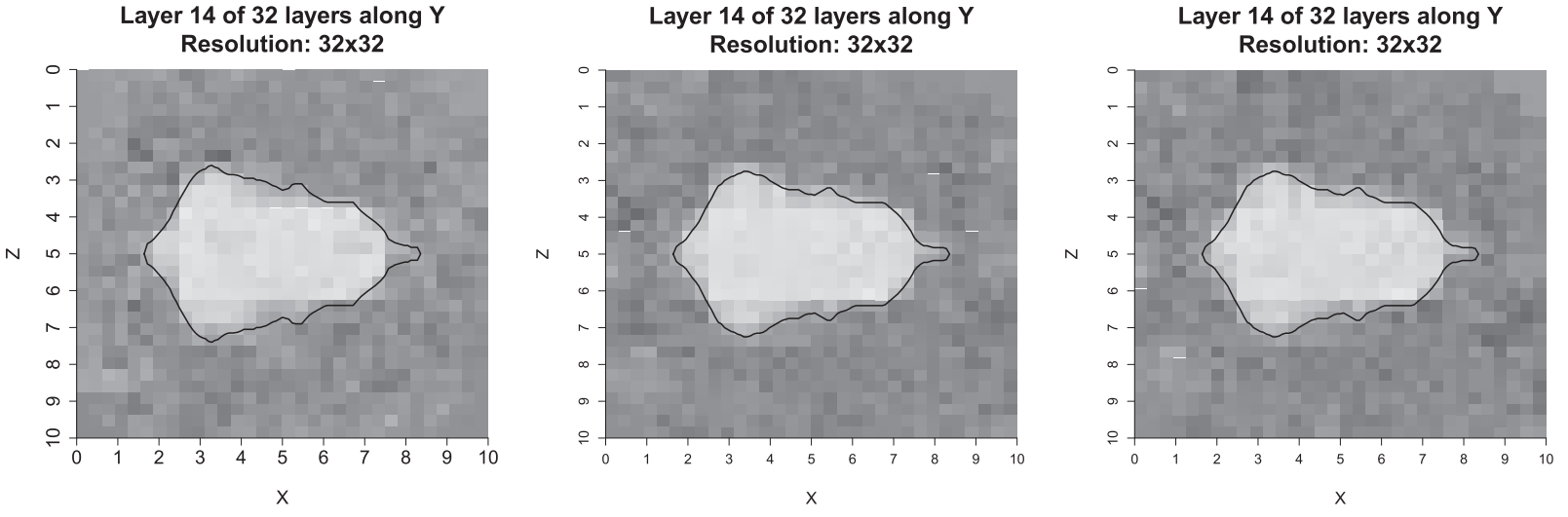
Now we investigate the effect of changing network connectivity ratio, which is defined as the number of edges divided by the number of all possible ones. Without loss of generality, here we adopt the regularized least-square objective function. Note that in our proposed decentralized algorithm, we



(a) Ground truth (b) Centralized (c) Reliable link (d) Reliable probability = 0.6

(e) Ground truth (f) Centralized (g) Reliable link (h) Reliable probability = 0.6

Figure 7. Seismic Tomography comparison at Layer 14 (upper row) and Layer 18 (bottom row). (a) and (e) are the ground truth. Centralized solution in (b) and (f) are obtained using centralized solver to the centralized formulation. (c) and (g), (d) and (h) are the tomography results of decentralized solutions (at 10th iteration) with reliable link probability =1 and 0.6, respectively.



(a) No packet loss (b) 10% packet loss (c) 30% packet loss

Figure 8. Application of seismic imaging – Effect of packet loss (vertical slice at layer 14).

are supposed to obtain faster convergence speed when each node has higher number of neighbours. Higher neighbour count means more nodes would receive the information after one broadcast of certain node. It accelerates information diffusion among all the nodes in the network and thus help all the nodes reach consensus faster. From Figure 4 we find that the higher the connectivity ratio is, the faster the convergence is in terms of average objective value and the consensus. Note that Ratio2 (node degree = 10) is our default setting and it shows close performance to Ratio3 (node degree = 15). We conclude that the developed scheme still works in very sparse networks, which can be leveraged to cover large-scale region in sensor network applications. Note that the experiments on the influence of network connectivity ratio are conducted as empirical study of the proposed algorithm. We will analyse the theoretical justification as our future work.

4.4. Additional results for the application of seismic imaging

We also test our proposed algorithm in seismic imaging, which is an important application of sensor networks [11,23,24]. Figure 5 shows the architecture of seismic tomography in sensor networks. Note that traditional seismic tomography problem can be formulated as regularized least-square problem, thus we adopt the same settings of decentralized regularized least-square problem as before. We use a 3-D synthetic model with resolution $32 \times 32 \times 32$. The monitored area contains a magma chamber (low velocity area) in a 10 km^3 cube. The number of sensor nodes is set to 100 and they are randomly distributed on top of the region. 400 events are generated and we compute the travel times from every event to each node based on the ground truth, and send the event location and travel time to corresponding node. To simulate the event location estimation and ray tracing errors, a white Gaussian noise is added to the travel time to construct the sensor node observations (arrival times). Details on the steps of seismic tomography can be found in [25]. The ground truth of the magma model with resolution $128 \times 128 \times 128$ (highest resolution for illustration) is depicted in Figure 6.

In addition to the convergence behaviour investigated under unreliable links, we illustrate the tomography results with this effect considering the application of seismic imaging in sensor networks (see Figure 7).

We also evaluate the robustness of our proposed algorithm from another perspective - packet loss. We simulate packet loss by setting partial vector of the broadcast to zero. We test with packet loss ratios 10 and 30%, respectively. The results at 50 iterations are shown in Figure 8. It is clear that the distinction between the result without packet loss is relatively small even at the case of 30% packet loss ratio. This validates the fault-tolerances and robustness of our proposed method in the applications frequently suffering from severe packet loss, such as (wireless) sensor networks-based seismic tomography. Similar as the previous subsection, the theoretical analysis of the effect of packet loss to the proposed algorithm is also an interesting direction to explore in the future.

5. Conclusion

This work investigated the decentralized consensus optimization problem, where the objective function is the sum of a set of convex and differentiable local functions. We proposed an asynchronous and broadcast-based algorithm, which does not require any synchronisation among the sensor nodes. The developed algorithm is then leveraged to solve big data computing problem in sensor networks. We conducted extensive tests of proposed algorithm on various aspects. The experiment results show that our designed method outperforms the benchmark.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was partially supported by National Science Foundation [grant number 1066391], [grant number 1442630], and [grant number 1125165].

References

[1] Matei I, Baras J. Performance evaluation of the consensus-based distributed subgradient method under random communication topologies. IEEE J Sel Topics Signal Proces. 2011;5:754–771.
[2] Nedic A, Ozdaglar A. Distributed subgradient methods for multi-agent optimization. IEEE Trans Autom Control. 2009;54:48–61.
[3] Nedic A, Olshevsky A. Distributed optimization over time-varying directed graphs. 2013 IEEE 52nd Annual Conference on Decision and Control (CDC). Firenze, Italy; Dec 2013. p. 6855–6860.
[4] Nedic A, Olshevsky A. Stochastic gradient-push for strongly convex functions on time-varying directed graphs. 2014. arXiv:1406.2075.
[5] Dusan Jakovetic Joao Xavier JMFM. Fast distributed gradient methods. 2014. arXiv:1112.2972v4.
[6] Zargham M, Ribeiro A, Jadbabaie A. A distributed line search for network optimization. American Control Conference (ACC), 2012. Montreal, Canada; June 2012. p. 472–477.
[7] Xiao L, Boyd S, Lall S. A scheme for Robust distributed sensor fusion based on average consensus. Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, IPSN '05, Los Angeles, California. 2005.
[8] Terelius UTH, Murray R. Decentralized multi-agent optimization via dual decomposition. IFAC. 2011;18:11245–11251.
[9] Shi G, Johansson KH. Finite-time and asymptotic convergence of distributed averaging and maximizing algorithms. 2012. arXiv:1205.1733.
[10] Shi W, Ling Q, Wu G, et al. Extra: An exact first-order algorithm for decentralized consensus optimization. 2014. arXiv:1404.6264.
[11] Zhao L, Song WZ, Ye X. Fast decentralized gradient descent method and applications to in-situ seismic tomography. 2015 IEEE International Conference on Big Data (Big Data). Santa Clara, CA; Oct 2015. p. 908–917.
[12] Wei E, Ozdaglar A. On the o(1/k) convergence of asynchronous distributed alternating direction method of multipliers. 2013. arXiv:1307.8254
[13] Iutzeler CF, Bianchi P, Hachem W. Asynchronous distributed optimization using a randomized alternating direction method of multipliers. 2013. arXiv:1303.2837
[14] Tsitsiklis J, Bertsekas D, Athans M. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. IEEE Trans Autom Control. 1986;31:803–812.
[15] Aysal TC, Yildiz ME, Sarwate AD, et al. Broadcast gossip algorithms for consensus. IEEE Trans Signal Proces. 2009;57:2748–2761.
[16] Kempe D, Dobra A, Gehrke J. Gossip-based computation of aggregate information. Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, FOCS '03. Cambridge, MA; 2003. p. 482–491.
[17] Iutzeler F, Ciblat P, Hachem W, et al. New broadcast based distributed averaging algorithm over wireless sensor networks. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Kyoto, Japan; March 2012. p. 3117–3120
[18] Ustebay D, Coates M, Rabbat M. Greedy gossip with eavesdropping. 3rd International Symposium on Wireless Pervasive Computing, 2008. ISWPC 2008. Santorini, Greece; May 2008. p. 759–763.
[19] Nedic A. Asynchronous broadcast-based convex optimization over a network. IEEE Trans Autom Control. 2011;56:1337–1351.
[20] Boyd S, Ghosh A, Prabhakar B, et al. Randomized gossip algorithms. IEEE/ACM Trans Networks. 2006;14:2508–2530.
[21] Bekkerman R, Bilenko M, Langford J. Scaling up machine learning: parallel and distributed approaches. Proceedings of the 17th ACM SIGKDD International Conference Tutorials, KDD '11 Tutorials, San Diego, California; 2011.
[22] Polyak BT. Introduction to optimization. New York (NY): Optimization Software Inc.; 1987.
[23] Lees JM. Seismic tomography of magmatic systems. J Volcanol Geotherm Res. 2007;167:37–56.
[24] Iyer HM, Dawson PB. Imaging volcanoes using teleseismic tomography. London: Chapman and Hall; 1993.
[25] Bording RP, Gersztenkorn A, Lines LR, et al. Applications of seismic travel-time tomography. Geophys J R Astron Soc. 1987;90:285–303.

Appendix 1. Proof of Theorem 3.4

***Proof*:** We define s_k^t to be the vector with components $\left[x_k^i\right]_t, \forall i \in \mathcal{V}$, where $\left[x_k^i\right]_t$ is the t-th element of node i's estimate at iteration k. Using $x_k^i = y_k^i - \left[y_k^i - \alpha_{i,k}\nabla F_i(x_k^i)\right] I(i \in J_k) - y_k^i I(i \in J_k)$ we can have:

$$s_k^t = W_k s_{k-1}^t + d_k^t. \tag{A1}$$

Here d_k^t is defined as a vector with

$$\left[d_k^t\right]_i = \left[-\alpha_{i,k}\nabla F_i(x_k^i)\right]_t \text{ for } i \in J_k, \ \left[d_k^t\right]_i = 0 \text{ otherwise.} \tag{A2}$$

From the definition of s_k^t, it can be shown that:

$$\left[\bar{x}_k\right]_t = \frac{1}{m}\mathbf{1}^T s_k^t \tag{A3}$$

where $\mathbf{1} = [1, 1, \cdots, 1]^T$ denotes a vector containing all 1's. In addition, plugging (A3) into (A1) yields:

$$\left[\bar{x}_k\right]_t = \frac{1}{m}\left(\mathbf{1}^T W_k s_{k-1}^t + \mathbf{1}^T d_k^t\right). \tag{A4}$$

Combing (A1) and (A4) derives the following.

$$s_k^t - \left[\bar{x}_k\right]_t \mathbf{1} = \left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k\right) s_{k-1}^t + \left(\mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^T\right) d_k^t \tag{A5}$$

where $\mathbf{I}$ represents the identity matrix. Based on the definition and leveraging the stochasticity of matrix W_k, it can be shown that $W_k\mathbf{1} = \mathbf{1}$. Hence, we can have:

$$\left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k\right)\left[\bar{x}_{k-1}\right]_t \mathbf{1} = 0. \tag{A6}$$

Adding the right part of (A6) into both sides of (A4) yields:

$$s_k^t - \left[\bar{x}_k\right]_t \mathbf{1} = \left(W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k\right)\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right) + \left(\mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^T\right) d_k^t \tag{A7}$$

To simplify the notation, we define

$$Q_k = W_k - \frac{1}{m}\mathbf{1}\mathbf{1}^T W_k, \quad U = \mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^T$$

The next step is to take the norm and conditional expectation on both sides of equation (A7):

$$\begin{aligned}&\mathbb{E}\left[\left\|s_k^t - \left[\bar{x}_k\right]_t \mathbf{1}\right\| \,|\Omega_{k-1}\right] \\ &\leq \mathbb{E}\left[\left\|Q_k\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right)\right\| \,|\Omega_{k-1}\right] + \mathbb{E}\left[\left\|U d_k^t\right\| \,|\Omega_{k-1}\right]\end{aligned} \tag{A8}$$

where Ω_k is the σ-algebra containing the past history up to iteration k, i.e.

$$\Omega_k = \left\{x_0^i, i_t, j_t, \forall i \in \mathcal{V}, t = 0, 1, \cdots k\right\}. \tag{A9}$$

By using Lemma 3.2 (μ is the largest eigenvalue mentioned there), we can upper bounding the first term in the right-hand side of (A8) as follows.

$$\mathbb{E}\left[\left\|Q_k\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right)\right\|^2 \,|\Omega_{k-1}\right] \leq \mu\left\|s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right\|^2. \tag{A10}$$

Based on the property that $\mathbb{E}\left[\|z\|\right] \leq \sqrt{\mathbb{E}\left[\|z\|^2\right]}$, (A10) can be transformed to:

$$\mathbb{E}\left[\left\|Q_k\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right)\right\| \,|\Omega_{k-1}\right] \leq \sqrt{\mu}\left\|s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right\|. \tag{A11}$$

The remaining part is that we need to upper bound the second item in the right hand side of (A8). We find that U is a projection matrix since:

$$U\mathbf{1} = \left(\mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^T\right)\mathbf{1} = \mathbf{0}. \tag{A12}$$

Then the norm of matrix U is 1. In consequence, we can obtain the following (also from the definition of d_k^t in (A2)):

$$\left\|Ud_k^t\right\|^2 \leq \left\|d_k^t\right\|^2 \leq \sum_{i\in J_k}\left\|\alpha_{i,k}\nabla F_i(x_k^i)\right\|^2. \tag{A13}$$

The bound of $\alpha_{i,k}$ is shown in Lemma 3.3, thus the right hand side of (A13) can be further bounded as follows.

$$\left\|Ud_k^t\right\|^2 \leq \sum_{i\in J_k}\alpha_{i,k}^2\left\|\nabla F_i(x_k^i)\right\|^2 \leq \frac{4m^2}{k^2p_*^2}\sum_{i\in J_k}\left\|\nabla F_i(x_k^i)\right\|^2. \tag{A14}$$

Now take conditional expectation on both sides of (A14), together with the assumed bounded gradient condition, also the cardinality of J_k is no more than m, we obtain

$$\mathbb{E}\left[\left\|Ud_k^t\right\|^2|\Omega_{k-1}\right] \leq \frac{4m^3}{k^2p_*^2}G^2. \tag{A15}$$

Using the inequality (converting (A10) to (A11)) again yields

$$\mathbb{E}\left[\left\|Ud_k^t\right\||\Omega_{k-1}\right] \leq \frac{2m\sqrt{m}}{kp_*}G. \tag{A16}$$

At this point, we have all the bounds of the terms in the right hand side of (A8). Plugging (A11) and (A16) into (A8) implies

$$\mathbb{E}\left[\left\|s_k^t-\left[\bar{x}_k\right]_t\mathbf{1}\right\||\Omega_{k-1}\right] \leq \sqrt{\mu}\left\|s_{k-1}^t-\left[\bar{x}_{k-1}\right]_t\mathbf{1}\right\|+\frac{2m\sqrt{m}}{kp_*}G. \tag{A17}$$

Since $\frac{1}{k-1} > \frac{1}{k}$, it can be shown that

$$\begin{aligned}&\frac{1}{k}\mathbb{E}\left[\left\|s_k^t-\left[\bar{x}_k\right]_t\mathbf{1}\right\||\Omega_{k-1}\right]\\ &\leq \frac{1}{k-1}\left\|s_{k-1}^t-\left[\bar{x}_{k-1}\right]_t\mathbf{1}\right\|-\frac{1-\sqrt{\mu}}{k}\left\|s_{k-1}^t-\left[\bar{x}_{k-1}\right]_t\mathbf{1}\right\|+\frac{2m\sqrt{m}}{k^2p_*}G.\end{aligned} \tag{A18}$$

Using the second claim of Lemma 3.1, we can see the following almost surely for any t.

$$\sum_{k=1}^{\infty}\frac{1}{k}\left\|s_{k-1}^t-\left[\bar{x}_{k-1}\right]_t\mathbf{1}\right\|<\infty. \tag{A19}$$

In addition, based on the definition of s_k^t, for any node i, it follows that

$$\sum_{k=1}^{\infty}\frac{1}{k}\left\|x_{k-1}^i-\bar{x}_{k-1}\mathbf{1}\right\|<\infty. \tag{A20}$$

Thus the first part of Theorem 3.4 is proved. For the second claim, we need to first show the following almost surely.

$$\lim_{k\to\infty}\left\|s_k^t-\left[\bar{x}_k\right]_t\mathbf{1}\right\|=0. \tag{A21}$$

Equation (A19) implies that

$$\liminf_{k\to\infty}\left\|s_k^t-\left[\bar{x}_k\right]_t\mathbf{1}\right\|=0. \tag{A22}$$

In order to show (A21), then we have to prove that the convergence of $\left\|s_k^t-\left[\bar{x}_k\right]_t\mathbf{1}\right\|$ when $k\to\infty$.

We will use the first claim of Lemma 3.1 to prove this. First take the square norm and conditional expectation on both sides of (A7). Further derivations are shown as follows.

$$\begin{aligned}
&\mathbb{E}\left[\left\|s_k^t - \left[\bar{x}_k\right]_t \mathbf{1}\right\|^2 |\Omega_{k-1}\right] \\
&\leq \mathbb{E}\left[\left\|Q_k\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right)\right\|^2 |\Omega_{k-1}\right] + \mathbb{E}\left[\left\|Ud_k^t\right\|^2 |\Omega_{k-1}\right] \\
&\quad + 2\sqrt{\mathbb{E}\left[\left\|Q_k\left(s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right)\right\|^2 |\Omega_{k-1}\right]} \times \sqrt{E\left[\left\|Ud_k^t\right\|^2 |\Omega_{k-1}\right]}.
\end{aligned} \tag{A23}$$

Plugging (A10), (A11) and (A15)–(A16) into (A23) yields

$$\begin{aligned}
&\mathbb{E}\left[\left\|s_k^t - \left[\bar{x}_k\right]_t \mathbf{1}\right\|^2 |\Omega_{k-1}\right] \\
&\leq \mu \left\|s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right\|^2 + \frac{4m^3}{k^2 p_*^2} G^2 + \sqrt{\mu}\left\|s_{k-1}^t - \left[\bar{x}_{k-1}\right]_t \mathbf{1}\right\| \frac{2m\sqrt{m}}{kp_*} G.
\end{aligned} \tag{A24}$$

Using (A19) and the first claim of Lemma 3.1, it is clear to see that $\left\|s_k^t - \left[\bar{x}_k\right]_t \mathbf{1}\right\|$ converges almost surely for any t. With this (A21) is proved. Leveraging the definition of s_k^t one more time, the almost sure convergence of the disagreement is verified (second conclusion). This completes the entire proof of Theorem 3.4. □

Appendix 2. Proof of Theorem 3.5

***Proof*:** We start by looking at (7) with $i \in J_k$. Subtracting x (some point in the feasible set) on both sides of (7) and taking square norm yields the following.

$$\left\|x_k^i - x\right\|^2 \leq \left\|y_k^i - x\right\|^2 + \alpha_{i,k}^2 \left\|\nabla F_i(x_k^i)\right\|^2 - 2\alpha_{i,k}\left(\nabla F_i(x_k^i)\right)^T \left(y_k^i - x\right). \tag{B1}$$

By using the equality $\alpha_{i,k} = \left(\alpha_{i,k} - \frac{1}{k\delta_i}\right) + \frac{1}{k\delta_i}$ and the inequality in Lemma 3.3, we can bound the right-most term in (B1) as follows.

$$\begin{aligned}
\left\|x_k^i - x\right\|^2 \leq \left\|y_k^i - x\right\|^2 + \alpha_{i,k}^2 \left\|\nabla F_i(x_k^i)\right\|^2 - \frac{2}{k\delta_i}\left(\nabla F_i(x_k^i)\right)^T \left(y_k^i - x\right) \\
+ \frac{4}{k^{3/2-q} p_*^2} \left\|\left(\nabla F_i(x_k^i)\right)^T \left(y_k^i - x\right)\right\|.
\end{aligned} \tag{B2}$$

Based on the fact that $2a^T b \leq \|a\|^2 + \|b\|^2$, the inner product in the right hand side of the (B2) can be bounded as follows.

$$\begin{aligned}
\left\|x_k^i - x\right\|^2 \leq (1+\beta_k)\left\|y_k^i - x\right\|^2 - \frac{2}{k\delta_i}\left(\nabla F_i(x_k^i)\right)^T \left(y_k^i - x\right) \\
+ \left(\alpha_{i,k}^2 + \beta_k\right)\left\|\nabla F_i(x_k^i)\right\|^2
\end{aligned} \tag{B3}$$

where $\beta_k = \frac{2}{k^{3/2-q} p_*^2}$. By the convexity of function F_i and the bounded (sub)gradient condition, the following inequality holds for arbitrary a, b, and c [19].

$$\nabla F_i(a)^T (a - b) \geq F_i(c) - F_i(b) - G\|a - c\|. \tag{B4}$$

Plugging (B4) into (B3) with $a = x_k^i$, $b = x$, $c = \bar{x}_{k-1}$ we can have

$$\begin{aligned}
\left\|x_k^i - x\right\|^2 \leq (1+\beta_k)\left\|y_k^i - x\right\|^2 - \frac{2}{k\delta_i}\left(F_i(\bar{x}_{k-1}) - F_i(x)\right) \\
+ \frac{2G}{k\delta_i}\left\|y_k^i - \bar{x}_{k-1}\right\| + \frac{4G}{k\delta_i}\left\|x_{k-1}^i - \bar{x}_{k-1}\right\| + \tau_k \left\|\nabla F_i(x_k^i)\right\|^2
\end{aligned} \tag{B5}$$

where $\tau_k = \frac{4m^2}{k^2 p_*^2} + \beta_k$. To bound the term $\left\| y_k^i - \bar{x}_{k-1} \right\|$, we will use the property in [19] described in (B10). Now taking conditional expectation and the bounded (sub)gradient condition, it follows that

$$\mathbb{E}\left[\left\| x_k^i - x \right\|^2 | \Omega_{k-1}, i_k, J_k \right] \leq (1+\beta_k) \left\| y_k^i - x \right\|^2 - \frac{2}{k\delta_i} \left(F_i(\bar{x}_{k-1}) - F_i(x) \right) + \frac{2G}{k\delta_i} \left\| y_k^i - \bar{x}_{k-1} \right\| + \frac{4G}{k\delta_i} \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| + \tau_k G^2. \tag{B6}$$

Recall the definition of δ_i, it is the probability of node i updates (the event that it receives broadcast from any of its neighbours). Hence, the fact holds $\delta_i \geq \frac{p_*}{m}$. Then (B6) can be modified to

$$\mathbb{E}\left[\left\| x_k^i - x \right\|^2 | \Omega_{k-1}, i_k, J_k \right] \leq (1+\beta_k) \left\| y_k^i - x \right\|^2 - \frac{2}{k\delta_i} \left(F_i(\bar{x}_{k-1}) - F_i(x) \right) + \frac{2mG}{kp_*} \left\| y_k^i - \bar{x}_{k-1} \right\| + \frac{4mG}{kp_*} \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| + \tau_k G^2. \tag{B7}$$

Now let $x = x^*$ where x^* is an optimal point of the objective function. Substituting this into (B7) yields

$$\mathbb{E}\left[\left\| x_k^i - x^* \right\|^2 | \Omega_{k-1}, i_k, J_k \right] \leq (1+\beta_k) \left\| y_k^i - x^* \right\|^2 - \frac{2}{k\delta_i} \left(F_i(\bar{x}_{k-1}) - F_i(x^*) \right) + \frac{2mG}{kp_*} \left\| y_k^i - \bar{x}_{k-1} \right\| + \frac{4mG}{kp_*} \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| + \tau_k G^2. \tag{B8}$$

Incorporating the case when $i \notin J_k$ ($x_k^i = y_k^i$) with the current formula (which assumes $i \in J_k$), and also with the definition that δ_i denotes the total probability that node i updates, we obtain

$$\begin{aligned} \mathbb{E}\left[\left\| x_k^i - x^* \right\|^2 | \Omega_{k-1} \right] \leq (1+\beta_k) \mathbb{E}\left[\left\| y_k^i - x^* \right\|^2 | \Omega_{k-1} \right] \\ - \frac{2}{k} \left(F_i(\bar{x}_{k-1}) - F_i(x^*) \right) + \frac{2mG}{kp_*} \mathbb{E}\left[\left\| y_k^i - \bar{x}_{k-1} \right\| | \Omega_{k-1} \right] \\ + \frac{4mG}{kp_*} \mathbb{E}\left[\left\| x_{k-1}^i - \bar{x}_{k-1} \right\| | \Omega_{k-1} \right] + \delta_i \tau_k G^2. \end{aligned} \tag{B9}$$

It can be shown that for any x the following property holds [19].

$$\sum_{i=1}^{m} \mathbb{E}\left[\left\| y_k^i - x \right\| | \Omega_{k-1} \right] \leq \sum_{i=1}^{m} \left\| x_{k-1}^i - x \right\|. \tag{B10}$$

At this point, summing up both sides of (B9) over all the nodes $i \in \mathcal{V}$, applying (B10) and the definition in (1) yields

$$\sum_{i=1}^{m} \mathbb{E}\left[\left\| x_k^i - x^* \right\|^2 | \Omega_{k-1} \right] \leq (1+\beta_k) \sum_{i=1}^{m} \left\| x_{k-1}^i - x^* \right\|^2 - \frac{2}{k} \left(F(\bar{x}_{k-1}) - F(x^*) \right) + \frac{6mG}{kp_*} \sum_{i=1}^{m} \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| + \sum_{i=1}^{m} \delta_i \tau_k G^2. \tag{B11}$$

It can be seen that the summation of β_k over k (from 1 to ∞) is bounded. Furthermore, the last term in (B11) meets the condition in Lemma 3.1 due to the definition of τ_k. From the first claim of theorem 3.4, we can see that the following holds almost surely.

$$\sum_{k=1}^{\infty} \frac{6mG}{kp_*} \sum_{i=1}^{m} \left\| x_{k-1}^i - \bar{x}_{k-1} \right\| < \infty \tag{B12}$$

Considering the last two terms in (B11) as one item along with the fact that $F(\bar{x}_{k-1}) - F(x^*) \geq 0$, we can see that all the conditions of Lemma 3.1 have been satisfied. Hence, it concludes that the sequence $\left\{ \sum_{i=1}^{m} \left\| x_k^i - x^* \right\|^2 \right\}$ converges and

$$\sum_{k=\tilde{k}}^{\infty} \frac{1}{k} \left(F(\bar{x}_{k-1}) - F(x^*) \right) < \infty \tag{B13}$$

Similar as the proof in Theorem 3.4, it can be deducted from (B13) that

$$\lim_{k\to\infty} \inf F(\bar{x}_{k-1}) = F(x^*) \tag{B14}$$

Since the sequence $\left\{\sum_{i=1}^m \left\|x_k^i - x^*\right\|^2\right\}$ converges and (B14) holds for any point x^* in the set of optimal solutions X^*, we know that there exists a subsequence $\left\{\bar{x}_{k_j}\right\}$ (of sequence $\left\{\bar{x}_k\right\}$) such that $\bar{x}_{k_j} \to \hat{x}$ for some $\hat{x}$ in the feasible set X and $\lim_{j\to\infty} F(\bar{x}_{k_j}) = F(x^*)$. By using continuity of function F and the fact that $\bar{x}_{k_j}$ converges to $\hat{x}$, it follows that $\lim_{j\to\infty} F(\bar{x}_{k_j}) = F(\hat{x})$. Hence, we have $F(\hat{x}) = F(x^*)$, which means $\hat{x}$ belongs to the optimal solution set X^*. Now we can see that the sequence $\left\{\sum_{i=1}^m \left\|x_k^i - \hat{x}\right\|^2\right\}$ converges, together with the fact that $\left\{\sum_{i=1}^m \left\|x_k^i - \bar{x}_k\right\|^2\right\} \to 0$ as $k \to \infty$ (based on the second claim of theorem 3.4), we know $\left\|\bar{x}_k - \hat{x}\right\|^2$ converges. Since the subsequence $\left\|\bar{x}_{k_j} - \hat{x}\right\|^2 \to 0$, there is $\left\|\bar{x}_k - \hat{x}\right\| \to 0$, which shows that $\left\{\bar{x}_k\right\}$ converges to an optimal point ($\hat{x}$) of the problem. Finally, using the second claim of theorem 3.4 again, it can be obtained that the sequence $\left\{x_k^i\right\}$ generated by any node $i \in \mathcal{V}$ converges to the same optimal solution point almost surely. The proof of the theorem is thus complete. □