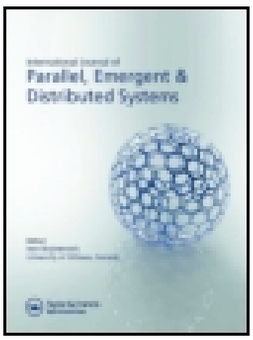


International Journal of Parallel, Emergent and Distributed Systems

ISSN: 1744-5760 (Print) 1744-5779 (Online) Journal homepage: http://www.tandfonline.com/loi/gpaa20

Decentralized consensus in distributed networks

Liang Zhao & WenZhan Song

To cite this article: Liang Zhao & WenZhan Song (2016): Decentralized consensus in distributed networks, International Journal of Parallel, Emergent and Distributed Systems, DOI: 10.1080/17445760.2016.1233552

To link to this article: https://doi.org/10.1080/17445760.2016.1233552

Published online: 28 Sep 2016.

INTERNATIONAL JOURNAL OF PARALLEL, EMERGENT AND DISTRIBUTED SYSTEMS, 2016
http://dx.doi.org/10.1080/17445760.2016.1233552

Taylor & Francis
Taylor & Francis Group

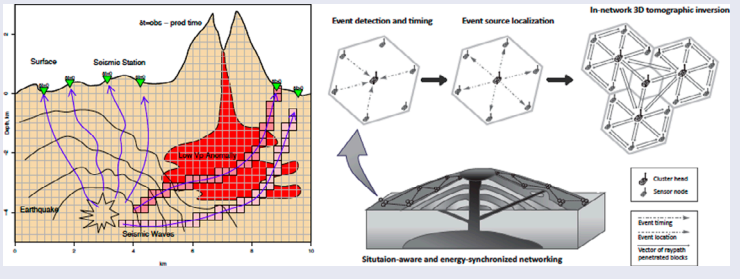
Decentralized consensus in distributed networks

Liang Zhao[a] and WenZhan Song[b]

[a]Department of Computer Science, Georgia State University, Atlanta, GA, USA; [b]College of Engineering, University of Georgia, Athens, GA, USA

ABSTRACT

We consider the decentralised consensus optimization problem in this paper. A fast decentralised gradient-based algorithm is proposed for a general class of problems with convex and differentiable objective functions. The developed method guarantees that all the nodes in the network would reach consensus on the global variable eventually. Based on the proposed decentralised algorithm, a broadcast-based protocol is designed, which is capable of providing solution for real-time *in situ* seismic imaging. Extensive numerical experiments on both synthetic and real sensor network seismic data sets validate the superior efficiency of the proposed algorithm over the other benchmarks.



It is the architecture of the seismic tomography in sensor networks.

ARTICLE HISTORY
Received 28 July 2016
Accepted 4 September 2016

KEYWORDS
Big data; decentralised computing; in-network processing; seismic tomography; sensor network

1. Introduction

In this paper, we study the decentralised consensus optimization problem such that a network of nodes collaboratively optimise an objective consisting of summation of local functions (hold in each node). The solution methodology can be utilised to a class of practical applications such as *in situ* seismic tomography in large-scale sensor networks.

The state-of-art of seismic imaging involves collecting the raw seismic data from sensors to data loggers then manually retrieving data for post processing which may take months to complete.[1] However, it is highly desirable to have the capability of imaging in real-time since the time scales of geological hazard mitigation can be tens of minutes. Recent sensor network technology has matured to the point where it is now possible to deploy and maintain large networks for real-time data acquisition.[2] However, existing tomography algorithms cannot be easily implemented in sensor networks because they rely on centralised algorithms and require transmission of huge amount of data over the network, which is infeasible due to severe bandwidth limitations. We are thus motivated

CONTACT Liang Zhao lzhao5@student.gsu.edu

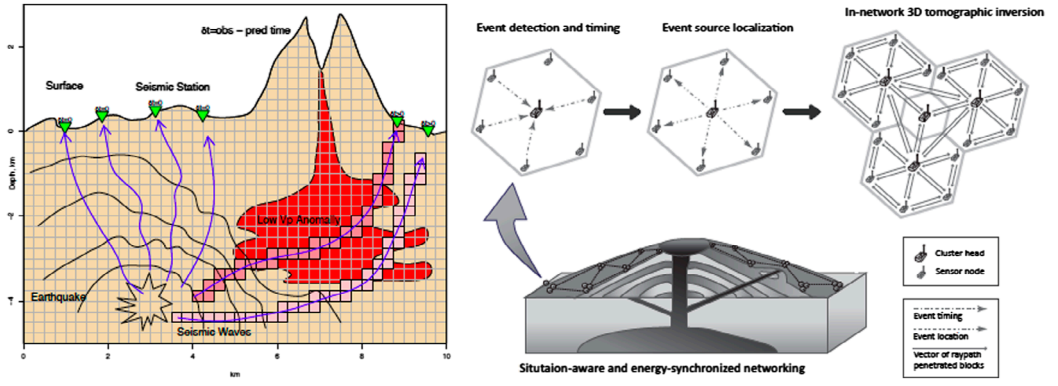


Figure 1. Illustration of seismic tomography and the new decentralised sensor network. Left: principle of travel-time seismic tomography. Right: real-time decentralised volcano tomography.

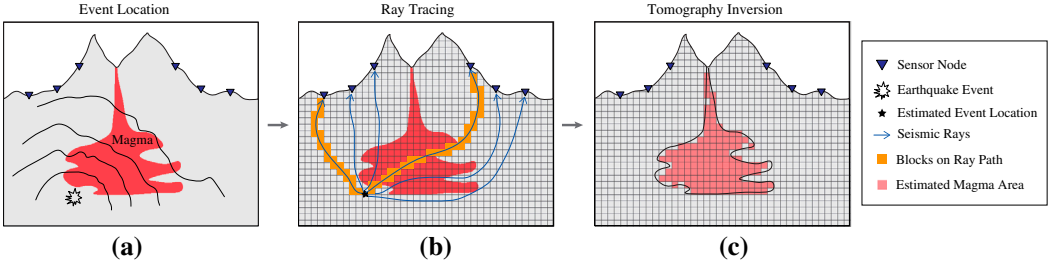


Figure 2. Procedure of seismic tomography.

to propose a paradigm-shifting decentralised framework for real-time *in situ* seismic tomography (Figure 1).

The principle employed for seismic tomography in this paper is travel-time seismic tomography, which consists of three steps (Figure 2).[3]

(1) *Event location.* After certain seismic event occurs, the nodes detect and then determine arrival times, which are used to estimate the source location and origin time (Figure 2(a)).
(2) *Ray tracing.* At each sensor node, ray tracing is performed to find the rays travel from the seismic event sources to the receivers (sensor nodes) based on the underlying reference velocity model. (Figure 2(b)).
(3) *Tomographic inversion.* The traced ray paths are used to perform a reconstruction of the velocity structure within the volcano (Figure 2(c)).

In this paper, we focus on the third step (tomographic inversion), which is the most computational-intensive and time-consuming aspect of seismic tomography (assuming the first two steps are already done).

In recent years, a great volume of research has been conducted in distributed convex optimization. It has huge potential in modelling and algorithmic design for decentralised big data computing problems, which aims to reduce the computation and communication burdens.[4] The problem in this paper has a general form and can be expressed as follows.

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

where p nodes are in the network and they need to collaboratively estimate the model parameters x. Each node i locally holds the function F_i and can communicate only with its immediate neighbours.

In the presenting work, we propose a decentralised computing scheme that every node or agent in the network estimates the global variable by using local private data and communicating with direct neighbours (neighbours within its radio communication range). The developed framework enjoys the features such as: computation and communication balanced, multi-hop communication free, fault-tolerant and privacy-preserving, etc. Furthermore, this kind of decentralised mechanism is scalable and capable of solving a huge class of big data computing problems.

This paper is organised as follows. In the remaining context of this section, we survey the related work, summarise our key contributions and introduce the notations and assumptions. In Section 2, we derive and interpret the design of the proposed fast decentralised consensus algorithm. Section 3 shows the main convergence results in the presenting work. In Sections 4 and 5, the proposed algorithm is applied to the problem of seismic tomography and we conduct extensive numerical tests on synthetic and real world seismic data sets to demonstrate the practical performance of the proposed algorithm. Section 6 concludes the paper.

1.1. Related work

Much attention has been paid to fully distributed (decentralised) consensus optimization problem, especially in applications like distributed machine learning, multi-agent optimization, etc. Several algorithms have been proposed for solving general convex and (sub)differentiable functions. Considering the problem in (1), (sub)gradient-based methods have been proposed.[5–10] However, it has been analysed that the aforementioned methods can only converge to a neighbourhood of an optimal solution in the case of fixed step size.[11] Modified algorithms have been developed in [9,10], which use diminishing step sizes in order to guarantee to converge to an true solution. Other related algorithms were discussed in [12–18], which share similar ideas. The D-NC algorithm proposed in [10] was demonstrated to have an an outer-loop convergence rate of $O(1/k^2)$ in terms of objective value error. The rate is same as the optimal centralised Nesterov's accelerated gradient method and decentralised algorithms usually have slower convergence rate than the centralised versions. However, the number of consensus iterations within outer-loop is growing significantly along the iteration. Shi et al. [19] developed a method based on correction on mixing matrix for DGD method [11] without diminishing step sizes.

The algorithms mentioned above are based on synchronous models. Distributed optimization methods for asynchronous models have been designed in [20–22]. However, it is worth noting that their convergence rates are usually slower than the counterparts in synchronous models.

1.2. Contributions

Our key contribution is three-fold. First, we develop a fast gradient-based method for solving the decentralised consensus optimization problem. Second, we provide a general framework, which can be used for algorithmic design in decentralised consensus problem. Third, we develop a protocol for the problem of seismic tomography in sensor networks based on the proposed decentralised algorithm. The protocol is demonstrated to be a promising approach for real-time *in situ* seismic imaging.

1.3. Notations and assumptions

Notation: Let $\mathbf{x} \in \mathbb{R}^{np\times 1} := \left[x_{(1)}^T, x_{(2)}^T, \ldots, x_{(p)}^T\right]^T$, where $x_{(i)}$ is a column vector containing local estimate of common interest x at node i. Similarly, define $\mathbf{F}(\mathbf{x}) = \sum_{i=1}^{p} F_i(x_i)$ and $\nabla\mathbf{F}(\mathbf{x}) \in \mathbb{R}^{np\times 1} := \left[\nabla F_1(x_{(1)})^T, \ldots, \nabla F_p(x_{(p)})^T\right]^T$, where $\nabla F_i(\,\cdot\,)$ denotes the gradient of function F_i. Symbol $\otimes$ denotes the kronecker product. $\mathbf{z}$, $\mathbf{y}$, $\mathbf{v}$ are auxiliary variables that have the same dimension and structure as $\mathbf{x}$.

Assumption 1.1: *Define the optimal value of* (1) *as* $F^* := \inf_{x\in\mathbb{R}^n} F(x)$ *and the solution set as* $X^* := \{x \in \mathbb{R}^n : F(x) = F^*\}$. *We assume* X^* *is non-empty.*

Assumption 1.2: *Every local objective function* $F_i, \forall i$ *is convex and smooth.*

Assumption 1.3: *The gradient of local objective function* $\nabla F_i, \forall i$ *is Lipschitz continuous with constant* $L_i > 0$: $\|\nabla F_i(x) - \nabla F_i(y)\| \leq L_i\|x - y\|$.

Assumption 1.4: *Assume the network is connected and we have some mixing matrix* $\mathbf{W}$ *such that* $\mathbf{W}$ *is symmetric stochastic matrix, where* $\mathbf{W}_{ij} \in (0, 1)$ *denotes the mixing parameter between node i and j if* (i, j) *are immediate neighbours.* $\mathbf{W}_{ij} = 0$ *if* (i, j) *are not neighbours.*

Assumption 1.5: *The gradient of local objective function* $\nabla F_i, \forall i$ *is bounded by some non-negative constant* $B \geq 0$, *i.e.* $\|\nabla F_i\| \leq B$.

Note that the assumptions adopted above are common for analysis of decentralised optimization problem.[10,11]

2. Algorithm design and interpretation

In this section, we first describe the proposed decentralised consensus algorithm and then we show the idea and interpretation of the algorithm design.

2.1. Algorithm design

The proposed iterative method is described in Algorithm 1 where k indexes the iteration round.

Algorithm 1: Fast Decentralised Consensus Algorithm (FDC)

1: Initialize $\mathbf{z}(-1) = 0$ and arbitrary $\mathbf{x}(-1) = \mathbf{x}(0) = \mathbf{v}(0)$. Construct an average mixing matrix $\mathbf{W}$. Let $\widetilde{\mathbf{W}} := \frac{1}{2}\left(\mathbf{I} + \mathbf{W}^\gamma\right)$ and $\widehat{\mathbf{W}} := \mathbf{W}^\gamma$, where $\gamma = \left\lceil \frac{\log 2}{-\log \mu} \right\rceil$. $\theta_j = \frac{2}{j+1}$ and initialize $j = 0$. Select scalar parameter $\kappa_{\min}$.
2: **for** $k = 1, 2, \ldots,$ **do**
3: if $j > \kappa_{\min}$ then $j = 1$; else $j = j + 1$.

$$\mathbf{z}(k-1) = \mathbf{z}(k-2) + \left((\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}) \otimes \mathbf{I}\right)\mathbf{x}(k-2) \tag{2a}$$

$$\mathbf{y}(k) = (1-\theta_j)\mathbf{x}(k-1) + \theta_j\mathbf{v}(k-1) \tag{2b}$$

$$\mathbf{x}(k) = (\widehat{\mathbf{W}} \otimes \mathbf{I})\left[\mathbf{y}(k) - \frac{1}{L}\nabla\mathbf{F}(\mathbf{y}(k))\right] - \mathbf{z}(k-1) \tag{2c}$$

$$\mathbf{v}(k) = \mathbf{x}(k-1) + \frac{1}{\theta_j}(\mathbf{x}(k) - \mathbf{x}(k-1)) \tag{2d}$$

4: **end for**
5: Output $\mathbf{x}(k)$.

In Algorithm 1, $L = \max\{L_i\}$ where L_i is the lipschitz constant of local objective F_i of node i. μ is the second largest singular value of mixing matrix $\mathbf{W}$. Detail selection of matrix $\mathbf{W}$ will be discussed later. The dimension of the identity matrix $\mathbf{I}$ equals to the size of decision variable $\mathbf{x}$ (i.e. $\mathbf{I} \in \mathbb{R}^{n\times n}$ in our setting).

2.2. Design of mixing matrix W

An important set-up is the design of the mixing matrix $\mathbf{W}$. Although Assumption 1.4 only requires $\mathbf{W}$ matrix to be symmetric stochastic, it could actually impact the performance of the algorithm.[23,24] There are several off-the-shelf designs for mixing matrix $\mathbf{W}$ in [9,11,15,23–25]. However, there is no general rule for determining the best $\mathbf{W}$ in decentralised consensus optimization problem. We adopt the Metropolis–Hastings matrix in [23,24] here since it is easy to implement and has good performance in general. It is described as follows.

$$\mathbf{W}_{ij} = \begin{cases} 1/\left(\max\{d_i, d_j\} + 1\right) & \text{if } j \in \mathcal{N}_i \text{ and } j \neq i \\ 1 - \sum 1/\left(\max\{d_i, d_j\} + 1\right), \forall j \in \mathcal{N}_i & \text{if } j = i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where d_i and d_j are the degrees of the neighbour sets $\mathcal{N}_i$ and $\mathcal{N}_j$, respectively.

2.3. Optimality condition of problem (1)

Before performing analysis on the proposed algorithm FDC, we provide the condition of the optimal solution of problem (1) in the following theorem.

Theorem 2.1: *If the sequence $\{\mathbf{x}(k)\}$ generated by FDC converges to a limiting point $\mathbf{x}^* := \lim_{k\to\infty}\mathbf{x}(k)$, then $\mathbf{x}^*$ is consensual such that $x^*_{(1)} = x^*_{(2)} = \cdots = x^*_{(p)}$ and $\{x^*_{(i)}, \forall i\}$ is an optimal solution of consensus optimization problem in (1).*

***Proof*:** We utilise the result (Proposition 2.1 in [19]) and prove the theorem by showing FDC algorithm satisfies the consensual and optimality conditions in [19]. Without loss of generality, we assume the dimension of decision variable $n = 1$. Now take the difference of $\mathbf{x}(k)$ and $\mathbf{x}(k-1)$ according to (2c) we can have:

$$\begin{aligned} \mathbf{x}(k) - \mathbf{x}(k-1) = \widehat{\mathbf{W}}\left[\mathbf{y}(k) - \frac{1}{L}\nabla\mathbf{F}(\mathbf{y}(k)) - \mathbf{y}(k-1) + \frac{1}{L}\nabla\mathbf{F}(\mathbf{y}(k-1))\right] \\ - \left(\mathbf{z}(k-1) - \mathbf{z}(k-2)\right) \end{aligned} \tag{4}$$

We assume for now that $\mathbf{x}(k)$ converges to $\mathbf{x}^*$ as $k \to \infty$. Based on (2a), (2b) and (2d), (4) is equivalent to:

$$\mathbf{x}^* - \mathbf{x}^* = \mathbf{0} - \widetilde{\mathbf{W}}\mathbf{x}^* = \mathbf{0} \tag{5}$$

which implies that $\mathbf{W}^\gamma\mathbf{x}^* = \mathbf{x}^*$. It is straightforward to see that $\mathbf{x}^*$ should be consensual. Next, when $k \to \infty$, (2c) becomes:

$$\mathbf{x}^* = \widehat{\mathbf{W}}\left[\mathbf{x}^* - \frac{1}{L}\nabla\mathbf{F}(\mathbf{x}^*)\right] - \sum_{t=0}^{\infty}\widetilde{\mathbf{W}}\mathbf{x}(t) \tag{6}$$

Since $\widehat{\mathbf{W}}\mathbf{x}^* = \mathbf{x}^*$, (6) implies that:

$$\widehat{\mathbf{W}}\nabla\mathbf{F}(\mathbf{x}^*) = -L\sum_{t=0}^{\infty}\widetilde{\mathbf{W}}\mathbf{x}(t) \tag{7}$$

Left-multiply $\mathbf{1}^T$ to both sides of (7) yielding (using the definitions of $\widehat{\mathbf{W}}$ and $\widetilde{\mathbf{W}}$):

$$\mathbf{1}^T\nabla\mathbf{F}(\mathbf{x}^*) = \mathbf{0} \tag{8}$$

This is indeed the optimality condition of the decentralised consensus problem. □

The above theorem indicates that if the sequence generated by FDC can converge to a limiting point, then it is assured that FDC can converge to the optimal solution of the decentralised consensus problem in (1). In the convergence analysis section later, we will demonstrate that FDC is proved to meet the condition and can eventually provide the optimal solution of (1).

2.4. Interpretation of FDC method (Algorithm 1)

In this section, we discuss the rational and derivation of our proposed algorithm design. To solve the problem in (1) in a decentralised manner, the consensual formulation can be expressed as:

$$\min_{\mathbf{x}\in\mathbb{R}^p} \mathbf{F}(\mathbf{x}) : \mathbf{W}\mathbf{x} = \mathbf{x} \tag{9}$$

The constraint requires $\mathbf{x}$ to be consensual due to the property of the mixing matrix W. Assume $\mathbf{W}$ is symmetric and $\mathbf{U}$ is symmetric such that $\mathbf{U}^2 = \mathbf{I} - \mathbf{W}$. Then $\mathbf{W}\mathbf{x} = \mathbf{x}$ if and only if $\mathbf{U}\mathbf{x} = \mathbf{0}$ holds. The problem in (9) is thus equivalent to:

$$\min_{\mathbf{x}\in\mathbb{R}^p} \mathbf{F}(\mathbf{x}) : \mathbf{U}\mathbf{x} = \mathbf{0} \tag{10}$$

Note that problem in (10) contains the exact consensus constraint. We first convert (10) into an equivalent unconstrained composite optimization problem as follows.

$$\min_{\mathbf{x}\in\mathbb{R}^p} \mathbf{F}(\mathbf{x}) + \mathbf{G}(\mathbf{x}) \tag{11}$$

where $\mathbf{G}(\mathbf{x})$ is an indicator function such that:

$$\mathbf{G}(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{U}\mathbf{x} = \mathbf{0}; \\ +\infty, & \text{else.} \end{cases} \tag{12}$$

If we apply the reformulated proximal-graident method (FISTA) in [26], it has the following procedures:

$$\mathbf{y}(k) = (1-\theta_k)\mathbf{x}(k-1) + \theta_k \mathbf{v}(k-1) \tag{13}$$

$$\mathbf{x}(k) = \arg\min_{\mathbf{s}} \left\{ \mathbf{G}(\mathbf{s}) + \frac{1}{2}\|\mathbf{s} - \left(\mathbf{y}(k) - t_k \nabla \mathbf{F}(\mathbf{y}(k))\right)\|^2 \right\} \tag{14}$$

$$\mathbf{v}(k) = \mathbf{x}(k-1) + \frac{1}{\theta_k}(\mathbf{x}(k) - \mathbf{x}(k-1)) \tag{15}$$

Notice that (13) and (15) are the same as (2b) and (2d). Based on the definition of $\mathbf{G}(x)$, (14) can be converted to the following constrained optimization problem:

$$\mathbf{x}(k) = \arg\min_{\mathbf{U}\mathbf{s}=\mathbf{0}} \left\{ \frac{1}{2}\|\mathbf{s} - \left(\mathbf{y}(k) - t_k \nabla \mathbf{F}(\mathbf{y}(k))\right)\|^2 \right\} \tag{16}$$

the update of $\mathbf{x}(k)$ can be done by a projection operation as follows.

$$\mathbf{x}(k) = \mathbf{P}_\Omega \left(\mathbf{y}(k) - t_k \nabla \mathbf{F}(\mathbf{y}(k))\right) \tag{17}$$

where the subspace $\Omega = \{\mathbf{x} \in \mathbb{R}^p : \mathbf{U}\mathbf{x} = \mathbf{0}\}$, which is the nullspace of matrix $\mathbf{U}$. From the following theorem we can know the 'ideal' projection matrix.

Theorem 2.2: *Denote the projection matrix in* (17) *as* $\mathbf{P}^*$*, then we have* $\mathbf{P}^* = \mathbf{1}\mathbf{1}^T/p$*, where* $\mathbf{1}$ *is the column vector with all ones.*

***Proof*:** Please refer to Appendix 1 for the detailed proof. □

Discussion: From a sensor network point of view, Theorem 2.2 indicates that in each iteration, every node needs to obtain the *exact* average of all the nodes' values. The key challenge here is how to design a scheme to approximate the exact average situation. In the seismic tomography problem we focus on, communication is more expensive than computation, it is favourable to perform communication as less as possible. To this end, we first use a mixing matrix $\widehat{\mathbf{W}}$ to replace $\mathbf{P}^*$ in (17) and then we add a correction term (i.e. $\mathbf{z}(k-1)$, see (2a) and (2c)) to compensate the '*inexact projection*' caused by using $\widehat{\mathbf{W}}$. This 'correction term' idea is similar to the one used in the EXTRA method in [19]. Notice that in our proposed FDC algorithm, within each outer iteration, γ (where γ is a small positive value) rounds of consensus communication are performed and the communication is carried out only among local neighbours. Multiplying mixing matrix $\mathbf{W}$ once means a set of operations that every node sends out its current estimates to its neighbours, receives from neighbours and then mixes with neighbours's

values. Thus $\mathbf{W}^{\gamma}$ represents repeating the aforementioned process γ times. In the presenting work, we also adapt a restart scheme [27] into our FDC algorithm. The restart scheme will tune the parameters approximating the 'optimal' parameter choice, which is unknown in practice. It will further improve the convergence rate.

3. Convergence analysis

In this section, we investigate the convergence behaviour of the proposed algorithm.

Recall our goal is solving (1) in a decentralised fashion. We thus formulate a consensus optimization problem in (9). An optimal solution of (9) should be consensual and minimising the objective function in the same time. Hence two important measures must be considered for analysing our algorithm:

- *consensus measure*: it measures how close every node in the network reaches the consensus solution.
- *optimality measure*: it determines whether the consensual solution is an optimal solution of the problem.

In order to analyse the aforementioned measures, we define the following quantities: $\tilde{x}_{(i)}(k) = x_{(i)}(k) - \frac{1}{p}\sum_{i}^{p} x_{(i)}(k)$ and $\tilde{\mathbf{x}}(k) = \left[\tilde{x}_{(1)}(k)^T, \tilde{x}_{(2)}(k)^T, \ldots, \tilde{x}_{(p)}(k)^T\right]^T$. The convergence results regarding the consensus and optimality are summarised in the following two theorems.

Theorem 3.1: *Consider the sequence* $\{\mathbf{x}(k)\}$ *generated by Algorithm 1. Then the quantity* $\tilde{\mathbf{x}}(k)$ *follows that* $\lim_{k\to\infty}\|\tilde{\mathbf{x}}(k)\| = 0$. *That is, the solution* $\{x_{(i)}, \forall i\}$ *converges to a same consensual solution.*

***Proof*:** The quantity $\tilde{\mathbf{x}}(k)$ can be considered as 'disagreement' since it shows how mutually apart the solutions of different nodes are. By tracking the disagreement iterates, we can measure the consensus among the nodes. We assume the dimension $n = 1$ here to simplify the proof. The proof can be extended to case $n > 1$.

To begin the proof, from (2a), we can have:

$$\mathbf{z}(k-1) = \sum_{t=0}^{k-2}\left(\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}\right)\mathbf{x}(t) \tag{18}$$

Now plug (18) into (2c) and also based on (2b) and (2d), we can obtain the following:

$$\begin{aligned}
\mathbf{x}(k) &= \widehat{\mathbf{W}}\mathbf{y}(k) - \frac{1}{L}\widehat{\mathbf{W}}\nabla\mathbf{F}(\mathbf{y}(k)) - \mathbf{z}(k-1)\\
&= (1-\theta_k)\widehat{\mathbf{W}}\mathbf{x}(k-1) + \theta_k\widehat{\mathbf{W}}\mathbf{v}(k-1) - \sum_{t=0}^{k-2}\left(\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}\right)\mathbf{x}(t) - \frac{1}{L}\widehat{\mathbf{W}}\nabla\mathbf{F}(\mathbf{y}(k))\\
&= \frac{2k-1}{k+1}\widehat{\mathbf{W}}\mathbf{x}(k-1) + \frac{2-k}{k+1}\widehat{\mathbf{W}}\mathbf{x}(k-2) - \sum_{t=0}^{k-2}\left(\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}\right)\mathbf{x}(t) - \frac{1}{L}\widehat{\mathbf{W}}\nabla\mathbf{F}(\mathbf{y}(k))\\
&= t_k\widehat{\mathbf{W}}\mathbf{x}(k-1) + (1-t_k)\widehat{\mathbf{W}}\mathbf{x}(k-2) - \sum_{t=0}^{k-2}\left(\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}\right)\mathbf{x}(t) - \frac{1}{L}\widehat{\mathbf{W}}\nabla\mathbf{F}(\mathbf{y}(k))
\end{aligned} \tag{19}$$

Here we replace θ_j with θ_k (the version without restart scheme, it will not affect the convergence results) to simplify the notation. Also we define $t_k = \frac{2k-1}{k+1}$.

Subtracting the updating rule of $\mathbf{x}(k+1)$ (deducted from (19)) with (19) yields:

$$\begin{aligned}
\mathbf{x}(k+1) - \mathbf{x}(k) = {} & t_{k+1}\widehat{\mathbf{W}}\mathbf{x}(k) + (1 - t_{k+1} - t_k)\widehat{\mathbf{W}}\mathbf{x}(k-1) - (1-t_k)\widehat{\mathbf{W}}\mathbf{x}(k-2)\\
& - \left(\widetilde{\mathbf{W}} - \widehat{\mathbf{W}}\right)\mathbf{x}(k-1) + \frac{1}{L}\widehat{\mathbf{W}}\left[\nabla\mathbf{F}(\mathbf{y}(k)) - \nabla\mathbf{F}(\mathbf{y}(k+1))\right]
\end{aligned} \tag{20}$$

At this point, left-multiply both sides of the above equation by $(\mathbf{I}-\mathbf{P}^*)$ where $\mathbf{P}^*$ is the ideal projection matrix denoted in Theorem 2.2. The left-hand part then becomes $\tilde{\boldsymbol{x}}(k+1)-\tilde{\boldsymbol{x}}(k)$ by definition. To cope with the right-hand side, we first derive some equalities. Let $\widehat{\mathbf{W}}^{'}=\widehat{\mathbf{W}}-\mathbf{P}^*$ and $\widetilde{\mathbf{W}}^{'}=\widetilde{\mathbf{W}}-\mathbf{P}^*$, combining the definitions of $\mathbf{P}^*$, $\widehat{\mathbf{W}}$ and also Assumption 1.4 ($\mathbf{W}$ is doubly stochastic matrix), it follows that:

$$(\mathbf{I}-\mathbf{P}^*)\widehat{\mathbf{W}}=\widehat{\mathbf{W}}^{'}=\widehat{\mathbf{W}}^{'}(\mathbf{I}-\mathbf{P}^*);\ (\mathbf{I}-\mathbf{P}^*)\widetilde{\mathbf{W}}=\widetilde{\mathbf{W}}^{'}=\widetilde{\mathbf{W}}^{'}(\mathbf{I}-\mathbf{P}^*)$$

Equation $(\mathbf{I}-\mathbf{P}^*)\widehat{\mathbf{W}}=\widehat{\mathbf{W}}^{'}$ can be obtained as follows.

$$\begin{aligned}(\mathbf{I}-\mathbf{P}^*)\widehat{\mathbf{W}}&=\widehat{\mathbf{W}}-\mathbf{P}^*\mathbf{W}^{\gamma}=\widehat{\mathbf{W}}-\left(\mathbf{P}^*\mathbf{W}\right)\mathbf{W}^{\gamma-1}\\&=\widehat{\mathbf{W}}-\mathbf{P}^*\mathbf{W}^{\gamma-1}=\widehat{\mathbf{W}}-\left(\mathbf{P}^*\mathbf{W}\right)\mathbf{W}^{\gamma-2}\\&\cdots\\&=\widehat{\mathbf{W}}-\mathbf{P}^*=\widehat{\mathbf{W}}^{'}\end{aligned}\tag{21}$$

$\widehat{\mathbf{W}}^{'}=\widehat{\mathbf{W}}^{'}(\mathbf{I}-\mathbf{P}^*)$ can be proved by showing $\widehat{\mathbf{W}}^{'}\mathbf{P}^*=\mathbf{0}$:

$$\widehat{\mathbf{W}}^{'}\mathbf{P}^*=\left(\widehat{\mathbf{W}}-\mathbf{P}^*\right)\mathbf{P}^*=\mathbf{P}^*-\mathbf{P}^*=\mathbf{0}$$

The equalities regrading matrix $\widetilde{\mathbf{W}}$ can be obtained in a similar manner.

Based on the above properties, we can have:

$$\begin{aligned}\tilde{\boldsymbol{x}}(k+1)-\tilde{\boldsymbol{x}}(k)=&\,t_{k+1}\widehat{\mathbf{W}}^{'}\tilde{\boldsymbol{x}}(k)+(1-t_{k+1}-t_k)\widehat{\mathbf{W}}^{'}\tilde{\boldsymbol{x}}(k-1)-(1-t_k)\widehat{\mathbf{W}}^{'}\tilde{\boldsymbol{x}}(k-2)\\&-\left(\widetilde{\mathbf{W}}^{'}-\widehat{\mathbf{W}}^{'}\right)\tilde{\boldsymbol{x}}(k-1)+\frac{1}{L}\widehat{\mathbf{W}}^{'}\left[\nabla\mathbf{F}(\mathbf{y}(k))-\nabla\mathbf{F}(\mathbf{y}(k+1))\right]\end{aligned}\tag{22}$$

In a matrix form, the dynamics of the disagreement $\tilde{\boldsymbol{x}}(k)$ can be expressed as follows.

$$\begin{aligned}\begin{bmatrix}\tilde{\boldsymbol{x}}(k+1)\\ \tilde{\boldsymbol{x}}(k)\\ \tilde{\boldsymbol{x}}(k-1)\end{bmatrix}=&\begin{bmatrix}\mathbf{I}+t_{k+1}\widehat{\mathbf{W}}^{'} & (2-t_{k+1}-t_k)\widehat{\mathbf{W}}^{'}-\widetilde{\mathbf{W}}^{'} & (t_k-1)\widehat{\mathbf{W}}^{'}\\ \mathbf{I} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{x}}(k)\\ \tilde{\boldsymbol{x}}(k-1)\\ \tilde{\boldsymbol{x}}(k-2)\end{bmatrix}\\&-\begin{bmatrix}\frac{1}{L}\widehat{\mathbf{W}}^{'}\left[\nabla\mathbf{F}(\mathbf{y}(k+1))-\nabla\mathbf{F}(\mathbf{y}(k))\right]\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}\end{aligned}\tag{23}$$

Note that (23) is the state equation for the dynamics of vector $\left[\tilde{\boldsymbol{x}}(k+1)^T,\tilde{\boldsymbol{x}}(k)^T,\tilde{\boldsymbol{x}}(k-1)^T\right]^T$. Hence instead of directly tackling $\tilde{\boldsymbol{x}}(k)$, we will be bounding the norm of the 'augmented' vector to prove the consensus result of the theorem. Refer to appendix for remaining details of the proof. □

Theorem 3.2: *Let all assumptions hold. The sequence* $\{\mathbf{x}(k)\}$ *generated by Algorithm* 1 *converges to a limiting point* $\mathbf{x}^*$ *such that* $\{x^*_{(i)},\forall i\}$ *is the same point in the optimal set* X^* *of* (1).

***Proof*:** Theorem 2.1 indicates that once the solution generated by FDC becomes consensual, it should also be an optimal point of the consensus optimization problem. Together with Theorems 3.1 and 3.2 can be verified accordingly. □

4. Application to *in situ* seismic tomography

In previous, we have discussed the design and analysis of the proposed fast decentralised consensus algorithm. In this section, we illustrate a motivating application – *in situ* seismic tomography and demonstrate how the FDC method fits into it.

The traditional centralised travel-time tomography inversion problem is to solve a linear system of equations:

$$\mathbf{Ax} = \mathbf{b} \tag{24}$$

where **A** matrix contains the ray information from the source to the receiver. Vector **b** is constructed using travel-time information. **x** is the vector to be estimated, which stores the slowness value (reciprocal of velocity) in each block. The inversion problem is equivalent to finding the least-squares solution **x** such that,

$$\mathbf{x} = \arg\min_{\mathbf{x}} \parallel \mathbf{Ax} - \mathbf{b} \parallel^2 \tag{25}$$

In fact, the above system (24) is usually inconsistent since vector **b** is noise-corrupted. Thus, regularisation technique leveraged to better reconstruct the tomography (the regularisation term can vary, we use Tikhonov regularisation here since it is the most popular one in seismic inversion problem):

$$\min_{\mathbf{x}} \ \frac{1}{2}\|\mathbf{Ax} - \mathbf{b}\|_2^2 + \lambda^2\|\mathbf{x}\|_2^2 \tag{26}$$

Notice that problem (26) can be decomposed since the ray information and travel-time information in **A** and **b** are originally generated in a distributed manner. Hence the local objective function F_i of (1) in this scenario can be regarded as:

$$F_i = \frac{1}{2}\|\mathbf{A}_i\mathbf{x} - \mathbf{b}_i\|_2^2 + \lambda_i^2\|\mathbf{x}\|_2^2 \tag{27}$$

where *i*th sensor node has the knowledge of $\mathbf{A}_i$ and $\mathbf{b}_i$ only.

It can be seen that all the assumptions hold regarding the problem formulation here. With the domain knowledge in seismic tomography, the bounded gradient Assumption 1.5 is also valid in this case.

4.1. Decentralised seismic imaging protocol design

In this subsection, we show the protocol design for the *in situ* seismic imaging. The proposed protocol is developed by leveraging the FDC algorithm into the distributed sensor network. Assuming the medium is wireless, we are motivated to propose a broadcast-based protocol. It is fast in seismic computing and also aiming to minimise the communication cost. The protocol is illustrated in Algorithm 2.

Algorithm 2: Decentralised Seismic Imaging Protocol (DSIP): For sensor *i*

1: Each sensor node $i, \forall i$ does update on $y_{(i)}(k)$ (subvector of $\mathbf{y}(k)$ corresponding to node i) based on Equation (2b).
2: Sensor node i computes the quantity $\left[y_{(i)}(k) - \frac{1}{L}\nabla F_i(y_{(i)}(k))\right]$.
3: Sensor node i broadcasts the quantity $\left[y_{(i)}(k) - \frac{1}{L}\nabla F_i(y_{(i)}(k))\right]$ and $x_{(i)}(k-2)$ to all its neighbours within its communication range.
4: Sensor node i receives neighbours' values and performs mixing on them (weighted average) according to ith row of the mixing matrix **W**.
5: Node i conducts additional $(\gamma - 1)$ rounds of steps 3–4.
6: Node i does update on $z_{(i)}(k-1)$ based on Equation (2a).
7: Node i does update on $x_{(i)}(k)$ based on Equation (2c).
8: Node i does update on $v_{(i)}(k)$ based on Equation (2d).
9: Repeats steps 1–8 until the maximum number of iterations is reached.

We note that the DSIP protocol is a two-phase process: *communication phase* and *computation phase*. The communication stage is carried out in steps 3 and 4. These two steps are implementing the second item in (2a) and the first term in (2c), respectively. Considering the operation **Wx**, it indeed enforces each node to mix neighbours' values with its own value. For sensor i, according to Assumption 1.4, the mixing operation is performed as follows.

$$\sum \mathbf{W}_{ij} x_{(j)} \rightarrow x_{(i)}, \forall j \in \mathcal{N}_i \tag{28}$$

where $\mathcal{N}_i$ denotes the set of neighbours of node i (including itself).

The computation phase consists of steps 1–2 and 6–8. Note that in the whole iteration process, each node only needs one gradient computation and simple matrix (vector) addition operations. This kind of light-weight protocol is feasible and promising in practice, especially in applications such as seismic tomography where computation and energy-wise resource is very limited.

Discussion: Other issues may also affect the practical performance of the proposed algorithm. For example, the initialization of **x**. The proposed algorithm guarantees to converge to the true solution with arbitrary initialization. Nevertheless, if we are given the domain knowledge on the possible value or range of the solution **x**, then we can start with a closer solution than certain random value. It will then reach the optimal solution faster in the iterative process.

5. Performance evaluation

5.1. Experiment settings

In this section, we investigate the performance of the proposed FDC algorithm, particularly on the seismic tomography problem.

The FDC algorithm is studied on three different data sets: 2D synthetic data set, 3D synthetic data set and 3D real seismic tomography data set. The synthetic data sets are also constructed simulating the problem of seismic tomography in sensor network. The proposed scheme is implemented in MATLAB and distributed network emulator named common open research emulator (CORE) and EMANE network emulator.[28,29] The reason they are considered as the development and evaluation platform is that the real sensor nodes in our VolcanoSRI[1] project will be some low-power devices, and CORE and EMANE will allows us to closely emulate the future deployment. Experiments are performed on a PC with an Intel i5-3.0G HZ CPU and 8 GB memory.

Regarding the problem model, we use the regularised least squares model in (26) mentioned in the last section, where the regularisation parameter λ is set to 1 in all scenarios. The matrix **A** and vector **t** is constructed by stacking the sub-matrices (sub-vectors) of all the nodes. The resolution means the number of blocks along the x, y and z-axis over the interested region. In each data set, the solution of this 'centralized' problem is pre-computed using LSQR method.[30] This centralised solution is used in the convergence measures and seen as the best benchmark for other decentralised algorithms in terms of the tomography result.

In the presenting work, three recent decentralised methods: EXTRA,[19] D-NG & D-NC [10] are compared with our proposed FDC algorithm. For the decentralised formulation, the corresponding parameter λ_i for each node i is set to $1/p$, where p is the total number of nodes in the network. The communication networks are generated randomly with certain average node degrees.

We exam the quantitative performance of the decentralised algorithm by using the following three metrics.

(1) Relative error along the iteration number (communication round). The relative error is obtained by computing $\sqrt{\left(\sum_{i=1}^{m} \|x_{(i)}(k) - x^*\|_2^2\right) / \left(\sum_{i=1}^{m} \|x_{(i)}(0) - x^*\|_2^2\right)}$ in kth iteration where x^* is the pre-computed centralised solution. This metric can show us how close each node's solution to the 'optimal' point on average.

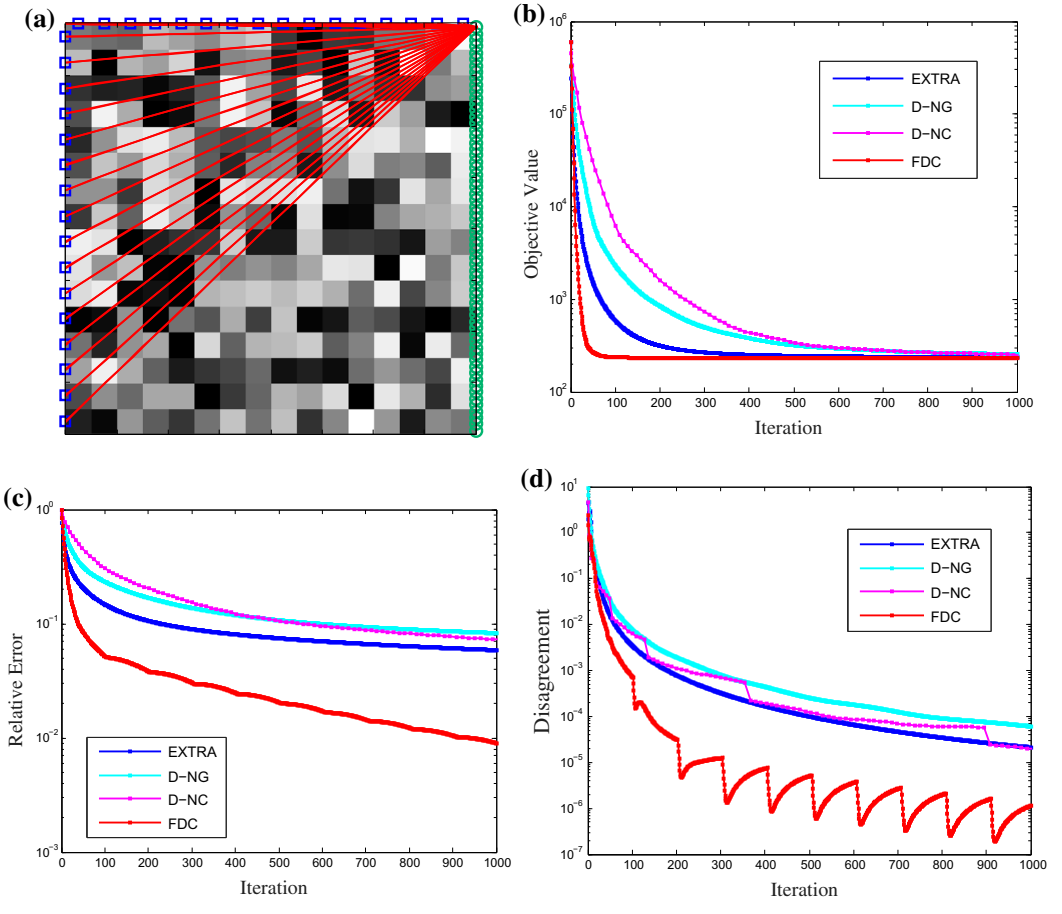


Figure 3. Convergence behaviour comparison in 2-D synthetic seismic data set. (a) is the overview of the experiment environment. (b) is the illustration of average objective value for all the methods and comparison plot in terms of objective value. (c) is the plot of relative error comparing FDC with EXTRA, D-NG and D-NC methods. (d) is the comparison plot in terms of the disagreement measure.

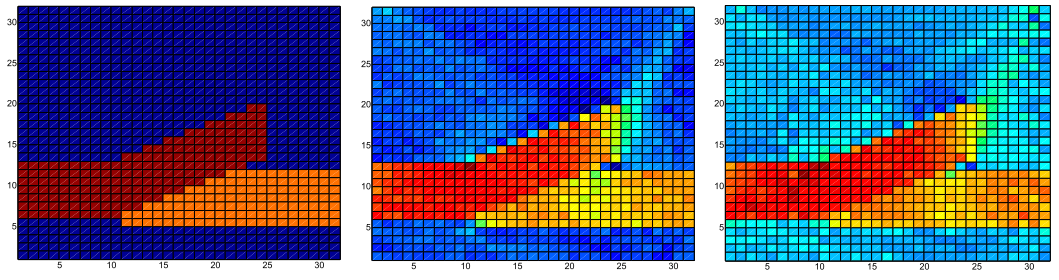
(2) The disagreement quantity $\|\tilde{\boldsymbol{x}}\|$ along the iteration round. The metric corresponds to the consensus measure discussed in Section 3. It provides the information on how fast the nodes in the network reach consensus (agreement).

(3) Average objective value along the iteration number. The metric is defined as $\frac{1}{p}\sum_{i=1}^{p} F_i(x_{(i)}(k))$. Recall that the objective function in (9) is indeed the scaled version of this metric. Hence it is necessary to check how the average objective value decreases.

Notice that the optimization problem (26) we investigate is strongly convex. Thus there is only one unique optimal solution x^* and all the valid decentralised algorithms should converge to a same point. In general convex case, various methods might converge to different optimal points although the objective function is minimised.

5.2. Synthetic 2-D case

The synthetic 2-D seismic data set is generated by using the code in [31]. A seismic tomography test problem is created with a 2-D square domain. The event sources are located on the right boundary (green dots) and receivers (seismographs) are scattered along the left and top boundary (blue squares).



(a) Groud truth (b) Centralized (c) FDC

Figure 4. Tomography result of FDC algorithm in synthetic 2-D data set. Centralised solution in (b) is obtained after running 10 iterations of LSQR method. (c) is the tomography result of FDC algorithm by averaging the solutions of all the nodes in the network after 30 message communications (iterations).

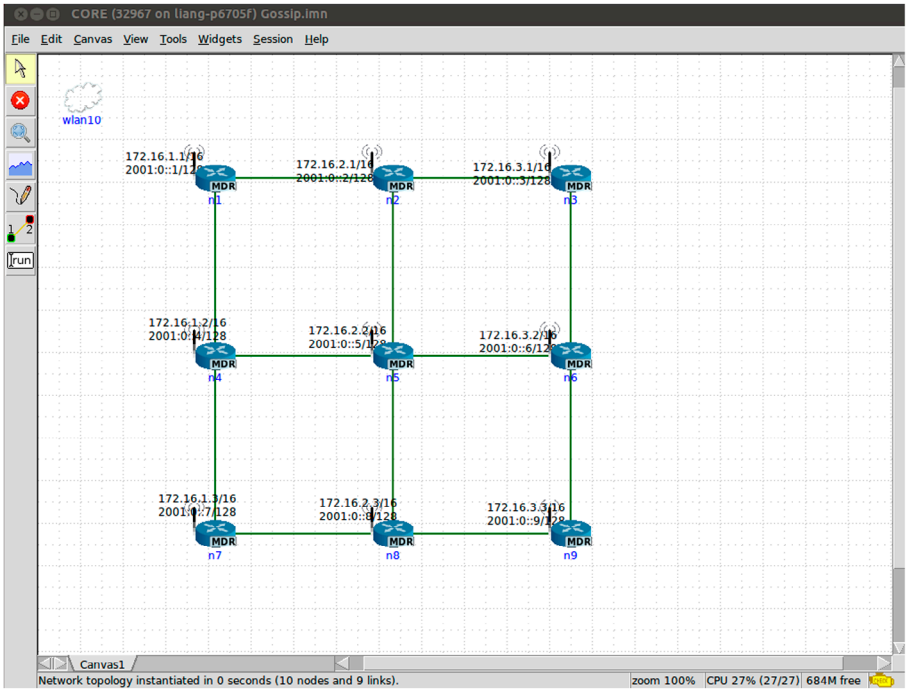


Figure 5. CORE Emulator GUI.

The seismic rays are set to be transmitted from each source to each receiver (red lines) (See Figure 3(a) for illustration).

The square region is partitioned into 32×32 smaller blocks (the resolution is 32×32) and we want to estimate the values (slowness) in these blocks (i.e. value of x). One twenty-eight events simulated and there are 64 sensor nodes in the network. Hence the dimension of matrix $\mathbf{A}$ is 8192×1024 and $\mathbf{x}$ is 1024×1, $\mathbf{b}$ is 8192×1, accordingly. A 5% gaussian noise is added into the data vector $\mathbf{b}$. The size of each submatrix $\mathbf{A}_i, \forall i$ is 128×1024. The communication network is constructed such that each node has three immediate neighbours on average. The experiment results are depicted in Figures 3 and 4.

In Figure 3, we can see that the proposed FDC algorithm outperforms the other benchmarks in terms of all the three convergence measures. Note that the curves of FDC have steeper slopes than EXTRA, D-NG and D-NC. It implies that FDC has better convergence rate performance in practice. As

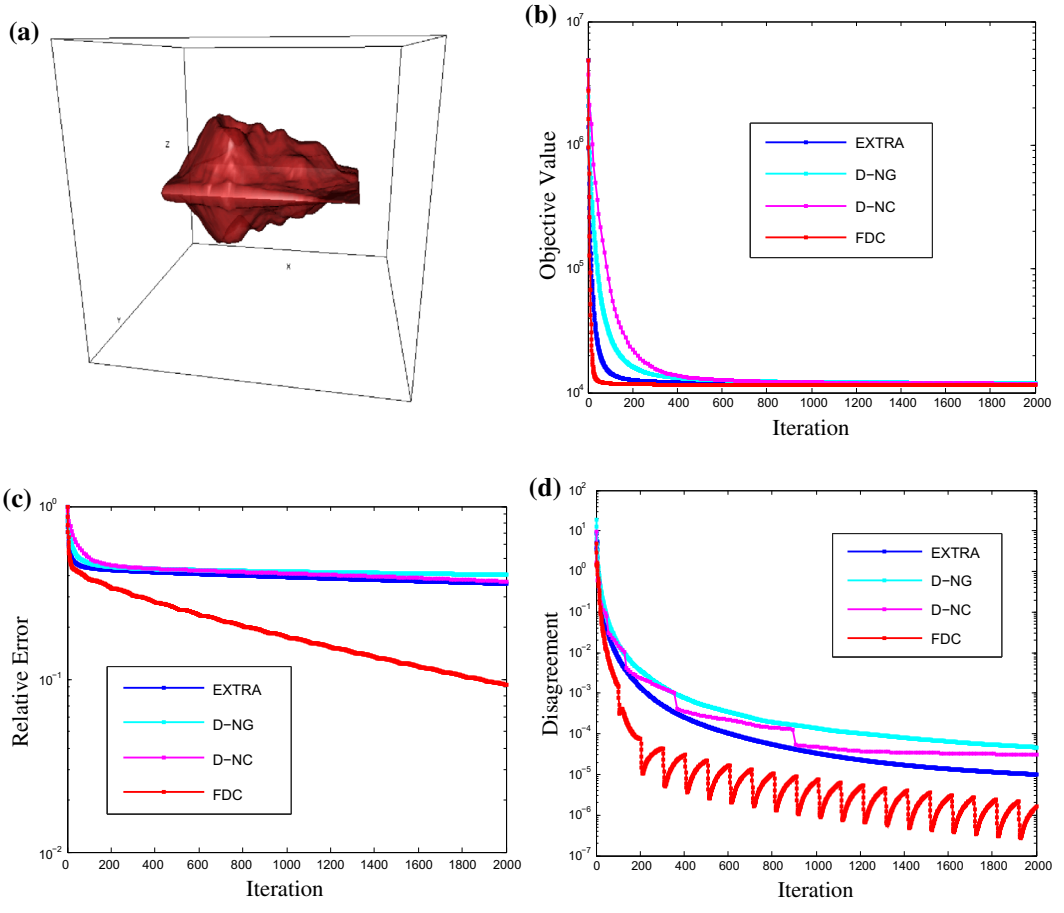


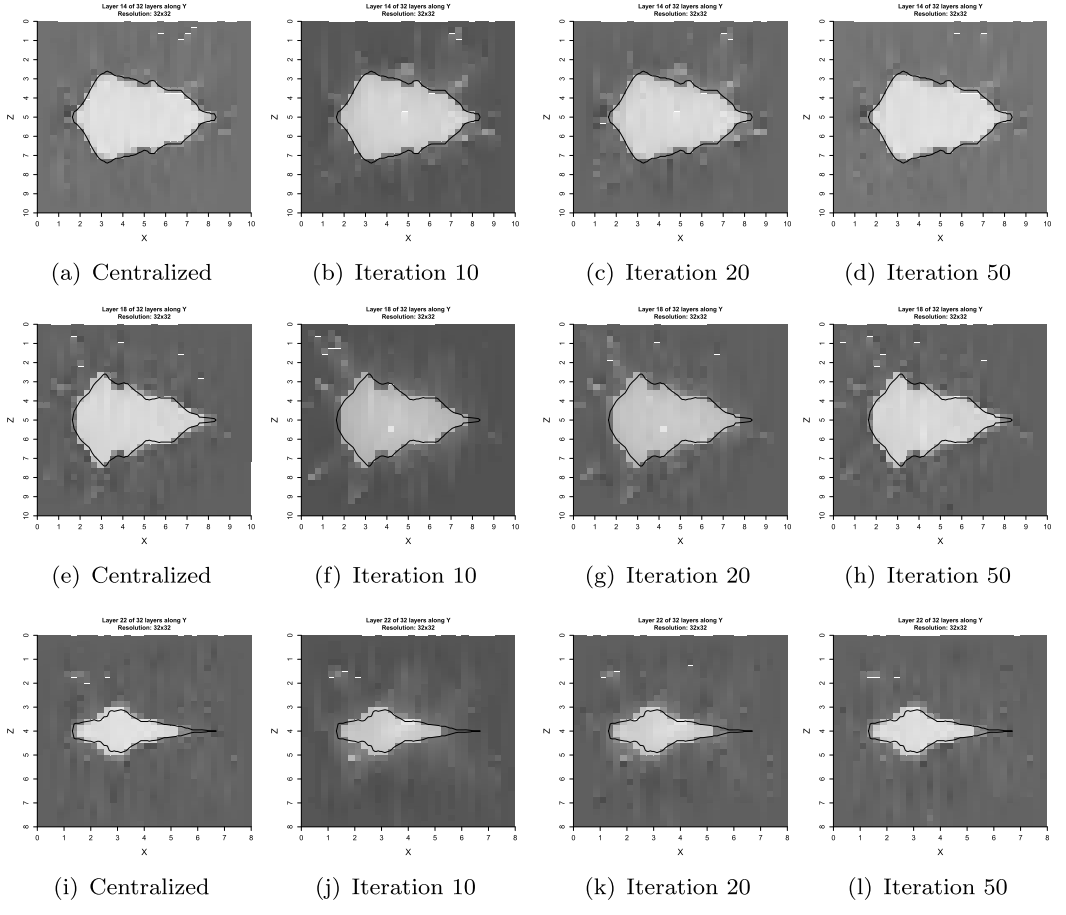
Figure 6. Convergence behaviour comparison in 3-D synthetic seismic data set. (a) is the ground truth of the magma model with resolution 128 × 128 × 128. (b) is the illustration of average objective value for all the methods and comparison plot in terms of objective value. (c) is the plot of relative error comparing FDC with EXTRA, D-NG and D-NC methods. (d) is the comparison plot in terms of the disagreement measure.

the iteration number goes, the difference between FDC and the other methods would become even bigger.

Figure 4 demonstrates the tomography result of FDC scheme along with the images of ground truth and the centralised solution. Note that the quality of the recovered tomography depends on the model we use. Designing appropriate model consists of choosing regularisation methods and parameters, which is determined by the application. In this paper, we focus on the computation methodology. In this situation, the centralised solution can be seen as optimal given a specific model. In addition, we propose solution based upon fast decentralised algorithm, which is approaching the centralised solution iteratively. We can find that the FDC result in Figure 3(c) is close to the counterpart of centralised solution in Figure 3(b). More importantly, the difference is mainly on the background. The region we are interested named 'fault zone' (the brown and yellow part in Figure 3(a)) is almost recovered in FDC result comparing to the corresponding part in Figure 3(b).

5.3. Synthetic 3-D case

In this subsection, we evaluate the proposed FDC algorithm by demonstrating a simulated seismic data set on a 3-D synthetic model of resolution 32 × 32 × 32. The model 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

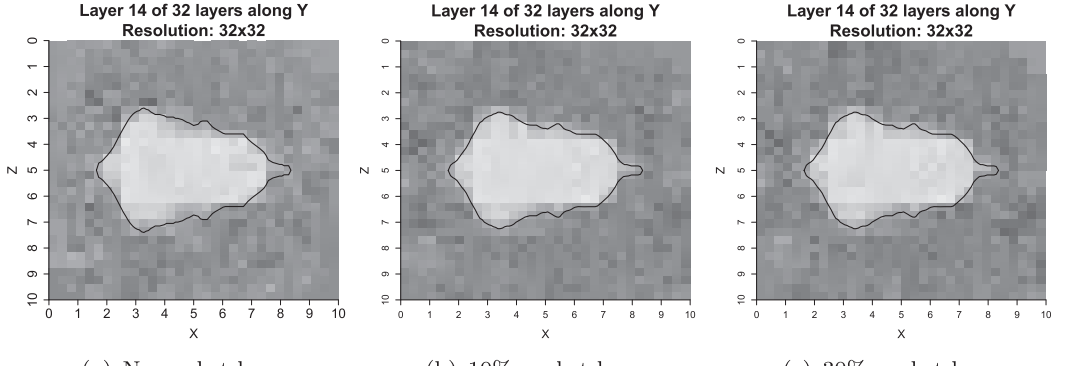


(a) Centralized (b) Iteration 10 (c) Iteration 20 (d) Iteration 50

(e) Centralized (f) Iteration 10 (g) Iteration 20 (h) Iteration 50

(i) Centralized (j) Iteration 10 (k) Iteration 20 (l) Iteration 50

Figure 7. Seismic tomography results of FDC in 3-D synthetic data set. at Layer 22. (a)–(d) are the vertical-sliced results at layer 14. (e)–(h) are the tomography comparison at layer 18. The seismic images related to layer 22 are shown in (i)–(l).



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

Figure 8. Effect of packet loss in FDC.

distributed on top of the region. Four hundred 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,

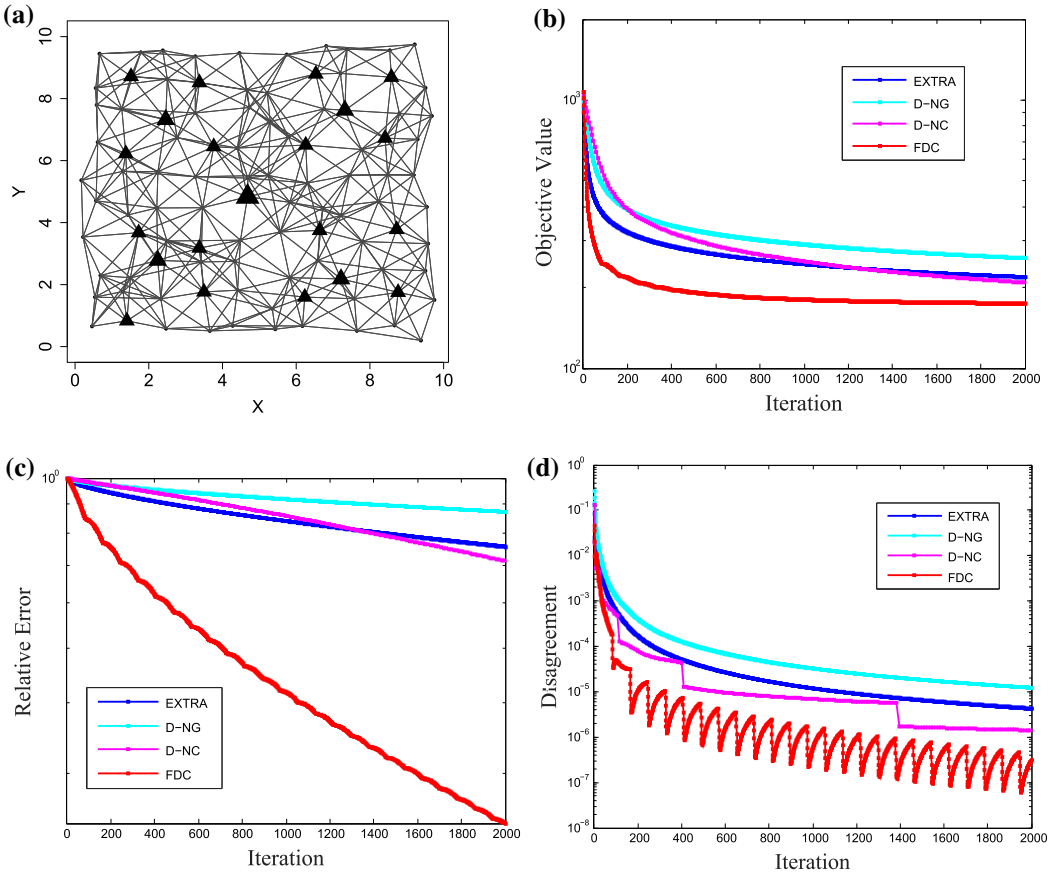


Figure 9. Convergence behaviour comparison in 3-D real seismic data set. (a) is a sketchy example of the network topology. (b) is the illustration of average objective value for all the methods and comparison plot in terms of objective value. (c) is the plot of relative error comparing FDC with EXTRA, D-NG and D-NC methods. (d) is the comparison plot in terms of the disagreement measure.

a white Gaussian noise is added to the travel time to construct the sensor node observations (arrival times). Similar as the 2-D case, $\mathbf{A}$ is formed as a $40,000 \times 32,768$ sparse matrix. For the communication network, each senor is assumed to have 10 neighbours on average within its communication range.

From Figure 6, we see that FDC converges much faster than the other algorithms in all three measures. An interesting observation is that in both the 2-D and the underlying case, D-NC is slower than EXTRA method while D-NC has a better theoretical convergence rate than EXTRA ($O(1/k^2)$ vs. $O(1/k)$). The reason might be that there are too many consensus communication rounds within each outer loop. The number of consensus rounds in D-NC grows rapidly as the iteration number k increases. These kind of communications might be wasted and unnecessary for fast convergence of algorithm.[10,19]

We also implement the protocol DSIP in C++ and test it in CORE network emulator (See Figure 5 for an example of 3 by 3 grid network in CORE GUI). Each node in the network is allocated a Linux Virtual Machine in the emulation platform. The DSIP protocol is added as a service in each node. All the nodes over the network then start and execute the service in a distributed manner. The workflow of the program is summarised as follows.

(1) Node i performs initialization and set up related services.
(2) Node i starts a 'listener' thread and keeps waiting for the neighbours' information.

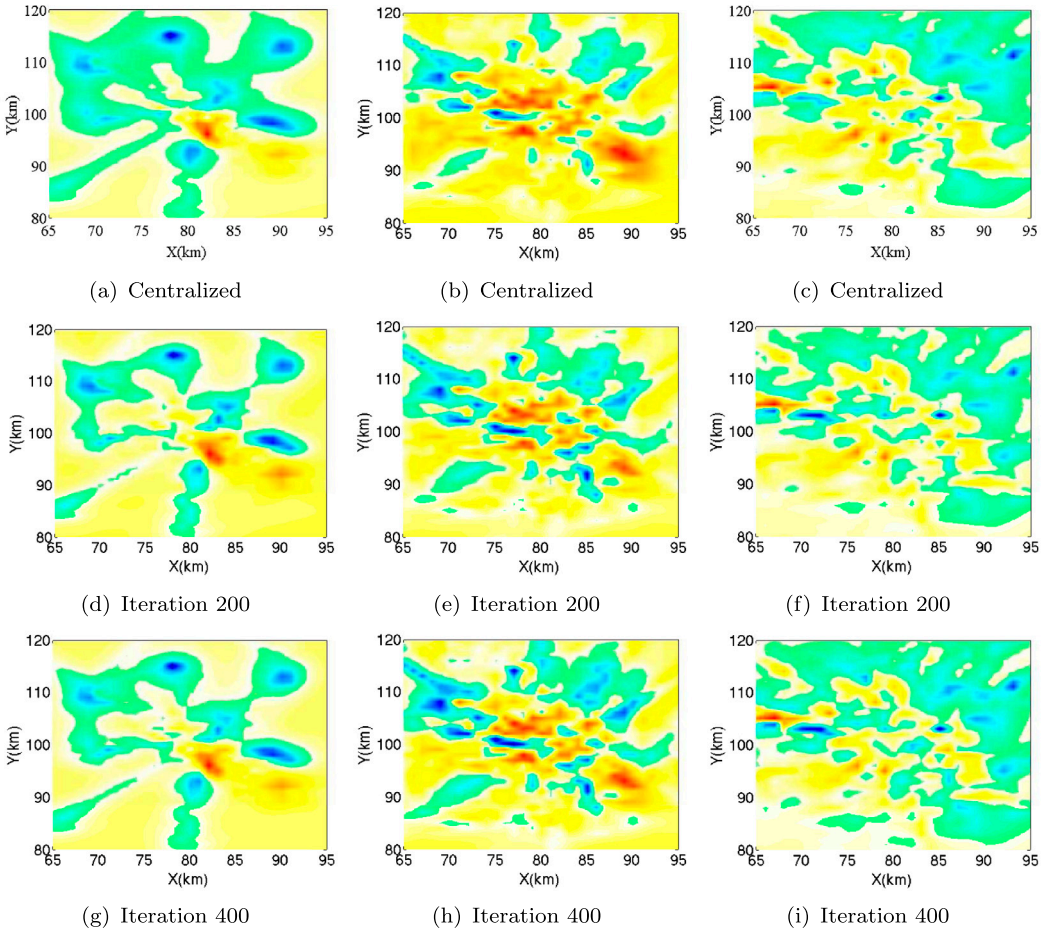


Figure 10. Seismic tomography comparison of the 3-D real data set. The first column (a), (d) and (g) describe the solutions of vertical slice layer 5 at depth 0.9 km (top: centralised solution, middle: FDC result after 200 iteration, bottom: FDC tomography at 400th iteration). The middle column (b), (e) and (h) exhibit the tomography results of layer 7 at depth 2.9 km. (c), (f) and (i) are results at layer 9 of depth 4.9 km.

(3) After certain time, the node does the mixing operation according to the solutions received from its neighbours during that period. The mixing parameters $\mathbf{W}_{ij}$ (for node i) may vary in each iteration depending on how many neighbours' solutions are actually received by node i.
(4) The mixed solution is then used for the following computation phase to do update.
(5) Node i broadcasts the updated information using UDP protocol.
(6) The loop is repeated until the pre-defined number of iterations is reached.

We compare the centralised and decentralised solutions (at iteration 10, iteration 20 and iteration 50). The results are vertical slices along the axes *Y*. In Figure 7, we demonstrate the tomography results at layer 14, layer 18 and layer 22, respectively. An important observation is that at iteration 10, the decentralised solution has began to recovery the envelop of the magma area (shown in the black solid line.). Furthermore, at iteration 50, the decentralised solution is almost visually indistinguishable with the centralised solution. This property empowers FDC-based framework to be a promising solution providing real-time *in situ* processing, which is desirable in micro-seismic tomography, disaster monitoring and other applications.

The robustness of FDC is evaluated in Figure 8. We test with packet loss ratios 10 and 30% in the emulator. It is clear to see 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-tolerantance and robustness of FDC method, especially in applications like seismic tomography, which always suffers from severe packet loss.

5.4. Real data case

In this subsection, we study the performance of the proposed FDC in real 3-D data set. The experiments are based on 10 years (2001–2011) real seismic event data around volcano Mount St. Helens in Washington, USA. The data were collected originally from 78 stations and we construct them into 11 clusters. The cluster heads will form a network which FDC-based protocol would perform on. A sketchy example is shown in Figure 9(a). The size of the $\mathbf{A}$ matrix is $18,161 \times 768,000$ and the underlying resolution is $160 \times 200 \times 24$. Notice that the scheme in this case is set to be hierarchical. Each sensor node firstly transmits its data into its cluster head and the cluster heads will then perform the decentralised algorithm to collaboratively solve the problem. The black triangles denotes the cluster heads in the network. The reason we assume this hierarchical processing scheme is that in this case each sub-system is very under-determined (number of rows in $\mathbf{A}_i$ is much less than the number of columns). That means each node contains very limited information for reconstructing the whole tomography. By gathering data from nodes into local cluster heads, each head would have reasonable amount of information for the following decentralised computing. It might be better than the fully decentralised version in terms of communication overhead in practice.

Note that unlike synthetic data used in previous section, there is no ground truth in this real data scenario. In other words, the true velocity structure of volcano Mount St. Helens is currently unknown. Hence we focus on the comparison of the proposed methods with centralised processing scheme, which can be seen as a benchmark that fully utilise the data available.

Similar to the previous two data sets, FDC demonstrates better convergence performance in Figure 9. It is worth noting that the relative error (in Figure 9(c)) is higher than the other two cases for certain number of iterations. This is because the decentralised computing problem is 'harder' to solve in this real data set.

Figure 10 illustrates vertical slices of tomography model with various depths (along Z-axis). The range of X-axis is from 65 to 95 km, and the range of Y-axis is from 80 to 120 km. The colour in Figure 10 represents the relative velocity perturbation in specific location. More red means larger (negative) value of perturbation. More blue means larger (positive) value of perturbation. It is shown in Figure 10 that FDC (at the solution of 200 iterations) can effectively invert the tomography model close to the benchmark (centralized algorithm) using real data.

6. Conclusion

This paper investigated the decentralised consensus optimization problem where the objective function is the sum of a bunch of convex and differentiable local functions. We proposed a fast decentralised consensus algorithm, which does not require diminishing step sizes. The developed algorithm is then leveraged to solve big data computing problem in seismic tomography. We conducted extensive tests of proposed algorithm on both synthetic and real data sets. The experiment results show that our designed method outperforms the other benchmarks in all cases. The merits of the proposed methods elucidate that they are promising solutions for real-time *in situ* seismic tomography in the near future.

Note

1. http://sensorweb.cs.gsu.edu/?q=VolcanoSRI.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Liang Zhao is currently a PhD student in the Department of Computer Science, Georgia State University, Atlanta, GA, USA. His current research is focused on the cyber-physical system, distributed and decentralised optimisation and big data analytics. He obtained his MS from Lehigh University, PA, USA.

WenZhan Song is now a professor in the College of Engineering, University of Georgia. His research mainly focuses on sensor web, smart grid and smart environment where sensing, computing, communication and control play a critical role and need a transformative study. His research has received 6 million+ research funding from NSF, NASA, USGS, Boeing, etc. since 2005. He is an IEEE Senior Member.

References

[1] Lees JM. Seismic tomography of magmatic systems. J. Volcanol. Geotherm. Res. 2007;167:37–56.
[2] Song WZ, Huang R, Xu M, et al. Air-dropped sensor network for real-time high-fidelity volcano monitoring. In: The 7th Annual International Conference on Mobile Systems, Applications and Services (MobiSys); Poland: Kraków; 2009 Jun.
[3] Bording RP, Gersztenkorn A, Lines LR, et al. Applications of seismic travel-time tomography. Geophys. J. R. Astron. Soc. 1987;90:285–303. doi:10.1111/j.1365-246X.1987.tb00728.x.
[4] Cevher V, Becker S, Schmidt M. Convex optimization for big data: scalable, randomized, and parallel algorithms for big data analytics. Signal Process. Mag. IEEE. 2014;31:32–43.
[5] Matei I, Baras J. Performance evaluation of the consensus-based distributed subgradient method under random communication topologies. Sel. Top. Signal Process. IEEE J. 2011;5:754–771.
[6] Nedic A, Ozdaglar A. Distributed subgradient methods for multi-agent optimization. IEEE Trans. Autom. Control. 2009;54:48–61.
[7] Nedic A, Olshevsky A. Distributed optimization over time-varying directed graphs. In: 2013 IEEE 52nd Annual Conference on Decision and Control (CDC); 2013 Dec; Florence. p. 6855–6860.
[8] Nedic A, Olshevsky A. Stochastic gradient-push for strongly convex functions on time-varying directed graphs. 2014. arXiv:1406.2075.
[9] Chen I-A. Fast distributed first-order methods [PhD thesis]. Boston (MA): Massachusetts Institute of Technology; 2012.
[10] Moura MF, Jakovetic D, Xavier J. Fast distributed gradient methods. 2014. arXiv:1112.2972v4.
[11] Yuan K, Ling Q, Yin W. On the convergence of decentralized gradient descent. 2013. arXiv:1310.7063.
[12] Zargham M, Ribeiro A, Jadbabaie A. A distributed line search for network optimization. In: American Control Conference (ACC), 2012; 2012 Jun; Montreal. p. 472–477.
[13] Xiao L, Boyd S, Lall S. A scheme for robust distributed sensor fusion based on average consensus. In: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, IPSN '05, Los Angeles, California; Piscataway, NJ: IEEE Press; 2005. Available from: http://dl.acm.org/citation.cfm?id=1147685.1147698.
[14] Tsitsiklis J, Bertsekas D, Athans M. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. IEEE Trans. Autom. Control. 1986;31:803–812.
[15] Tsitsiklis JN. Problems in decentralized decision making and computation. Technical report. DTIC Document; 1984.
[16] Terelius UTH, Murray R. Decentralized multi-agent optimization via dual decomposition. The 18th World Congress of the International Federation of Automatic Control (IFAC) in Milano; Italy, 2011.
[17] Shi G, Johansson KH. Finite-time and asymptotic convergence of distributed averaging and maximizing algorithms. 2012. arXiv:1205.1733.
[18] Rabbat M, Nowak R. Distributed optimization in sensor networks. Third International Symposium on Information Processing in Sensor Networks, 2004. IPSN 2004; 2004 Apr; Berkeley (CA). p. 20–27.
[19] Shi W, Ling Q, Wu G. Extra: an exact first-order algorithm for decentralized consensus optimization. 2014. arXiv:1404.6264.
[20] Wei E, Ozdaglar A. On the o(1/k) convergence of asynchronous distributed alternating direction method of multipliers. 2013. arXiv:1307.8254.
[21] Iutzeler PCF, Bianchi P, Hachem W. Asynchronous distributed optimization using a randomized alternating direction method of multipliers. 2013. arXiv:1303.2837.
[22] Zhao L, Song WZ, Shi L, et al. Decentralised seismic tomography computing in cyber-physical sensor systems. Cyber-phys. Syst. 2015;1:91–112.
[23] Xiao L, Boyd S. Fast linear iterations for distributed averaging. Syst. Control Lett. 2003;53:65–78.

[24] Boyd S, Diaconis P, Xiao L. Fastest mixing markov chain on a graph. SIAM Rev. 2004;46:667–689. doi:10.1137/S0036144503423264.
[25] Xiao L, Boyd S, Kim SJ. Distributed average consensus with least-mean-square deviation. J. Parallel Distrib. Comput. 2007;67:33–46. doi:10.1016/j.jpdc.2006.08.010.
[26] Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. Imaging Sci. 2009;2:183–202. doi:10.1137/080716542.
[27] O'Donoghue B, Candes E. Adaptive restart for accelerated gradient schemes. Found. Comput. Math. 2015;15:715–732. doi:10.1007/s10208-013-9150-3.
[28] Ahrenholz J. Comparison of CORE network emulation platforms. In: Military Communications Conference, 2010 – MILCOM 2010; 2010 Oct; San Jose (CA). p. 166–171.
[29] Ahrenholz J, Goff T, Adamson B. Integration of the CORE and EMANE network emulators. In: Military Communications Conference, 2011 – MILCOM 2011; 2011; Baltimore (MD). p. 1870–1875.
[30] Paige CC, Saunders MA. LSQR: an algorithm for sparse linear equations and sparse least squares. ACM Trans. Math. Software. 1982;8:43–71. doi:10.1145/355984.355989.
[31] Hansen PC, Saxild-Hansen M. Air tools - a matlab package of algebraic iterative reconstruction methods. J. Comput. Appl. Math. 2012;236:2167–2178. Inverse Problems: Computation and Applications. Available from: http://www.sciencedirect.com/science/article/pii/S0377042711005188.
[32] Petersen KB, Pedersen MS. The matrix cookbook. 2012. Version 20121115. Available from: http://www2.imm.dtu.dk/pubdb/p.php?3274.

Appendix 1. Proof of Theorem 2.2

Based on linear algebra theory, we can obtain the projection matrix onto the nullspace of matrix $\mathbf{U}$ as:

$$\mathbf{P}^* = \mathbf{I} - \mathbf{U}^T \left(\mathbf{U}\mathbf{U}^T\right)^{-1} \mathbf{U} \tag{A1}$$

Now we express $\mathbf{W}$ using its eigendecomposition as $\mathbf{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, where $\mathbf{Q}$ is the matrix containing eigenvectors of $\mathbf{W}$ such that $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_p]$ and $\mathbf{Q}$ is also normalized to be unitary. $\mathbf{\Lambda} = \text{diag}\left\{\lambda_1(\mathbf{W}), \lambda_2(\mathbf{W}), \ldots, \lambda_p(\mathbf{W})\right\}$, where $\lambda_i(\mathbf{W})$ is the ith largest eigenvalue of $\mathbf{W}$ and the corresponding eigenvector is $\mathbf{q}_i$.

Recall the definition of $\mathbf{U}$ that $\mathbf{U}\mathbf{U}^T = \mathbf{I} - \mathbf{W}$. In addition, based on the unitary property of matrix $\mathbf{Q}$ and the Woodbury matrix identity,[32] we can have the following deductions:

$$\begin{aligned}
\mathbf{P}^* &= \mathbf{I} - \mathbf{U}^T \left(\mathbf{U}\mathbf{U}^T\right)^{-1} \mathbf{U} \\
&= \mathbf{I} - \mathbf{U}^T \left(\mathbf{I} - \mathbf{W}\right)^{-1} \mathbf{U} \\
&= \mathbf{I} - \mathbf{U}^T \left(\mathbf{I} - \mathbf{Q}\mathbf{3}\mathbf{Q}^T\right)^{-1} \mathbf{U} \\
&= \mathbf{I} - \mathbf{U}^T \left[\mathbf{I} + \mathbf{Q}\left(\Lambda^{-1} - \mathbf{Q}^T\mathbf{Q}\right)^{-1} \mathbf{Q}^T\right] \mathbf{U} \\
&= \mathbf{W} - \mathbf{U}^T\mathbf{Q}\left(\Lambda^{-1} - \mathbf{I}\right)^{-1} \mathbf{Q}^T\mathbf{U} \\
&= \mathbf{W} - \mathbf{U}^T\mathbf{Q}\widehat{\Lambda}\mathbf{Q}^T\mathbf{U}
\end{aligned} \tag{A2}$$

where diagonal matrix $\widehat{\Lambda} = \left(\Lambda^{-1} - \mathbf{I}\right)^{-1}$. Next, under Assumption 1.4, we can have the following identities.

$$\mathbf{W}\mathbf{1} = \mathbf{1}\,;\ \mathbf{1}^T\mathbf{W} = \mathbf{1}^T. \tag{A3}$$

Now the proof of the theorem boils down to proving the following equality:

$$\mathbf{W} - \mathbf{U}^T\mathbf{Q}\widehat{\Lambda}\mathbf{Q}^T\mathbf{U} = \mathbf{1}\mathbf{1}^T/p \tag{A4}$$

Left-multiply $\mathbf{1}^T$ and right-multiply $\mathbf{1}$ onto both sides of (A3) yielding:

$$p - \mathbf{1}^T\mathbf{U}^T\mathbf{Q}\widehat{\Lambda}\mathbf{Q}^T\mathbf{U}\mathbf{1} = p \tag{A5}$$

Hence the remaining is to show $\mathbf{1}^T\mathbf{U}^T\mathbf{Q}\widehat{\Lambda}\mathbf{Q}^T\mathbf{U}\mathbf{1} = 0$. Since we have:

$$\begin{aligned}\|\mathbf{Q}^T\mathbf{U}\mathbf{1}\|^2 &= \mathbf{1}^T\mathbf{U}^T\mathbf{Q}\mathbf{Q}^T\mathbf{U}\mathbf{1}\\ &= \mathbf{1}^T\mathbf{U}^T\mathbf{U}\mathbf{1}\\ &= \mathbf{1}^T(\mathbf{I}-\mathbf{W})\mathbf{1}\\ &= 0\end{aligned} \tag{A6}$$

implying that $\mathbf{Q}^T\mathbf{U}\mathbf{1} = 0$, thus (A5) is valid, which completes the proof of Theorem 2.2.

Appendix 2. Proof of Theorem 3.1

We borrow the idea in [10] to prove the theorem. Set the notation that:

$$\Delta(k,u) = \prod_{s=1}^{-k+u-1}\begin{bmatrix}\mathbf{I}+t_{k+s}\widehat{\mathbf{W}}' & (2-t_{k+s}-t_{k+s-1})\widehat{\mathbf{W}}'-\widetilde{\mathbf{W}}' & (t_{k+s-1}-1)\widehat{\mathbf{W}}'\\ \mathbf{I} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I} & \mathbf{0}\end{bmatrix}$$

and

$$\Theta(k) = \begin{bmatrix}\frac{1}{L}\widehat{\mathbf{W}}'\left[\nabla\mathbf{F}(\mathbf{y}(k))-\nabla\mathbf{F}(\mathbf{y}(k+1))\right]\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}$$

then by iterating the right-hand side of the state transition Equation (23) (set $\left[\tilde{\boldsymbol{x}}(1)^T,\tilde{\boldsymbol{x}}(0)^T,\tilde{\boldsymbol{x}}(-1)^T\right]^T = [\mathbf{0},\mathbf{0},\mathbf{0}]$), we can have

$$\left[\tilde{\boldsymbol{x}}(k+1)^T,\tilde{\boldsymbol{x}}(k)^T,\tilde{\boldsymbol{x}}(k-1)^T\right]^T = \sum_{u=0}^{k}\Delta(k,u+1)\,\Theta(u) \tag{B7}$$

The norm of the $\Theta(u)$ can be upper bounded (by using Assumption 1.5 and the property that the largest eigenvalue of $\widehat{\mathbf{W}}'$ is less than 1) since

$$\|\widehat{\mathbf{W}}'\left[\nabla\mathbf{F}(\mathbf{y}(k))-\nabla\mathbf{F}(\mathbf{y}(k+1))\right]\| \le \|\nabla\mathbf{F}(\mathbf{y}(k))\| + \|\nabla\mathbf{F}(\mathbf{y}(k+1))\| \le 2pG \tag{B8}$$

To upper bound the norm of $\left[\tilde{\boldsymbol{x}}(k+1)^T,\tilde{\boldsymbol{x}}(k)^T,\tilde{\boldsymbol{x}}(k-1)^T\right]^T$, the next step is then bounding the norm of $\Delta(k,u)$, which boils down to bounding the norm of the following matrix.

$$\begin{bmatrix}1+t_{k+1}\lambda_i(\widehat{\mathbf{W}}') & (2-t_{k+1}-t_k)\lambda_i(\widehat{\mathbf{W}}')-\lambda_i(\widetilde{\mathbf{W}}') & (t_k-1)\lambda_i(\widehat{\mathbf{W}}')\\ 1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix} \tag{B9}$$

where $\lambda_i(\cdot)$ denotes the ith largest eigenvalue of the underlying matrix. Based on the properties of $\widehat{\mathbf{W}}'$ and $\widetilde{\mathbf{W}}'$, we can show that the norm of the varying part of (B9) is less than 1. Thus the norm of $\Delta(k,u)$ is decreasing along the iteration number k. Finally we can obtain that

$$\sum_{t=0}^{\infty}\|\left[\tilde{\boldsymbol{x}}(s+1)^T,\tilde{\boldsymbol{x}}(s)^T,\tilde{\boldsymbol{x}}(s-1)^T\right]^T\| \le \infty$$

Furthermore, $\lim_{k\to\infty}\left[\tilde{\boldsymbol{x}}(s+1)^T,\tilde{\boldsymbol{x}}(s)^T,\tilde{\boldsymbol{x}}(s-1)^T\right]^T\| = 0$. Theorem 3.1 can be then justified accordingly. Please refer details in [10].