ATU: An Aggregate-Then-Update Diffusion Intelligent Estimation Scheme for Adaptive Networked Systems

Mou Wu, *Member, IEEE*, Naixue Xiong, *Senior Member, IEEE*, Liansheng Tan, Jianguo Wei, *Member, IEEE*, and Jie Wu, *Fellow, IEEE*

***Abstract*—For distributed estimation arising in the nonlinear least squares (NLLSs) problems over adaptive networks, where every node has the abilities of data processing and learning, only the incomplete local data are exploited by the traditional noncooperative method, thereby resulting in the degradation on estimation performance. In this article, a cooperative diffusion strategy is proposed by using a Gauss–Newton (GN) method in order to fully utilize the diversity of temporal–spatial data on local updates. The proposed algorithm includes two steps, i.e., aggregate then update (ATU), where the aggregating step collects in real time the global information instead of local information due to the diffusion strategy, and the updating step implements the local GN iteration. The resulting ATU diffusion algorithm is a distributed and cooperative system without any increase on communication cost, as compared with the noncooperative version. Based on the detailed convergence analysis for ATU, which is fundamental to the promotion of this algorithm, the sufficient conditions for convergence are derived and the evidences of faster convergence than the noncooperative version are provided. The simulation results confirm the obtained theoretical derivations by applying the ATU algorithm to an NLLS-based target localization problem and show the cooperation gains in many aspects, such as the convergence rate, steady-state accuracy, and robustness to noisy range, step size, node, and link failures.**

***Index Terms*—Adaptive networks, cooperation strategy, diffusion, distributed estimation, Gauss–Newton (GN) method, nonlinear least squares (NLLSs), target localization.**

Manuscript received June 12, 2020; revised January 27, 2021; accepted March 31, 2021. Date of publication April 27, 2021; date of current version May 18, 2022. This work was supported by the Local Science and Technology Development Fund guided by Central Government under Grant 2019ZYYD009. This article was recommended by Associate Editor W. Sun. *(Corresponding author: Naixue Xiong.)*

Mou Wu is with the School of Computer Science and Technology, Hubei University of Science and Technology, Xianning 437100, China (e-mail: mou.wu@163.com).

Naixue Xiong is with the School of Resources Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China (e-mail: xiongnaixue@gmail.com).

Liansheng Tan is with the College of Science and Engineering, University of Tasmania, Hobart, TAS 7001, Australia, and also with the Computer Science Department, Central China Normal University, Wuhan 430079, China (e-mail: l.tan@mail.ccnu.edu.cn).

Jianguo Wei is with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: jianguo@tju.edu.cn).

Jie Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TSMC.2021.3071880.

Digital Object Identifier 10.1109/TSMC.2021.3071880

I. Introduction

GAUSS–NEWTON (GN) method has been widely found in many applications, such as deep learning in artificial intelligence and neural network [1]–[3], and parameter estimation in a networked system [4], [5]. Derived from Newton's method, GN algorithm discards the second-order terms in the computation of Hessian for small residual nonlinear least squares (NLLSs) problems, thereby only computing the Jacobian for saving in computation and maintaining fast convergence. Such amount of computations can be further reduced via the mathematical process. In order to compute easily the first derivative of the objective function, the perturbed GN method is proposed in [6], where a perturbed derivative version substitutes the original one. The truncated GN method [7] is proposed to implement the inexact update instead of the exact one. The truncated-perturbed GN method [7] integrates the above two advantages into the update step.

Many scenarios can be modeled as the NLLS problem, whose solution depends on the performance of GN, such as computer vision [8], image alignment and reconstruction [9], [10], network-based localization [11], [12], signal processing for direction-of-arrival estimation and frequency estimation [13], logistic regression [14], and power system state estimation [5]. Despite the widespread utility, it is difficult to exploit the original GN method as a fully cooperative scheme for a distributed network, since its iteration rule involves the matrix inverse operator, which is ideally suited to be implemented in a centralized way. However, for the well-known advantages, such as load balancing and robustness, a distributed algorithm with the improvement of performance is preferred.

As the problem of minimizing a convex objective function, an NLLS problem can also be solved by most common unconstrained optimization methods, such as the first-order gradient descent [15] and second-order Newton's methods [16], both of which are studied as error feedback-based neural network learning methods [17], [18]. The biggest advantage of a gradient/subgradient descent method is its simplicity with low computational cost. However, the estimation accuracy and convergence rate are difficult to meet for many applications with high quality of service. Newton's method has a typically quadratical convergence rate, which is faster than the common linear gradient descent methods. However, the

heavy computation on the second-order Hessian restricts the applications of Newton's method in practice. Unlike Newton's method, the GN method only requires the knowledge of the first-order Jacobian without seriously compromising its convergence behavior. In an adaptive network, the diffusion version of the least mean square (LMS) method, referred to as stochastic gradient method, is proposed in [19]. The diffusion LMS method considers the gradient descent update using the real-time estimates from neighborhood, thus resulting in noticeable improvement. Such an intelligent scheme is absent in the study of most existing distributed optimization methods [20], which mainly focus on (sub)gradient-based primitive cooperation. Inspired by the diffusion LMS method, in this article, we attempt to develop a diffusion scheme for the GN method because of its above advantages.

As an attractive in-network data processing way, the diffusion strategy [21] differs from other distributed strategies, such as incremental cooperation [22], [23] and gossip cooperation [5], [24], which are based on the cyclic path and probabilistic diffusion, respectively. Without a fusion center (FC) as the centralized controller and any specialized exploration on routing as the incremental and the gossip ways, any peer-to-peer communication under the diffusion strategy can occur as long as they are linked. Such a way of communication makes it easier to achieve effective information sharing and cooperation via local interactions. The diffusion structure also shows the inherent advantages, including the robustness against failure of nodes and links, enhanced adaptation performance, and wider stability ranges [25]. Distributed estimation algorithms over the diffusion network have been proposed in the context of adaptive filtering, such as diffusion LMS [26], diffusion RLS [27], and diffusion Kalman filtering [28], [29].

The purpose of this work is to develop a diffusion GN strategy over an adaptive network, where every node senses the temporal data that are variable over the spatial domain and has the adaption ability by learning and cooperating. Several diffusion GN methods [12], [30] are proposed for solving the localization problem in wireless sensor networks. However, they are centralized in nature and implemented in a noncooperative way, in which the local intermediate estimates are not shared over the diffusion network.

Unlike the traditional diffusion GN methods, this article presents a novel cooperative diffusion strategy for GN method that can be used to solve the NLLS problems over an adaptive network. The motivation behind the cooperation scheme is that local estimates are exchanged among the neighboring nodes and fused into the local iteration step. In this case, the diversity of temporal–spatial data is fully utilized by local nodes. Such information exchange is potentially useful for the performance improvement for the noncooperative diffusion GN method. The resulting algorithm, named ATU, consists of aggregating and updating steps, where the aggregating step collects the real-time variations across the network based on the peer-to-peer diffusion protocol, such that the updating step can promptly respond to these variations. In short, the proposed algorithm has the advantages of distribution, cooperation, robustness, and fast convergence, as compared with the noncooperative diffusion algorithm. Our recent work [31] proposes a variant of ATU that implements in a first update and then aggregate way, namely, UTA. Under sufficient diffusion across the network, the UTA version achieves similar or slightly better performance to the ATU. However, the work [31] does not provide directly the detailed convergence analysis, thus resulting in the absence of such important theoretical results. More importantly, the performance of UTA degrades significantly in the case of sparse networks.

Our main contributions are twofold and summarized as follows.

1) The proposed strategy contributes significantly to solve the typical NLLS estimation problems over a diffusive networked system in a fully cooperative way between neighboring nodes. The resulting algorithm is simple but effective and provides better performance than the traditional methods with inadequate cooperation, where nodes exchange the GN descent information instead of immediate estimates. Such full cooperation provides the motivation for developing the new fusion modes based on cooperative iteration-type methods with other in-network processing, such as gossip and consensus, which have been widely studied in the distributed computing community.
2) Another important contribution of this article is to confirm theoretically and experimentally the convergence of the proposed diffusion algorithm and give the sufficient conditions for convergence in a global view. Furthermore, the cooperative benefit to achieve fast convergence is validated by investigating the spectral properties of the diffusion network based on a specified aggregate rule. We also provide several options used as the aggregate rule.

This article is organized as follows. In Section II, we describe the NLLS problems in a distributed network and introduce the traditional GN solution. In Section III, we derive the cooperative diffusion algorithm based on the aggregating then updating steps. Then convergence analysis with sufficient conditions of ATU follows in Section IV. Simulation results for target localization are provided in Section V. The whole paper is concluded in Section VI.

Notation: The operator $(\cdot)^{\mathrm{T}}$ denotes the transpose for a matrix or vector, and the operator $(\cdot)^{-1}$ denotes the inverse of a nonsingular matrix. Capital letters and small letters in bold are used when matrices and vectors are denoted, respectively, while scalars are denoted in the normal font. The 2-norm of a matrix $\mathbf{G}$ and vector $\mathbf{x}$ are denoted by $\|\mathbf{G}\|$ and $\|\mathbf{x}\|$, respectively. $\mathbf{I}_N$ and $\mathbf{1}_N$ denote the $N \times N$ identity matrix and $N \times 1$ vector, whose every entry is 1, respectively. $\otimes$ denotes the Kronecker product operation. We use subscripts k, l, u, and t to denote the node, and superscripts j and i to denote time.

II. Problem Formulation and the Centralized Gauss–Newton Solution

For an adaptive network represented by a set $\mathcal{N} = \{1, \ldots, N\}$, we would like to estimate an $M \times 1$ unknown

parameter vector $\mathbf{x} = [x_1, \ldots, x_M]^{\mathrm{T}}$ belonging to a closed convex set $\mathbb{X}$. Let $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_N(\mathbf{x})]^{\mathrm{T}} : \Re^M \to \Re^N$ be a continuous and differentiable global cost function throughout the network, where $f_k(\mathbf{x}) : \Re^M \to \Re$ is the individual cost function associated with node $k \in \mathcal{N}$ by collecting the measurements from the related events. The estimation problem can be formulated as

$$\min_{\mathbf{x}} \ \|\mathbf{f}(\mathbf{x})\|^2. \tag{1}$$

By rewriting $\|\mathbf{f}(\mathbf{x})\|^2 = \sum_{k=1}^{N} |f_k(\mathbf{x})|^2$, the object of each node in the network is to seek an $M \times 1$ vector $\mathbf{x}$ that solves the following NLLSs problem with the form:

$$\min_{\mathbf{x}} \ \sum_{k=1}^{N} |f_k(\mathbf{x})|^2. \tag{2}$$

The GN method is well recognized for solving NLLS problems. Let us consider an FC that can communicate with all nodes in the network. Given an initial good guess $\mathbf{x}^0$, a centralized scheme can be implemented on FC based on the GN update rule in an iterative way

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha^i \mathbf{d}^i \tag{3}$$

where $\mathbf{x}^i$ is the estimation of $\mathbf{x}$ at iteration i, $\mathbf{d}^i$ denotes a descent direction of GN, and α^i is the step size parameter that ensures $\mathbf{x}^{i+1}$ is nearer to a stationary point than $\mathbf{x}^i$.

In this article, we adopt the following assumption for the above optimization problem [16], [32].

Assumption 1:

1) The stationary points $\mathbf{x}^s \in \Re^M$ that satisfy the first-order condition

$$\mathbf{F}^{\mathrm{T}}(\mathbf{x}^s)\mathbf{f}(\mathbf{x}^s) = 0$$

always exist, where $\mathbf{F}(\mathbf{x})$ is the Jacobian of $\mathbf{f}(\mathbf{x})$ with the size $N \times M$ and the entries $\mathbf{F}(\mathbf{x})_{k,m} = \partial f_k(\mathbf{x})/\partial x_m$, $1 \le k \le N$ and $1 \le m \le M$.

2) For all $\mathbf{x} \in \mathbb{X}$ and $k \in \mathcal{N}$, let

$$\Sigma_{\min} = \min_{k \in \mathcal{N}} \left(\lambda_{\min}\left(\mathbf{F}^{\mathrm{T}}(\mathbf{x})\mathbf{F}(\mathbf{x})\right)\right)$$
$$\Sigma_{\max} = \max_{k \in \mathcal{N}} \left(\lambda_{\max}\left(\mathbf{F}^{\mathrm{T}}(\mathbf{x})\mathbf{F}(\mathbf{x})\right)\right)$$

where the notations $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimum and maximum eigenvalues, respectively, $0 < \Sigma_{\min} < \Sigma_{\max} < \infty$.

Under Assumption 1, a local minimizer of (1), denoted by $\mathbf{x}^*$ that belongs to the set of stationary points, always exists. Thus, the descent direction of GN update is written as

$$\mathbf{d}^i = \left[\mathbf{F}^{\mathrm{T}}(\mathbf{x}^i)\mathbf{F}(\mathbf{x}^i)\right]^{-1}\mathbf{F}^{\mathrm{T}}(\mathbf{x}^i)\mathbf{f}(\mathbf{x}^i). \tag{4}$$

By rewriting

$$\mathbf{F}(\mathbf{x}) = \mathrm{col}\left\{\frac{\partial f_1(\mathbf{x})}{\partial \mathbf{x}}, \frac{\partial f_2(\mathbf{x})}{\partial \mathbf{x}}, \ldots, \frac{\partial f_N(\mathbf{x})}{\partial \mathbf{x}}\right\} \quad (N \times M) \tag{5}$$

and defining

$$\mathbf{g}_k(\mathbf{x}) = \nabla f_k(\mathbf{x}) = \frac{\partial f_k(\mathbf{x})}{\partial \mathbf{x}}, \quad (1 \times M) \tag{6}$$

we get

$$\mathbf{d}^i = \left[\sum_{k=1}^{N} \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i)\mathbf{g}_k(\mathbf{x}^i)\right]^{-1} \sum_{k=1}^{N} \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i) f_k(\mathbf{x}^i). \tag{7}$$

Therefore, we have the following GN iteration update:

$$\mathbf{x}^{i+1} = \mathbf{x}^i - \alpha^i \left[\sum_{k=1}^{N} \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i)\mathbf{g}_k(\mathbf{x}^i)\right]^{-1} \sum_{k=1}^{N} \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i) f_k(\mathbf{x}^i). \tag{8}$$

To successfully implement (8) in a centralized way, we assume that the FC can communicate with all nodes over network and the same initial estimate is given by $\mathbf{x}_k^0 = \mathbf{x}^0, k \in \mathcal{N}$. In the centralized GN algorithm, the computation results of $\mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i)\mathbf{g}_k(\mathbf{x}^i)$ and $\mathbf{g}_k^{\mathrm{T}}(\mathbf{x}^i) f_k(\mathbf{x}^i)$ from each node k are aggregated by the FC to obtain the new estimate $\mathbf{x}^{i+1}$ based on (8). Then, the estimate $\mathbf{x}^{i+1}$ is returned to all nodes until an appropriate termination condition is satisfied, for example, $\|\mathbf{x}^{i+1} - \mathbf{x}^i\| \le \varepsilon$ or $i = I$, where ε and I are the predefined minimum norm decline and the maximum allowed number of iterations, respectively. Thus, the centralized GN includes actually a step of diffusion for new estimate $\mathbf{x}^{i+1}$ from FC to individual nodes.

Note that the centralized GN algorithm for NLLS problems is not guaranteed to converge to the local minimizer $\mathbf{x}^*$. For this, two preconditions need to be satisfied. First, we require that the initial guess for $\mathbf{x}^0$ is close to a local minimizer $\mathbf{x}^*$. Otherwise, $\mathbf{F}^{\mathrm{T}}(\mathbf{x})\mathbf{F}(\mathbf{x})$ is not a sufficiently good approximation to the Hessian of $f(\mathbf{x})$, thereby resulting in GN to give unexpected results. Second, the step size α^i is required to satisfy the Wolfe conditions [16], which guarantee the update $\mathbf{x}^{i+1}$ to be nearer $\mathbf{x}^*$ than $\mathbf{x}^i$. The step size denoted by α_k^i is usually selected by node k at different time i. The resulting algorithm is called the damped GN algorithm. One can easily design a line search method to find any guessed step size satisfying the following Wolfe conditions:

$$\left\|\mathbf{f}\left(\mathbf{x}^{i+1}\right)\right\|^2 \le \left\|\mathbf{f}(\mathbf{x}^i)\right\|^2 - b_1 \alpha^i \mathbf{f}^{\mathrm{T}}(\mathbf{x}^i)\mathbf{F}(\mathbf{x}^i)\mathbf{d}^i \tag{9}$$

and

$$\mathbf{f}^{\mathrm{T}}\left(\mathbf{x}^{i+1}\right)\mathbf{F}\left(\mathbf{x}^{i+1}\right)\mathbf{d}^i \ge b_2 \alpha^i \mathbf{f}^{\mathrm{T}}(\mathbf{x}^i)\mathbf{F}(\mathbf{x}^i)\mathbf{d}^i \tag{10}$$

with $0 < b_1 < b_2 < 1$. The first of the Wolfe conditions (9) guarantees that the step length α^i decreases the objective function $\|\mathbf{f}(\mathbf{x})\|^2$ at every iteration, while the second (10) tests whether the descent is sufficient. In this article, to simplify the mathematical discussion, the constant step size scheme $\alpha_k^i = \alpha \in (0, 1]$ will be adopted for every node k at any iteration i.

III. ATU Diffusion Gauss–Newton

In the centralized GN algorithm, the FC only exchanges information with its immediate neighborhood. However, for a large-scale network, it is difficult that all nodes in the network can communicate with the FC via 1-hop. For example, each node in wireless sensor networks [33] has the restricted communication capacity and energy supply. On the other hand, multihop communication techniques involve the complicated

routing protocols, which need to be well designed. Thus, peer-to-peer communication is preferred for the large-scale network.

For networked control systems [34], a distributed control implementation depends largely on neighborhood cooperation [35]. Our object is to develop a diffusion strategy for the GN algorithm, in which each node in the network can obtain the good estimate by combining the estimates from peer-to-peer neighbors. The idea behind this strategy is that the local node can leverage the global knowledge to seek the better estimate, since every node carries the information from different neighbors.

Consider the adaptive network $\mathcal{N}$, where any node k at time i receives a set of estimates $\{\mathbf{x}_l^i\}_{l\in\mathcal{N}_k}$ from all its 1-hop neighbors $\mathcal{N}_k$ including itself. Consequently, node k can aggregate the local estimates $\{\mathbf{x}_l^i\}_{l\in\mathcal{N}_k}$ by a weighted combination way denoted by

$$\mathbf{y}_k^i = \sum_{l\in\mathcal{N}_k} c_{kl}\mathbf{x}_l^i \tag{11}$$

where c_{kl} is the weighted coefficient between node k and $l \in \mathcal{N}_k$. And the conditions

$$\sum_{l\in\mathcal{N}_k} c_{kl} = 1 \ \text{ and } \ c_{kl} \in [0,1] \ \text{ for } \ l \in \mathcal{N}_k \tag{12}$$

hold.

The role of the above aggregation step is obvious, i.e., any node in the network can fuse the updated information from all nodes across the network, not just the neighborhoods, since every node tends to have the different neighbors for connected network. The latest estimates are passed by all neighbored nodes, thereby resulting in the good diffusion effect. Moreover, for a large-scale distributed network, every node can give an estimate based on the diffusion knowledge, while only FC does it in a centralized network. That is particularly useful for a distributed network, where the estimates can be accessed from any individual node over the network.

Once the aggregate estimate $\mathbf{y}_k^i$ is obtained as the local weighted estimate, any node k in the network can implement the GN update step when it is regarded as an FC in its neighborhood $\mathcal{N}_k$ as (8)

$$\mathbf{x}_k^{i+1} = \mathbf{y}_k^i - \alpha\big[\mathbf{Q}_k^i(\mathbf{y})\big]^{-1}\mathbf{q}_k^i(\mathbf{y}) \tag{13}$$

where we define

$$\begin{aligned}\mathbf{Q}_k^i(\mathbf{y}) &\triangleq \mathbf{F}_{l\in\mathcal{N}_k}^{\mathrm{T}}(\mathbf{y}_l^i)\mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{y}_l^i)\\ &\triangleq \sum_{l\in\mathcal{N}_k}\mathbf{g}_l^{\mathrm{T}}(\mathbf{y}_l^i)\mathbf{g}_l\big(\mathbf{y}_l^i\big) \quad (M\times M)\end{aligned} \tag{14}$$

$$\begin{aligned}\mathbf{q}_k^i(\mathbf{y}) &\triangleq \mathbf{F}_{l\in\mathcal{N}_k}^{\mathrm{T}}(\mathbf{y}_l^i)\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{y}_l^i)\\ &\triangleq \sum_{l\in\mathcal{N}_k}\mathbf{g}_l^{\mathrm{T}}(\mathbf{y}_l^i)f_l(\mathbf{y}_l^i) \quad (M\times 1)\end{aligned} \tag{15}$$

and $\mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{y}_l^i)$ is a matrix of size $n_k \times M$, each of whose rows is an $1\times M$ vector $\mathbf{g}_l(\mathbf{y}_l^i) = ([\partial f_l(\mathbf{x})]/[\partial\mathbf{x}])\,|_{\mathbf{x}=\mathbf{y}_l^i}$ for $l \in \mathcal{N}_k$, while $\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{y}_l^i)$ is a vector of size M with the entry $f_l(\mathbf{y}_l^i)$ for $l \in \mathcal{N}_k$.

Algorithm 1 ATU Diffusion GN Algorithm

1: Given the initial guesses $\mathbf{x}^0$, the accuracy ε, the maximum number of iterations I and the coefficients $\{c_{kl}\}$ satisfying (12) for all nodes
2: **for** each iteration $i = 0$ to I **do**
3: **for** each node $k = 1$ to N **do**
4: receive $\{\mathbf{x}_l^i\}$ from the immediate neighbor l of node k
5: **aggregate step**: $\mathbf{y}_k^i = \sum_{l\in\mathcal{N}_k} c_{kl}\mathbf{x}_l^i$
6: send $\{\mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)\mathbf{g}_k(\mathbf{y}_k^i), \mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)f_k(\mathbf{y}_k^i)\}$ to the immediate neighbor l
7: **update step**: $\mathbf{x}_k^{i+1} = \mathbf{y}_k^i - \alpha[\mathbf{Q}_k^i(\mathbf{y})]^{-1}\mathbf{q}_k^i(\mathbf{y})$
8: **if** $\|\mathbf{x}_k^{i+1} - \mathbf{x}_k^i\| < \varepsilon$ or $i \geq I$ **then**
9: return $\mathbf{x}_k^{i+1}$
10: **else**
11: send $\mathbf{x}_k^{i+1}$ to the immediate neighbour l
12: **end if**
13: **end for**
14: **end for**

Combining the above steps, we obtain the ATU diffusion algorithm.

In Algorithm 1, the norm $\|\mathbf{x}_k^{i+1} - \mathbf{x}_k^i\|$ is a very good indicator of the error, which is used to substitute the descent degree of the cost function between two successive iterations. When successive estimation values are close, it is safe to terminate the iteration for a descent-based iterative method. Such a termination criterion has been proved to be suitable for accurately measuring the error [16]. For the condition $i \geq I$, in real problems, the number of allowed iterations is related to the admissible computation time determined by the complexity of a different cost function. Thus, such a termination condition is application-dependent and serves mainly as a stopping rule of the algorithm to avoid infinite running time.

Fig. 1 shows the iterative procedure of implementing the ATU algorithm on every node, which consists of three steps. First, every node sends its current estimate to all its neighbors. Initially, all nodes are assigned with the same estimate $\mathbf{x}^0$. The received estimates are aggregated by every node based on a linear combiner (11). Specifically, at iteration i, node k obtains the aggregated estimate $\mathbf{y}_k^i$. Second, every node k returns the current computation of $\{\mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)\mathbf{g}_k(\mathbf{y}_k^i), \mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)f_k(\mathbf{y}_k^i)\}$ to its neighbors. Finally, the GN update (13) is implemented on every node. It is worth noting that the order of implementing aggregate step and update step can be reversed, which results in a new algorithm, named UTA. In our recent work [31], we provide the performance comparisons in detail between ATU and UTA, as well as their variants.

Deleting the aggregate step of the ATU algorithm (the fifth line in Algorithm 1) and substituting $\{\mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)\mathbf{g}_k(\mathbf{y}_k^i), \mathbf{g}_k^{\mathrm{T}}(\mathbf{y}_k^i)f_k(\mathbf{y}_k^i)\}$ with $\{\mathbf{g}_k^{\mathrm{T}}(\mathbf{x}_l^i)\mathbf{g}_k(\mathbf{x}_l^i), \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}_l^i)f_k(\mathbf{x}_l^i)\}$ in the sixth line, we obtain a noncooperative diffusion GN algorithm, where each node in the network acts as the FC to implement the centralized GN by communicating with all immediate neighbors. Its GN update step is given by

$$\mathbf{x}_k^{i+1} = \mathbf{x}_k^i - \alpha\big[\mathbf{Q}_k^i\big(\mathbf{x}_k^i\big)\big]^{-1}\mathbf{q}_k^i\big(\mathbf{x}_k^i\big) \tag{16}$$

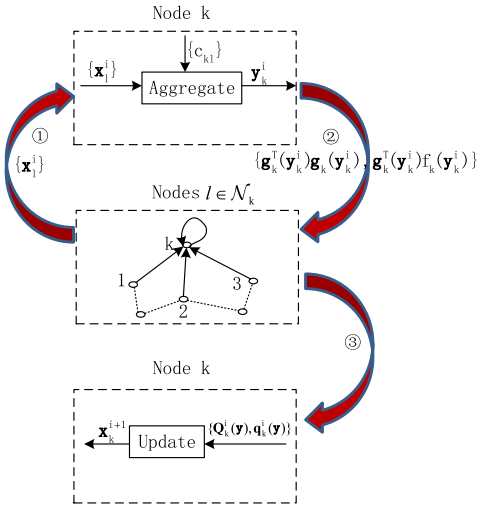


Fig. 1. ATU diffusion GN.

where we define

$$\mathbf{Q}_k^i(\mathbf{x}_k^i) \triangleq \mathbf{F}_{l\in\mathcal{N}_k}^{\mathrm{T}}(\mathbf{x}_k^i)\mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{x}_k^i) \triangleq \sum_{l\in\mathcal{N}_k} \mathbf{g}_l^{\mathrm{T}}(\mathbf{x}_k^i)\mathbf{g}_l(\mathbf{x}_k^i) \quad (M\times M) \tag{17}$$

and

$$\mathbf{q}_k^i(\mathbf{x}_k^i) \triangleq \mathbf{F}_{l\in\mathcal{N}_k}^{\mathrm{T}}(\mathbf{x}_k^i)\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x}_k^i) \triangleq \sum_{l\in\mathcal{N}_k} \mathbf{g}_l^{\mathrm{T}}(\mathbf{x}_k^i) f_l(\mathbf{x}_k^i) \quad (M\times 1). \tag{18}$$

Note that the arguments in expressions (14), (15), (17), and (18) show the main difference between ATU and noncooperative algorithms.

In order for the new algorithm to be applicable, it is important to note that the square matrix $\mathbf{Q}_k^i(\mathbf{y})$ needs to be invertible during iterations. However, it is impractical for a dynamic system. Two methods can be considered to address the problem. One is to add a small multiple ℓ_k of the identity matrix $\mathbf{I}$ to $\mathbf{Q}_k^i(\mathbf{y})$, such that $\mathbf{Q}_k^i(\mathbf{y}) + \ell_k\mathbf{I}$ is positive definite, which is called the Levenberg−Marquardt method [16]. This method has a difficulty on determining a good value for $\ell_k > 0$. The other is to compute the unique Moore−Penrose inverse, which gives a good approximate solution to the inversion of the matrix by using the singular value decomposition [36]. For simplicity, we assume that $\mathbf{Q}_k^i(\mathbf{y})$ is invertible in later discussion.

It is worth noting that our proposed ATU algorithm does not add any additional communication cost compared with the noncooperative GN algorithm. That is, the communication amount per iteration is $2n_kM + n_kM^2$ for ATU, UTA [31], as well as noncooperative GN, where n_k is the number of 1-hop neighbors of any node k. On the computation cost, there is a slight increase of n_kM multiplication and n_kM addition scalar operations on the aggregate step for every iteration. The total complexity for ATU or UTA is order of $O(M^3)$ since inversion operation of an invertible matrix with size $M\times M$ is inevitable by using the well-known Gauss–Jordan elimination.

So, the question that remains is how well the ATU diffusion GN algorithm performs in term of its expected convergence behavior, since the diffuse cooperation also fosters uncertainty. First, what are the sufficient conditions of convergence for the ATU algorithm? Second, is the ATU algorithm better on convergence, compared with its noncooperative counterpart? In other words, what are the benefits of cooperation? The following analysis and simulations will answer the above questions.

IV. Convergence Analysis

A. Assumptions and Data Model

To proceed with the analysis, several reasonable assumptions need to be given as commonly done in [5] and [37].

Assumption 2:

1) $\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x})$ is bounded for all $\mathbf{x}\in\mathbb{X}\subset\mathbb{R}^M$ near $\mathbf{x}^*$, and satisfies
$$\left\|\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x})\right\| \le e_{\max}$$
and
$$\left\|\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x}^*)\right\| = e_{\min}$$
where $\|\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x}^*)\|$ denotes the minimum value of $\|\mathbf{f}_{l\in\mathcal{N}_k}(\mathbf{x})\|$ when evaluated at $\mathbf{x}=\mathbf{x}^*$.
2) For all $\mathbf{x}\in\mathbb{X}$ and $k=1,\ldots,N$, let
$$\sigma_{\min} = \min_{k\in\mathcal{N}}\left(\lambda_{\min}\left(\mathbf{g}_k^{\mathrm{T}}(\mathbf{x})\mathbf{g}_k(\mathbf{x})\right)\right)$$
and
$$\sigma_{\max} = \max_{k\in\mathcal{N}}\left(\lambda_{\max}\left(\mathbf{g}_k^{\mathrm{T}}(\mathbf{x})\mathbf{g}_k(\mathbf{x})\right)\right)$$
where $0<\sigma_{\min}<\sigma_{\max}<\infty$.
3) Both $\mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{x})$ and $\mathbf{g}_k(\mathbf{x})$ are Lipschitz continuous on $\mathbb{X}$ with Lipschitz constant $\omega>0$ such that
$$\left\|\mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{x}) - \mathbf{F}_{l\in\mathcal{N}_k}(\mathbf{x}')\right\| \le \omega\left\|\mathbf{x}-\mathbf{x}'\right\|$$
and
$$\left\|\mathbf{g}_k(\mathbf{x}) - \mathbf{g}_k(\mathbf{x}')\right\| \le \omega\left\|\mathbf{x}-\mathbf{x}'\right\|$$
for all $\mathbf{x},\mathbf{x}'\in\mathbb{X}$. Furthermore, we have the following results [38]:
$$\left\|\mathbf{g}_k^{\mathrm{T}}(\mathbf{x})f_k(\mathbf{x}) - \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}')f_k(\mathbf{x}')\right\| \le \gamma_f\left\|\mathbf{x}-\mathbf{x}'\right\|$$
and
$$\left\|\mathbf{g}_k^{\mathrm{T}}(\mathbf{x})\mathbf{g}_k(\mathbf{x}) - \mathbf{g}_k^{\mathrm{T}}(\mathbf{x}')\mathbf{g}_k(\mathbf{x}')\right\| \le \gamma_F\left\|\mathbf{x}-\mathbf{x}'\right\|$$
where $\gamma_f \ge \omega(e_{\max}+\Sigma_{\max})$ and $\gamma_F \ge 2\Sigma_{\max}\omega$ are the corresponding Lipschitz constants.

In addition, the study of the local convergence behavior needs to be considered from the global view of network, since the performance of individual node depends on the whole network, including cooperation rule and network topology. Thus, we introduce the global quantities

$$\mathbf{x}_G^i \triangleq \mathrm{col}\left\{\mathbf{x}_1^i,\ldots,\mathbf{x}_N^i\right\}, \quad (NM\times 1)$$
$$\mathbf{y}_G^i \triangleq \mathrm{col}\left\{\mathbf{y}_1^i,\ldots,\mathbf{y}_N^i\right\}, \quad (NM\times 1)$$
$$\overline{\mathbf{x}}^* \triangleq \mathrm{col}\left\{\mathbf{x}^*,\ldots,\mathbf{x}^*\right\}, \quad (NM\times 1)$$

$$\mathbf{d}_G^i(\mathbf{y}) \triangleq \mathrm{col}\{\mathbf{d}_1^i(\mathbf{y}), \ldots, \mathbf{d}_N^i(\mathbf{y})\}, \quad (NM \times 1)$$
$$\mathbf{d}_G^i \triangleq \mathrm{col}\{\mathbf{d}_1^i, \ldots, \mathbf{d}_N^i\}, \quad (NM \times 1)$$

where

$$\mathbf{d}_k^i(\mathbf{y}) \triangleq \left[\mathbf{Q}_k^i(\mathbf{y})\right]^{-1}\mathbf{q}_k^i(\mathbf{y}), \quad k \in \mathcal{N}$$

and

$$\mathbf{d}_k^i \triangleq \left[\mathbf{Q}_k^i(\mathbf{x}_k^i)\right]^{-1}\mathbf{q}_k^i(\mathbf{x}_k^i), \quad k \in \mathcal{N}.$$

An $N \times N$ aggregation matrix $\mathbf{C}$ can be given with nonnegative real entries $\{c_{kl}\}$ that is redefined with the following conditions:

$$c_{kl} = 0 \ \text{ if } \ l \notin \mathcal{N}_k \ \text{ and } \ \sum_{l=1}^{N} c_{kl} = 1, c_{kl} \geq 0. \tag{19}$$

The conditions (19) indicate that the sum of all entries on each row of the matrix $\mathbf{C}$ is one, while the entry c_{kl} of $\mathbf{C}$ shows the degree of closeness between nodes k and l. We will see the influence of selecting $\{c_{kl}\}$ on the performance of the resulting algorithms in later simulations.

Similarly, we introduce an $N \times N$ adjacency matrix $\mathbf{\Phi}$ with the element $\varphi_{kl} \in \{0, 1\}$, in which $\varphi_{kl} = 1$ if node k is linked with node l; otherwise, 0.

We also introduce the extended aggregation matrix $\mathbf{G}$

$$\mathbf{G} \triangleq \mathbf{C} \otimes \mathbf{I}_M, \quad (NM \times NM).$$

B. Main Theoretical Results

In this section, we list the main theoretical results, while the detailed proof is placed in the associated supplementary material. We establish a temporal–spatial relation across network based on the following considerations. First, the diffusion strategy leads to the frequent spatial interaction between the neighborhoods, thereby each node k is influenced by both local information, such as f_k and spatial information from neighbors $l \in \mathcal{N}_k$ such as $\{f_l, \mathbf{x}_l\}$. Second, the iteration way decides that the estimates and the local collected information on each node k are time variant, i.e., $\{f_k^i, \mathbf{x}_k^i\}$.

To begin with (11), we have

$$\mathbf{y}_G^i = \mathbf{G}\mathbf{x}_G^i. \tag{20}$$

Using (20), we rewrite the local ATU update step (13) as a global representation

$$\mathbf{x}_G^{i+1} = \mathbf{G}\mathbf{x}_G^i - \alpha\mathbf{d}_G^i(\mathbf{y}). \tag{21}$$

Accordingly, we get the global noncooperative GN update step

$$\mathbf{x}_G^{i+1} = \mathbf{x}_G^i - \alpha\mathbf{d}_G^i. \tag{22}$$

Subtracting $\overline{\mathbf{x}}^*$ on both sides of (21) and embedding (22), we get

$$\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^* = \left(\mathbf{G}\mathbf{x}_G^i - \overline{\mathbf{x}}^* - \alpha\mathbf{d}_G^i\right) + \alpha\left(\mathbf{d}_G^i - \mathbf{d}_G^i(\mathbf{y})\right). \tag{23}$$

Using the triangle inequality for vectors, we get the following recursion:

$$\left\|\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^*\right\| \leq \left\|\mathbf{G}\mathbf{x}_G^i - \overline{\mathbf{x}}^* - \alpha\mathbf{d}_G^i\right\| + \alpha\left\|\mathbf{d}_G^i(\mathbf{y}) - \mathbf{d}_G^i\right\|. \tag{24}$$

Inequality (24) can be regarded as a temporal–spatial recursion relation, where the superscript i and the subscript G reflect the evolution of ATU algorithm from temporal and spatial dimensions, respectively. We establish the relation between ATU and noncooperative diffusion algorithms from the global perspective.

Consequently, we can obtain a new temporal–spatial recursion relation as lemma 1.

Lemma 1: Let Assumptions 1 and 2 hold. The recursion relation

$$\left\|\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^*\right\| \leq t_1\|\mathbf{G}\|^2\left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\|^2 + t_2\|\mathbf{G}\|\left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\| + \alpha\left\|\mathbf{d}_G^i(\mathbf{y}) - \mathbf{d}_G^i\right\| \tag{25}$$

holds, where

$$t_1 \triangleq \frac{\alpha\omega}{2\sqrt{\Sigma_{\min}}}, t_2 \triangleq \frac{(1-\alpha)\sqrt{\Sigma_{\max}}}{\sqrt{\Sigma_{\min}}} + \frac{\sqrt{2}N\alpha\omega e_{\min}}{\Sigma_{\min}}. \tag{26}$$

Proof: See Section I of the supplementary material. ■

Note that the left-hand side of (25) is the network-wide error norm at time $i+1$, while the right-hand side of (25) can be related to the network-wide error norm $\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\|$ at time i and we can confirm that $\|\mathbf{d}_G^i(\mathbf{y}) - \mathbf{d}_G^i\|$ is upper bounded by a given constant $\xi > 0$, that is,

Lemma 2: Let Assumptions 1 and 2 hold. The norm of the descent discrepancy $\mathbf{d}_G^i(\mathbf{y}) - \mathbf{d}_G^i$ is upper bounded as

$$\left\|\mathbf{d}_G^i(\mathbf{y}) - \mathbf{d}_G^i\right\| \leq \xi \triangleq \frac{N^2\gamma_f\Pi^i}{\sigma_{\min}} + \frac{\left(N\sqrt{\sigma_{\max}}\varepsilon_{\max} + N^2\gamma_f\Pi^i\right)\zeta_i}{\sigma_{\min}(1-\zeta_i)} \tag{27}$$

where

$$\Pi^i \triangleq a_2\sum_{j=1}^{i}(a_1)^{j-1} \tag{28}$$

$$a_1 \triangleq 1 + \frac{\alpha n_{kl} + 2\alpha n_{kl}\gamma_f}{2n_l\sigma_{\min}} \tag{29}$$

$$a_2 \triangleq \frac{\left(n_l + 3n_{k|l} + 3n_{l|k}\right)\alpha\sqrt{\sigma_{\max}}\varepsilon_{\max}}{2n_l\sigma_{\min}} \tag{30}$$

$$\zeta_i \triangleq \frac{N\gamma_F\Pi^i}{\sigma_{\min}} \in (0, 1). \tag{31}$$

n_{kl} denotes the number of nodes that are both in $\mathcal{N}_k$ and $\mathcal{N}_l$, $n_{k|l}$ denotes the number of nodes that are in $\mathcal{N}_k$ and not in $\mathcal{N}_l$.

Proof: See Section II in the supplementary material. ■

Then, we can rewrite the relation (25) of the network error between the successive two times for our ATU algorithm as

$$\left\|\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^*\right\| \leq t_1\|\mathbf{G}\|^2\left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\|^2 + t_2\|\mathbf{G}\|\left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\| + \alpha\xi \tag{32}$$

which can be regarded as a nonlinear discrete dynamical system.

Let $y^i \triangleq \|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\|$, we will simplify notation of (32) with the general form

$$y^{i+1} \leq t_1\|\mathbf{G}\|^2(y^i)^2 + t_2\|\mathbf{G}\|y^i + \alpha\xi \tag{33}$$

whose steady-state equilibrium is a level [39] that solves

$$y = \phi(y) = t_1\|\mathbf{G}\|^2 y^2 + t_2\|\mathbf{G}\|y + \alpha\xi. \tag{34}$$

Note that the steady-state equilibrium means that the variable y^i is invariant under the law of motion indicated by the dynamical system. With the expression (34), it is easy to know that the recursion (33) is governed by the dynamical system $y^{i+1} = \phi(y^i)$. Thus, guaranteeing the stability of the system $y^{i+1} = \phi(y^i)$ will be needed.

By applying the steady-state equilibrium theory [39] to analyze the dynamical system (32), we obtain its steady-state equilibrium point in the following theorem.

Theorem 1: Let Assumptions 1 and 2 hold. Under the following conditions:

$$\frac{1}{t_2 + 2\sqrt{t_1\alpha\xi}} < \|\mathbf{G}\| < \frac{t_2 + 2\sqrt{t_2^2 - t_1\alpha\xi}}{t_2^2 - 4t_1\alpha\xi} \tag{35}$$

and

$$\max\left\{\frac{t_2^2\|\mathbf{G}\|^2 - 2t_2\|\mathbf{G}\| - 3}{4t_1\xi\|\mathbf{G}\|^2}, 0\right\} < \alpha < \min\left\{\frac{t_2^2\|\mathbf{G}\|^2 - 2t_2\|\mathbf{G}\| + 1}{4t_1\xi\|\mathbf{G}\|^2}, 1\right\} \tag{36}$$

as long as the initial condition y^0 is smaller than $y_{\max}$, the nonlinear system (33) converges to the unique steady-state equilibrium point $y_{\min}$, where

$$y_{\max} = \frac{1 - t_2\|\mathbf{G}\| + \sqrt{(1 - t_2\|\mathbf{G}\|)^2 - 4t_1\alpha\xi\|\mathbf{G}\|^2}}{2t_1\|\mathbf{G}\|^2} \tag{37}$$

and

$$y_{\min} = \frac{1 - t_2\|\mathbf{G}\| - \sqrt{(1 - t_2\|\mathbf{G}\|)^2 - 4t_1\alpha\xi\|\mathbf{G}\|^2}}{2t_1\|\mathbf{G}\|^2}. \tag{38}$$

That is

$$\lim_{i\to\infty} y^i = y_{\min}, \quad \text{if } y_0 < y_{\max} \tag{39}$$

or equivalently

$$\lim_{i\to\infty} \left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\| = y_{\min}, \quad \text{if } \|\mathbf{x}_G^0 - \overline{\mathbf{x}}^*\| < y_{\max}. \tag{40}$$

Proof: See Section VI in the supplementary material. ■

Based on Theorem 1, the ATU algorithm converges asymptotically to the minimizer $\mathbf{x}^*$ if the initial global error $\|\mathbf{x}_G^0 - \overline{\mathbf{x}}^*\| < y_{\max}$ holds. Conversely, the initial condition $\|\mathbf{x}_G^0 - \overline{\mathbf{x}}^*\| > y_{\max}$ will lead to instability of algorithm and growing global error level. This sufficient condition suggests that the convergence of proposed algorithm depends on the good initial iterates, which is also required by the traditional GN method to give good estimates. When our algorithm is applied to problems with unfavorable initial guess and many local minimums, the selected step size should not be too large to guarantee the convergence and avoid missing the nearest local minimum. The Wolfe conditions are also applicable for this case.

C. Comparison of Convergence Behaviors

In this section, we try to provide a qualitative analysis of convergence behaviors for ATU and noncooperative algorithms. For this purpose, error recursions for our ATU algorithm and noncooperative algorithm can be described as follows:

$$\left\|\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^*\right\| \le \left(1 + \alpha\gamma_f\|\mathbf{\Lambda}_D\|\|\mathbf{\Omega}\|\right)\left\|\mathbf{G}\left(\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right)\right\| \tag{41}$$

and

$$\left\|\mathbf{x}_G^{i+1} - \overline{\mathbf{x}}^*\right\| \le \left(1 + \alpha\gamma_f\|\mathbf{\Lambda}_d\|\|\mathbf{\Omega}\|\right)\left\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\right\| \tag{42}$$

whose proof can be found in Section VII in the supplementary material.

From (6), we know that $\mathbf{y}_k^i$ is a convex combination of $\{\mathbf{x}_l^i\}$ for $l \in \mathcal{N}_k$. Thus, Assumption 1 holds for $\mathbf{y}_k^i$. Under Assumption 1(1), we have

$$\left\|\left[\mathbf{Q}_k^i(\mathbf{y})\right]^{-1}\right\| = \left\|\left[\sum_{l\in\mathcal{N}_k} \mathbf{g}_l^{\mathrm{T}}\left(\mathbf{y}_l^i\right)\mathbf{g}_l\left(\mathbf{y}_l^i\right)\right]^{-1}\right\| \le \frac{1}{n_k\sigma_{\min}} \tag{43}$$

and

$$\left\|\left[\mathbf{Q}_k^i\left(\mathbf{x}_k^i\right)\right]^{-1}\right\| \le \frac{1}{n_k\sigma_{\min}}. \tag{44}$$

Furthermore, we have

$$\|\mathbf{\Lambda}_D\| \le \frac{1}{n_k\sigma_{\min}} \tag{45}$$

and

$$\|\mathbf{\Lambda}_d\| \le \frac{1}{n_k\sigma_{\min}}. \tag{46}$$

Thus, we know that $1+\alpha\gamma_f\|\mathbf{\Lambda}_D\|\|\mathbf{\Omega}\|$ and $1+\alpha\gamma_f\|\mathbf{\Lambda}_d\|\|\mathbf{\Omega}\|$ are upper bounded by a small common constant when the small step size is selected.

The recursions (41) and (42) describe how the global error evolves over time for diffusion and noncooperative GN algorithms, respectively. It is important to note the difference between the linear structure (41) and the nonlinear structure (32). If we replace the lesser-or-equal with an equal sign in (41), the resulted linear system will be unstable due to $(1 + \alpha\gamma_f\|\mathbf{\Lambda}_D\|\|\mathbf{\Omega}\|)\|\mathbf{G}\| > 1$ if $\|\mathbf{G}\| \ge 1$ [39]. However, under guaranteed convergence conditions for diffusion ATU, (41) and (42) reveal the qualitative behavior of global error reduction in ATU and noncooperative GN algorithms, respectively.

To analyze the convergence behavior of the diffusion GN algorithm, we introduce the spectral radius of a square matrix [40], which is defined as the largest absolute value among its eigenvalues and denoted by $\rho(\cdot)$. Because of $\mathbf{G} = \mathbf{C}\otimes\mathbf{I}_M$, we have

$$\begin{aligned}\rho(\mathbf{G}) &= |\lambda_{\max}(\mathbf{G})| = |\lambda_{\max}(\mathbf{C}\otimes\mathbf{I}_M)| \\ &\le |\lambda_{\max}(\mathbf{C})||\lambda_{\max}(\mathbf{I}_M)| = |\lambda_{\max}(\mathbf{C})| = 1\end{aligned} \tag{47}$$

where $|\lambda_{\max}(\mathbf{C})| = 1$ is based on the known conclusion [41, Appendix C] if $\mathbf{C}$ satisfies (19). Thus, (47) indicates that all eigenvalues of $\mathbf{G}$ are smaller than 1, i.e., $|\lambda(\mathbf{G})| \le 1$.

Whenever we select an aggregation matrix $\mathbf{C}$ based on (19) so that $|\lambda(\mathbf{C})| \leq 1$, the spectral radius of $\mathbf{G}$ representing cooperative diffusion case is generally smaller than the spectral radius of $\mathbf{I}_N$ representing the noncooperation case. That is, the cooperative diffusion GN algorithm will enforce a reduction of error $\mathbf{x}_G^i - \overline{\mathbf{x}}^*$ over the noncooperative version at every iteration. In other words, the error norm $\|\mathbf{x}_G^i - \overline{\mathbf{x}}^*\|$ in the cooperation strategy decays more rapidly than that in noncooperation strategy.

The above analysis confirms the role of diffusion step in GN algorithm for improvements on convergence rate. In the following section, we will examine the profiles of $\|\mathbf{G}\|$ and $\rho(\mathbf{G})$ by investigating numerically their values.

V. Application to Node Localization Model

In order to assess the proposed algorithm performance, we apply the ATU diffusion to the example of node localization [42], which has become the important foundation of some applications in reality. Examples include target tracking [43], traffic monitoring [44], sensing coverage [45], and so on. These applications need to determine the location of objects without any specialized positioning devices, such as GPS. By collecting the information from the decentralized nodes located in a monitoring area, node localization algorithms aim to report the position estimate quickly, while maintaining accuracy and stability.

The typical node localization can be modeled as a NLLS problem [11], [30], which is described by using previous same notations as follows. We assume the position of target node is an $M \times 1$ unknown vector $\mathbf{x} = [x_1, \ldots, x_M]^{\mathrm{T}}$, where $M = 2$ for two-dimensional (2-D) plane or $M = 3$ for three-dimensional space. In a monitoring network, such as the wireless sensor network, which is denoted by $\mathcal{N} = \{1, \ldots, N\}$, beacon node (BN) $k \in \mathcal{N}$ can communicate with its immediate neighbors $\mathcal{N}_k$ within its communication range (CR), and is aware of its own position that is not known for other nodes. Thus, for the target localization problem, the objective would be to minimize the following cost function:

$$\begin{aligned} \|\mathbf{f}(\mathbf{x})\|^2 &= \sum_{k=1}^{N} |f_k(\mathbf{x})|^2 \\ &= \sum_{k=1}^{N} \left| \mathbf{x}^{\mathrm{T}}\mathbf{x} - 2\mathbf{h}_k^{\mathrm{T}}\mathbf{x} + \|\mathbf{h}_k\|^2 - |r_k|^2 \right|^2 \end{aligned} \tag{48}$$

where $\mathbf{h}_k$ is the $M \times 1$ coordinate of BN k, r_k is the measured distance between BN k and the target, and $f_k(\mathbf{x})$ denotes the error cost function on individual BN k. Generally, the measurement r_k^i at time i is a noisy version of the real distance R_k^i and can be modeled as

$$r_k^i = R_k^i \left(1 + \mathrm{noise}_k^i\right) \tag{49}$$

where noise_k^i is the additive ambient Gaussian noise with zero-mean and finite variance v_k^2.

Fig. 2 shows an illustration of adaptive network, where 20 BNs are deployed in a 50×50 m^2 2-D surveillance area and and the target is located at the center of area. For the sake of fairness, the positions of BNs are generated randomly in each simulation, while the target node is located stationarily in the center point of area. Given the parameter settings that the CR is 25 m between any two nodes, nodes in the network implement cooperatively the ATU diffusion algorithm to estimate the target position. The network-wide average error for all nodes will be used as the main performance metric, which is defined by

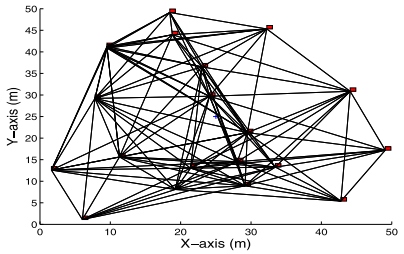


Fig. 2. Illustration of adaptive network in a random deployment, where the red rectangle denotes the target and the blue + denotes the measuring node.

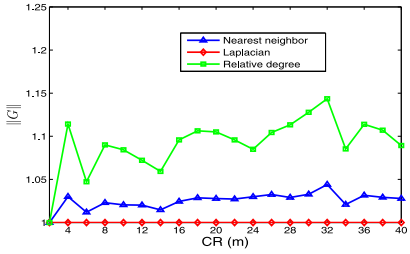


Fig. 3. Profile of $\|\mathbf{G}\|$ under different aggregate rules and radio ranges.

$$\mathrm{Error}_{\mathrm{network}}^i \triangleq \frac{1}{N} \sum_{k=1}^{N} \left(\left\| \mathbf{x}_k^i - \mathbf{x}^* \right\| \right). \tag{50}$$

A. Alternatives for Aggregate Rule

Many alternative aggregate rules can be easily obtained for our ATU algorithm to generate the aggregation matrix $\mathbf{C}$, including the well-known nearest neighbor rule, the Laplacian rule, and the relative degree rule [31].

Obviously, both nearest neighbor and Laplacian rule follow condition (19), such that the extended aggregation matrix $\mathbf{G}$ can be generated. Fig. 3 gives the profile of $\|\mathbf{G}\|$ against the number of nodes for the above three rules under randomly generated network topologies. As the figure shows, $\|\mathbf{G}\| \geq 1$ is always true for all aggregate rules and topologies. The profile of $\rho(\mathbf{G})$ is not shown in Fig. 3 since $\rho(\mathbf{G})$ is always equal to 1.

The following simulation results show that the above three rules exhibit unobservable discrepancy in terms of convergence performance. For simplicity, we evaluate the performance of the ATU algorithm only under the nearest neighbor rule, which is the easiest way.

B. Verifying the Convergence of ATU

Our theoretical results have presented the sufficient conditions of convergence for the ATU algorithm, i.e., the good initial estimate $\mathbf{x}_G^0$, the appropriate step size parameter α and network topology represented by $\mathbf{G}$. First, the GN method for solving the NLLS problems is intrinsic to a good initial estimate, such that the Hessian in the Newton method can be

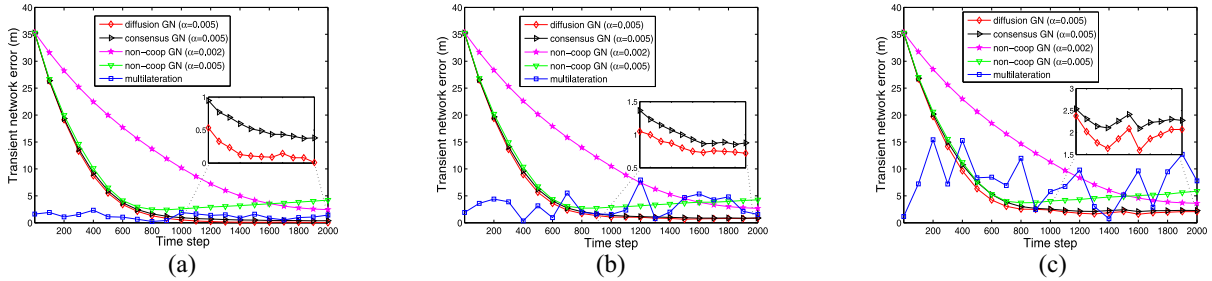


Fig. 4. Performance comparison on transient network error for 2000 time steps under noise standard deviation. (a) $v = 0.1$, (b) $v = 0.3$, and (c) $v = 0.5$.

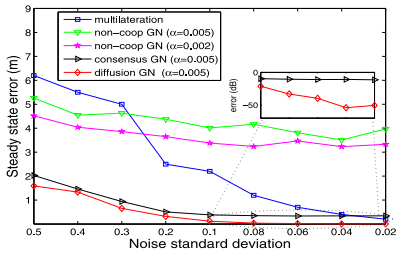


Fig. 5. Network-wide steady-state localization errors averaged over 20 nodes against noise variances at iteration 2000.

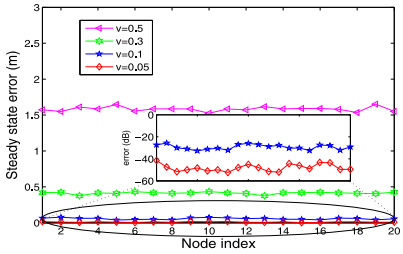


Fig. 6. Steady-state localization errors for all 20 nodes across the network.

approximate to simplify the computation of the second-order term. For the target localization application, many traditional methods can provide the coarse-grained initial guess with low communication and computation cost, such as triangulation [46]. In our simulation, all estimates start from 0, i.e., $\mathbf{x}_G^0 = 0$. Second, we consider the step size in our diffusion strategy as a constant for all nodes at every time, which needs to satisfy the condition (36). In general, a reasonable larger step size will exhibit the faster initial convergence. On the contrary, a smaller non-negative step size will show slow convergence, thus leading to more consumption of computation and communication resource.

We compare the convergence of diffusion algorithm with several range-based localization algorithms, including the well-known multilateration [46], noncooperative GN referred in (16), and consensus GN [12], where the last two methods are iterative and depend on the parameter of step size. To make the comparison fair, we use the same step size value $\alpha = 0.005$ under different noise variance, although our diffusion algorithm allows the step size selected in a larger scope, which will be investigated in subsequent section. Fig. 4 shows the comparison results on the transient network error over 2000 time steps when the noise standard deviation v is set as 0.1, 0.3, and 0.5 on all nodes, respectively. It is clear that larger noise variance leads to larger estimation errors for all four algorithms. However, our diffusion GN has better resistance to noisy communication than other algorithms that are incompetent for high noise levels. The reason is that each node in ATU absorbs global knowledge instead of local information. Accordingly, the GN descent step in ATU is affected solely by global noise level instead of local noise with large perturbation. The bad estimates resulted from biased local noise are shared by all nodes throughout the network.

In Fig. 4, we also can see that diffusion GN shows the faster convergence than noncooperative version. As we explained, the reason is that diffusion GN provides a decline on global error between two neighboring iterations over noncooperative solution. Such decline room benefits from the diffusion step that helps exchange temporal–spatial information across nodes in the network instead of local temporal information as in noncooperative schemes. Note that the noncooperative GN with a small step size $\alpha = 0.002$ causes the slower convergence, while it cannot guarantee to reach the steady state for a large step size $\alpha = 0.005$ (see the tail of its curve).

Fig. 5 shows the averaged steady-state estimation error over all 20 nodes at 2000 iterations under different noise variance levels. It is obvious that localization errors rely heavily upon the noise levels and diffusion version outperforms other versions on the steady-state performance. From Fig. 6, which shows the steady-state error over 20 nodes, the expected equilibrium effect is achieved by ATU since each node receives the global information instead of local limited information.

C. Robust to Step Size

As an important factor affecting the convergence of algorithm based on the sufficient condition (36), it is unrealistic to adjust frequently the value of step size in practice. Thus, the benefit of allowing a larger scope of step size is obvious. Under other parameters given as follows: $v = 0.1$ and $N = 20$, it can be seen from Fig. 7 that ATU algorithm exhibits fast convergence in the case of large step size (e.g., $\alpha = 0.05$) and maintains the stability of the system, while the noncooperative algorithm is very sensitive to the step size. Specially, the noncooperative algorithm cannot converge to the steady state and shows the growing error over time under the step size $\alpha = 0.005$. The observations verify that ATU can reach a higher fault tolerance level on selecting the step size, since the global diffusion step is helpful to mitigating the fluctuations on errors. The potential benefit is that the step size needs not to be carefully designed in practice. As we presented in

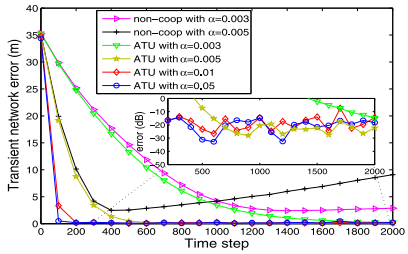


Fig. 7. Convergence comparison with noncooperative algorithms by varying the step size.

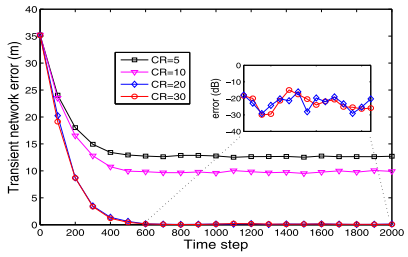


Fig. 8. Convergence comparison by varying CR.

the previous sections, the convergence of ATU can be guaranteed as long as the sufficient conditions about α and $\|\mathbf{G}\|$ are satisfied.

D. Impact of Network Connectivity

The performance of the ATU algorithm depends on the network connectivity to provide the information diffusion over the network, just as the sufficient condition (35) for $\|\mathbf{G}\|$ is required for reaching a stable equilibrium point. Both the number of nodes and CR will result in the change on the physical connectivity of network. In this simulation, we evaluate the impact of network topology on ATU performance by adjusting the CR, while keeping the number of nodes 20 and the step size $\alpha = 0.01$.

In Fig. 8, we randomly generate the learning curves for ATU algorithm under the CR of 5, 10, 20, and 30. Under CR $=$ 5, where the network is disconnected, it is shown that the ATU algorithm converges to a steady-state equilibrium with large steady state error. In this case, the diffusion strategy is not being fully exploited to reduce the steady-state error due to the low connectivity in the network disrupting the flow of information. From the discussion of Corollary 3 in Section V of the supplementary material and the investigation of diffusion step, a high connectivity network can effectively improve the convergence performance of ATU to achieve low steady-state error.

E. Robustness to Node and Link Failures

It is well known that the diffusion strategy is inherently robust to changes of the network topology, resulting from node or link failures. Figs. 9 and 10 show the transient network error in the presence of node and link failures when $N = 20$, CR $= 20$, and $\alpha = 0.01$. When the random failures without critical nodes or links are generated with probability 50% independently, where a node or link is critical if its removal will disconnect the network into two (or more)

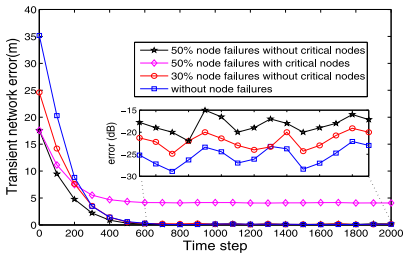


Fig. 9. Convergence comparison in the presence of node failures.

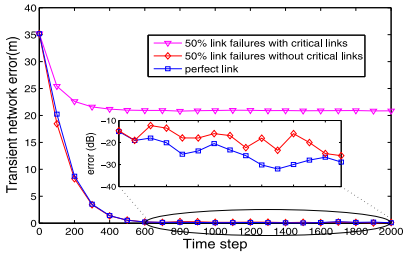
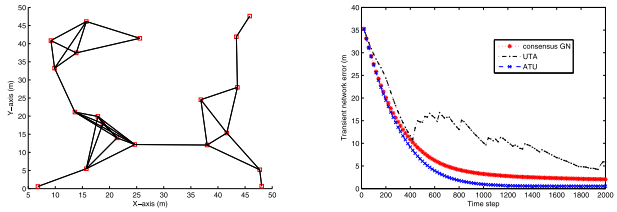


Fig. 10. Convergence comparison in the presence of link failures.



(a) (b)

Fig. 11. (a) Network topology used in this example. (b) Performance comparison between ATU and UTA.

separate components, the ATU algorithm is robust to these failures and exhibits excellent convergence and steady-state performance. However, in the presence of failures with critical nodes or links, the performance of ATU degrades significantly since local nodes cannot obtain the real-time global information through the diffusion step.

F. Comparison With UTA in Sparse Networks

From the perspective of implementing the mechanism, the ATU algorithm has the same processing and communication complexity as the UTA algorithm in the work [31] and the different order of two steps (i.e., ATU and UTA). However, the essential difference between them is that UTA needs to learn the knowledge from both 1-hop and 2-hops neighbors, while ATU uses information only from 1-hop neighbors to solve the global minimization problem (2) (see [31, eqs. (55) and (56)]). In other words, the performance of UTA depends largely on the neighboring nodes that are 1-hop and 2-hops away. Therefore, in the case of sparse networks with insufficient diffusion, UTA definitely will take a relatively long duration for convergence and also provide poor steady state performance. To illustrate this, we randomly generate a sparse but connected network with unbalanced node degrees by setting $N = 20$ and CR $= 15$, which is depicted in Fig. 11(a). Fig. 11(b) shows the corresponding convergence curves

by averaging over 100 experiments. It can be observed that ATU and consensus GN algorithms perform well due to their same ATU strategy, while the performance of UTA degrades significantly and becomes unacceptable. The results confirm obvious performance benefits of our ATU algorithm compared with UTA.

VI. Conclusion

In this article, we proposed the ATU diffusion GN paradigm for NLLSs problems in a distributed networked system and investigated elaborately its convergence performance. The presented theoretical results showed that the cooperation strategy can obtain a room for improvement in terms of convergence and guarantee algorithm's convergence when the derived sufficient conditions are satisfied. Particularly, cooperation has a robust effect on the network. The simulation results also showed that the cooperation strategy can lead to the advantages in many aspects, including convergence rate, robustness to node or link failures, and step size by applying the ATU algorithm to target localization in a wireless network.

Our future work will focus on three directions. First, our scheme can be combined with various popular topology to match different application scenarios, such as cyclic, clustered, and gossip-based network. Second, we will evaluate the feasibility by using our Aggregation-then-Update strategy to improve the performance for a batch of Newton-type methods in the adaptive networked system. Third, adaptive versions of our diffusion GN can be proposed by designing the adaptive step size to resist the fail of local convergence and the adaptive weights to avoid data incest and double counting from neighborhood.

References

[1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[2] L. Consolini and G. Lini, "A Gauss-Newton method for the synthesis of periodic outputs with central pattern generators," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 7, pp. 1394–1400, Jul. 2014.

[3] C. C. Wang *et al.*, "Distributed Newton methods for deep neural networks," *Neural Comput.*, vol. 30, no. 6, pp. 1673–1724, 2018.

[4] J. Wang and K. D. Pham, "An approximate distributed gradient estimation method for networked system optimization under limited communications," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 12, pp. 5142–5151, Dec. 2020.

[5] X. Li and A. Scaglione, "Convergence and applications of a gossip-based Gauss-Newton algorithm," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5231–5246, Nov. 2013.

[6] S. Gratton, A. S. Lawless, and N. K. Nichols, "Approximate Gauss-Newton methods for nonlinear least squares problems," *SIAM J. Optim.*, vol. 18, no. 1, pp. 106–132, 2007.

[7] J.-F. Bao, C. Li, W.-P. Shen, J.-C. Yao, and S. M. Guu, "Approximate Gauss-Newton methods for solving underdetermined nonlinear least squares problems," *Appl. Numer. Math.*, vol. 111, pp. 92–110, Jan. 2017.

[8] X. Xiong and F. D. L. Torre, "Supervised descent method for solving nonlinear least squares problems in computer vision," 2014. [Online]. Available: arXiv:1405.0601.

[9] F. D. L. Torre and M. H. Nguyen, "Parameterized kernel principal component analysis: Theory and applications to supervised and unsupervised image alignment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[10] M. Schweiger, S. R. Arridge, and I. Nissila, "Gauss-Newton method for image reconstruction in diffuse optical tomography," *Phys. Med. Biol.*, vol. 50, no. 10, pp. 2365–2386, 2005.

[11] G. C. Calafiore, L. Carlone, and M. Wei, "A distributed technique for localization of agent formations from relative range measurements," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 42, no. 5, pp. 1065–1076, Sep. 2012.

[12] B. Bejar, P. Belanovic, and S. Zazo, "Distributed consensus-based tracking in wireless sensor networks: A practical approach," in *Proc. 19th Eur. Signal Process. Conf.*, 2011, pp. 2019–2023.

[13] J. R. Jensen, M. G. Christensen, and S. H. Jensen, "Nonlinear least squares methods for joint DOA and pitch estimation," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 21, no. 5, pp. 923–933, May 2013.

[14] Y. Zhuang, W.-S. Chin, Y.-C. Juan, and C.-J. Lin, "Distributed Newton methods for regularized logistic regression," in *Proc. Pac. Asia Conf. Knowl. Disc. Data Min.*, 2015, pp. 690–703.

[15] X. Yang and X. Zheng, "Gradient descent algorithm-based adaptive NN control design for an induction motor," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 2, pp. 1027–1034, Feb. 2021.

[16] C. T. Kelley, *Iterative Methods for Optimization*. Philadelphia, PA, USA: SIAM, 1999.

[17] S. S. Haykin, *Neural Networks and Learning Machines*. New York, NY, USA: Prentice Hall, 2009.

[18] Z. Yao, J. Yao, and W. Sun, "Adaptive rise control of hydraulic systems with multilayer neural-networks," *IEEE Trans. Ind. Electron.*, vol. 66, no. 11, pp. 8638–8647, Nov. 2019.

[19] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.

[20] A. Nedić and J. Liu, "Distributed optimization for control," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 1, pp. 77–103, May 2018.

[21] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion strategy for optimization by networked agents," in *Proc. 25th Eur. Signal Process. Conf. (EUSIPCO)*, Aug. 2017, pp. 141–145.

[22] M. Wu, N. N. Xiong, and L. Tan, "An intelligent adaptive algorithm for environment parameter estimation in smart cities," *IEEE Access*, vol. 6, pp. 23325–23337, 2018.

[23] A. Khalili, A. Rastegarnia, and S. Sanei, "Performance analysis of incremental LMS over flat fading channels," *IEEE Trans. Control Netw. Syst.*, vol. 4, no. 3, pp. 489–498, Sep. 2017.

[24] I. Matei, C. Somarakis, and J. S. Baras, "A generalized gossip algorithm on convex metric spaces," *IEEE Trans. Autom. Control*, vol. 60, no. 5, pp. 1175–1187, May 2015.

[25] J. Chen, C. Richard, and A. H. Sayed, "Diffusion LMS over multitask networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2733–2748, Jun. 2015.

[26] R. Abdolee and B. Champagne, "Diffusion LMS strategies in sensor networks with noisy input data," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 3–14, Feb. 2016.

[27] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1865–1877, May 2008.

[28] G. Wang, N. Li, and Y. Zhang, "Diffusion distributed Kalman filter over sensor networks without exchanging raw measurements," *Signal Process.*, vol. 132, pp. 1–7, Mar. 2017.

[29] V. Vahidpour, A. Rastegarnia, A. Khalili, and S. Sanei, "Partial diffusion Kalman filtering for distributed state estimation in multiagent networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3839–3846, Dec. 2019.

[30] G. C. Calafiore, L. Carlone, and M. Wei, "A distributed Gauss-Newton approach for range-based localization of multi agent formations," in *Proc. IEEE Int. Symp. Comput. Aided Control Syst. Design*, 2010, pp. 1152–1157.

[31] N. Xiong, M. Wu, V. C. M. Leung, and L. T. Yang, "The effective cooperative diffusion strategies with adaptation ability by learning across adaptive network-wide systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Aug. 13, 2019, doi: 10.1109/TSMC.2019.2931060.

[32] A. Bjorck, *Numerical Methods for Least Squares Problems*. Philadelphia, PA, USA: SIAM, 1996.

[33] X. Zhu, J. Li, and M. Zhou, "Optimal deployment of energy-harvesting directional sensor networks for target coverage," *IEEE Syst. J.*, vol. 13, no. 1, pp. 377–388, Mar. 2019.

[34] X.-M. Zhang *et al.*, "Networked control systems: A survey of trends and techniques," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 1, pp. 1–17, Jan. 2020.

[35] T. Bai, S. Li, and Y. Zheng, "Distributed model predictive control for networked plant-wide systems with neighborhood cooperation," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 108–117, Jan. 2019.

[36] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD, USA: Johns Hopkins Univ., 2009.
[37] C. R. Johnson and R. A. Horn, *Topics in Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1991.
[38] E. D. E. Kenneth and J. Claes, *Applied Mathematics, Body and Soul: Derivatives and Geometry in IR3*. Heidelberg, Germany: Springer, 2004.
[39] O. Galor, *Discrete Dynamical Systems*. Berlin, Germany: Springer, 2007.
[40] W.-H. Steeb and Y. Hardy, *Matrix Calculus and Kronecker Product: A Practical Approach to Linear and Multilinear Algebra Second Edition*. Singapore: World Sci. Publ. Company, 2011.
[41] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Library in Signal Processing*, vol. 3. Oxford, U.K.: Academic, 2014, pp. 323–453.
[42] B. Xia, W. Yuan, N. Xie, and C. Li, "A novel statistical manifold algorithm for position estimation," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 6, pp. 1513–1518, Nov. 2019.
[43] F. Santoso, "A new framework for rapid wireless tracking verifications based on optimized trajectories in received signal strength measurements," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 11, pp. 1424–1436, Nov. 2015.
[44] Y. Kang, Z. Li, Y. Zhao, J. Qin, and W. Song, "A novel location strategy for minimizing monitors in vehicle emission remote sensing system," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 4, pp. 500–510, Apr. 2018.
[45] K. Derr and M. Manic, "Wireless sensor networks—Node localization for various industry problems," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 752–762, Jun. 2015.
[46] Y. Zhou, J. Li, and L. Lamont, "Multilateration localization in the presence of anchor location uncertainties," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 309–314.

Mou Wu (Member, IEEE) received the Ph.D. degree in radio physics from Central China Normal University, Wuhan, China, in June 2015.

He is currently an Associate Professor with the School of Computer Science and Technology, Hubei University of Science and Technology, Xianning, China. From 2018 to 2020, he is a Postdoctoral Fellow with the College of Intelligence and Computing, Tianjin University, Tianjin, China. His research interests include distributed algorithms for performance optimization over networked systems.

Naixue Xiong (Senior Member, IEEE) received the first Ph.D. degree in sensor system engineering from Wuhan University, Wuhan, China, in 2007, and the second Ph.D. degree in dependable sensor networks from the Japan Advanced Institute of Science and Technology, Nomi, Japan, in 2008.

He was with the Northeastern State University, Tahlequah, OK, USA; Georgia State University, Atlanta, GA, USA; the Wentworth Institute of Technology, Boston, MA, USA; and Colorado Technical University, Colorado Springs, CO, USA (Full Professor for about five years) for about ten years. He is currently a Professor with the Department of Mathematics and Computer Science, Northeastern State University. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.

Liansheng Tan received the Ph.D. degree from Loughborough University, Loughborough, U.K., in 1999.

He is currently a Professor with the Department of Computer Science, Central China Normal University, Wuhan, China. He was a Research Fellow with the Research School of Information Sciences and Engineering, The Australian National University, Canberra, ACT, Australia, from 2006 till 2009, and a Postdoctoral Research Fellow with the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON, Canada, in 2001. He has also held a number of visiting research positions with Loughborough University; the University of Tsukuba, Tsukuba, Japan; the City University of Hong Kong, Hong Kong; and the University of Melbourne, Melbourne, VIC, Australia. His research interests include resource allocation and management of wireless and wireline networks and routing and transmission control protocols.

Jianguo Wei (Member, IEEE) received the M.S. degree from Tianjin University, Tianjin, China, in 2004, and the Ph.D. degree from the Japan Advanced Institute of Science and Technology, Nomi, Japan, in 2007.

He is currently a Professor with the College of Intelligence and Computing, Tianjin University. He is also the Dean of the School of Computer Science, Qinghai Nationalities University, Xining, China. He has devoted himself to the research of speech signal processing, phonetics, and speech production.

Jie Wu (Fellow, IEEE) received the Ph.D. degree in computer engineering from Florida Atlantic University, Boca Raton, FL, USA, in 1989.

He is currently the Director of the Center for Networked Computing and also a Laura H. Carnell Professor with Temple University, Philadelphia, PA, USA, where he also serves as the Director of International Affairs with the College of Science and Technology. He served as a Chair of the Department of Computer and Information Sciences, Temple University from the summer of 2009 to the summer of 2016 and as an Associate Vice Provost for International Affairs from the fall of 2015 to the summer of 2017. He was the Program Director with the National Science Foundation and was a Distinguished Professor with Florida Atlantic University.

Dr. Wu was an IEEE Computer Society Distinguished Visitor and an ACM Distinguished Speaker. He is a China Computer Federation Distinguished Speaker.