A General Framework to Distribute Iterative Algorithms With Localized Information Over Networks

Thomas Ohlson Timoudas, Silun Zhang, Sindri Magnússon, *Member, IEEE*, and Carlo Fischione, *Member, IEEE*

***Abstract*—Emerging applications in the Internet of Things (IoT) and edge computing/learning have sparked massive renewed interest in developing distributed versions of existing (centralized) iterative algorithms often used for optimization or machine learning purposes. While existing work in the literature exhibits similarities, for the tasks of both algorithm design and theoretical analysis, there is still no unified method or framework for accomplishing these tasks. This article develops such a general framework for distributing the execution of (centralized) iterative algorithms over networks in which the required information or data is partitioned between the nodes in the network. This article furthermore shows that the distributed iterative algorithm, which results from the proposed framework, retains the convergence properties (rate) of the original (centralized) iterative algorithm. In addition, this article applies the proposed general framework to several interesting example applications, obtaining results comparable to the state of the art for each such example, while greatly simplifying and generalizing their convergence analysis. These example applications reveal new results for distributed proximal versions of gradient descent, the heavy ball method, and Newton's method. For example, these results show that the dependence on the condition number for the convergence rate of this distributed heavy ball method is at least as good as that of centralized gradient descent.**

***Index Terms*—Agents and autonomous systems, communication networks, distributed algorithms, optimization algorithms.**

I. Introduction

THE need for machine learning and optimization in Internet of Things (IoT) applications and edge computing systems has pushed a renewed interest into iterative algorithms that solve, for example, machine learning or optimization problems by distributed computations. Many recent developments of distributed iterative algorithms have adapted and further developed ideas from, for example, network consensus [1], dynamic average consensus and consensus tracking [2], [3], distributed optimization [4], [5], [6], and distributed learning [7], [8]. Concrete examples can be found throughout the vast literature on multiagent systems and consensus, having broad applications such as networked synchronization [9], [10], [11], formation control [12], [13], [14], multidimensional scaling [15], sensor fusion [16], [17], and signal recovery [18].

The idea of investigating the convergence properties of distributed iterative algorithms drawing on the massive wealth of existing work in the nondistributed (also referred to as centralized) setting, and combining it with a network consensus protocol, appears to be potentially quite fruitful. Yet, the development of such distributed algorithms from existing centralized ones still requires significant work, both for the algorithm design, and to establish their theoretical convergence properties, even though many examples share many characteristics from both aspects [19], [20]. There is an urgent need to standardize this process and the methods to transform centralized iterative algorithms into distributed ones, and to simplify the development of new distributed algorithms for future networked applications, while taking advantage of the significant existing work on centralized algorithms.

Such a unified framework would have additional benefits. For instance, many emerging IoT applications have very different requirements compared to traditional distributed settings, e.g., relying on massive wireless networks and/or (provisional) mesh networks without a central coordinator, with intermittent and/or limited communication in terms of bandwidth and data rates [21], [22]. Additionally, the data needed for many applications is generated by several different nodes with different

Manuscript received 4 October 2022; revised 25 April 2023; accepted 7 May 2023. Date of publication 25 May 2023; date of current version 5 December 2023. The work of Carlo Fischione was supported in part by the Digital Futures Research Center in KTH, in part by the SSF Project SAICOM, and in part by the VR Project MALEN. The work of Sindri Magnússon was supported by Digital Futures and the Swedish Research Council (Vetenskapsrådet) under Grant 2020-03607. Recommended by Associate Editor Y. Wang. *(Corresponding author: Thomas Ohlson Timoudas.)*

Thomas Ohlson Timoudas is with RISE Research Institutes of Sweden, Division Digital Systems, Computer Science, 164 40 Kista, Sweden (e-mail: thomas.ohlson.timoudas@ri.se).

Silun Zhang is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: silunz@mit.edu).

Sindri Magnússon is with the Department of Computer and System Science, Stockholm University, 114 19 Stockholm, Sweden (e-mail: sindrimagn@gmail.com).

Carlo Fischione is with the Digital Futures, KTH Royal Institute of Technology, 100 44 Stockholm, Sweden (e-mail: carlofi@kth.se).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2023.3279901.

Digital Object Identifier 10.1109/TAC.2023.3279901

ownership. As a result, these applications tend to emphasize higher degrees of decentralization and autonomy, data privacy, energy conservation, and interference mitigation. These network and communication considerations add another layer to the distributed algorithm design. A unified framework for distributing (centralized) iterative algorithms could also facilitate the analysis and integration of such an added communications layer, independently of the specifics of the "base" algorithm that is being distributed.

We shall now define the class of algorithms that we consider in this article. In many applications, a (the) solution $x^* \in \mathbb{R}^d$ to a given problem can be found as a fixed point of an iterative algorithm

$$x(t+1) = \Phi(x(t)) = F(x(t), Dx(t)) \tag{1}$$

given by a mapping Φ, where $F : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$ and $D : \mathbb{R}^d \to \mathbb{R}^m$ are some operators. Here, $x(t)$ is an iterate that is updated according to the algorithm $F(\cdot)$, and $Dx(t)$ is some additional information, possibly depending on $x(t)$, required to execute the algorithm, e.g., a gradient or a Hessian matrix. We shall refer to the operator D as the "data" operator. However, D does not (necessarily) represent a concrete data set, but rather some additional information derived from such a data set. Examples of algorithms that can be expressed in this form include vanilla/proximal gradient methods, Newton's method, and dynamic programming [23], [24].

Now that the class of algorithms under consideration has been made clear, we shall clarify the network setting—how the data are distributed throughout the network. In many of the emerging IoT applications, the data will not be located at a single node, but rather distributed between multiple nodes in the network. For instance, in networks where all the nodes exploit data in the same manner, $Dx(t)$ is often the average of local data operators [25]

$$D(x) = \frac{1}{n}\sum_{i=1}^{n} D_i(x) \tag{2}$$

where $D_i(x)$ is the local data (operator) of node i, e.g., $D_i(x)$ is the local gradient or Hessian matrix for the node i, evaluated at the point x. In highly dense and/or mesh networks, it may not be feasible to transmit all the local data $D_i x(t)$ to a central coordinator, e.g., due to power/interference constraints, or the need to route data through many nodes. In such cases, especially in the absence of a central coordinator, it may be therefore necessary to compute the solution using the algorithm in (1), in a distributed fashion, without access to the full data (operator) D, using instead local estimates of D based on communication between network neighbors (i.e., consensus/gossip).

The goal of this article is to develop a general method for distributing the execution of such iterative (centralized) algorithms in networks.

A. Related Work

In recent years, the study of distributed learning and optimization has seen tremendous interest and advances. In fact, many problems in engineering, learning, and coordination of networked agents can be formulated (at least in part) as an optimization problem. Early work on decentralized algorithms to solve such optimization problems include primal-dual approaches for resource allocation in networks [26], and incremental subgradient methods, by which the nodes sequentially broadcast their local (sub)gradients to the others, followed by a common gradient update, in a round-robin fashion [27], in random sequence [28], or more complicated deterministic sequence [29]. These ideas were further developed to better scale in networks with many nodes in [6], allowing nodes in (decentralized) mesh networks to transmit their local gradients and perform updates in parallel (and only to their direct network neighbors), using ideas from consensus (and in particular dynamic average consensus and consensus tracking [2], [3]) that had proved successful in applications to multiagent control [10], [11], and sensor fusion [16].

One major drawback of the early distributed subgradient method introduced in [6], is that it requires decaying step sizes. While suitable (and sufficient) for nonsmooth objective functions, this requirement results in sublinear convergence rates even for strongly convex objective functions – in contrast to the (nondistributed) standard gradient descent. The authors in [30] and [31] analyzed the effects of using a (sufficiently small) constant step size, and found that the (mean) squared error in fact decreases linearly, but only up to a certain, nonzero error bound (which increases with the step size). Subsequent work showed that it was possible to modify this algorithm to attain linear convergence rates in the strongly convex case.

Interestingly, several different approaches to achieve a linear convergence rate have been developed in recent years, and can be mainly divided into the following two approaches:

1) adaptive correction terms using previous iterates, e.g., EXTRA [32]; and
2) gradient tracking methods, e.g., Aug-DGM [33], DIG [34], and [35].

These ideas have also been adapted to the context of yet other optimization algorithms, such as Nesterov gradient descent [36], the heavy ball method [37], (inexact) Newton–Raphson [38], projected/proximal gradient descent [20], [39], [40], [41], and mirror descent [42]. Alongside these developments, decentralized versions of many other algorithms have been (independently) developed based on similar ideas, such as dual averaging [43], ADMM [44], [45], and forward-backward Bregman splitting schemes [46].

Many of these algorithms in the literature are very similar, with differences in order of operations, what exactly is communicated in the consensus steps, and/or differences due to the particular algorithm that is being decentralized. In particular, they all rely on ideas developed in earlier work on dynamic average consensus and consensus tracking [2], [3]. As a result, recent work has attempted to reconcile these in a unified algorithmic framework. In particular, [19], [20], [41], [47] have proposed general frameworks that unify algorithm design and convergence proofs for several different variations of distributed gradient descent, and their proximal versions, in the literature. A general separation principle for designing decentralized optimization algorithms, by combining a base optimization algorithm with consensus tracking, is proposed in [48]. Additionally, a unified

convergence analysis approach is proposed, and demonstrated on decentralized versions of gradient descent and ADMM, by regarding the gradient operators as a feedback regulator in a closed-loop dynamical system. These works provided a view to separate the ingredients of optimization and consensus average in distributed optimization algorithms, which depicts a divide and conquer strategy in distributed optimization, i.e., separating the development of new consensus dynamics, e.g., [2], [3], with the optimization iterations. In this article, we continue such a path to separate network collaborations within a class of general contraction iterations.

B. Summary of Our Contributions

We summarize our main contributions as follows.

- Given a general class of iterative algorithms designed for centralized settings, we formalize and develop a framework for distributing its execution in (mesh) networks, for which the information necessary for such an execution is spread throughout the network.
- We prove that, if the original algorithm has a linear convergence rate, then the distributed algorithm also has a linear convergence rate (with a different constant factor), for a general class of algorithms.
- As a special case, we show that our framework provides a unified method and analysis distributing multiple different first-order gradient methods (e.g., standard gradient descent, (proximal) heavy ball, proximal gradient descent), as well as second-order Newton methods.
- These results show, for instance, that the dependence on the condition number (the ratio between the smoothness and strong convexity parameters) of the convergence rate of our distributed (proximal) heavy ball method is no worse than for centralized gradient descent—an improvement on the existing state of the art.
- In this work, the analysis of proximal distributed versions follow immediately from the corresponding analysis of the nonproximal case. Accordingly, we can show that the proximal distributed version of (any) such algorithm enjoys the same convergence rate as the distributed version of the non- algorithm (analogous to the centralized setting).

C. Structure of this Article

Section II introduces the problem formulation, together with a list of assumptions we will use to analyze the problem and prove our theoretical results. The proposed distributed solution (algorithm) is given in Section III. Section IV contains our theoretical main results about the convergence (rate) of the proposed algorithm. The main results are applied to several examples from optimization in Section V. The results in Section V demonstrate the wide applicability of our framework, and also establish some new results for specific example algorithms. The proofs of the main theoretical results are given in Section VI.

D. Notation

In the rest of this article, without extra indication, all vectors are regarded as real-valued row vectors. The exception to this rule is that $\mathbf{1}_k$ always denotes a column vector in $\mathbb{R}^{k\times 1}$, whose components are all 1. Given n vectors $v_1,\dots,v_n\in\mathbb{R}^d$, we denote by $\overline{v}\in\mathbb{R}^d$ the average of the vectors:

$$\overline{v}=\frac{1}{n}\sum_{i=1}^{n}v_i.$$

Given a matrix $A\in\mathbb{R}^{m\times n}$, we denote by $A_{\cdot,i}$ and $A_{j,\cdot}$ the ith column and jth row of matrix A, respectively. The transpose of A is denoted by A^{T}.

For vectors, $\|\cdot\|_2$ denotes the Euclidean norm, and for matrices it denotes the induced operator norm, which is defined by

$$\|A\|_2=\max_{v\neq 0}\frac{\|Av\|_2}{\|v\|_2}.$$

The norm $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm of matrices, i.e.,

$$\|A\|_{\mathrm{F}}=\sqrt{\sum_{i,j}a_{ij}^2}$$

for matrices $A=(a_{ij})$.

The map I refers to "the" identity map, and the domain of the map is given by the context.

II. Problem Formulation

As mentioned in the introduction, many interesting applications involve finding fixed point solutions to an iterative algorithm of the form (1), which could be equivalently expressed as the (iterative) algorithm

$$\begin{aligned}x(t+1)&=F\left(x(t),y(t)\right)\\ y(t+1)&=Dx(t+1).\end{aligned}\tag{3}$$

Recall that $F:\mathbb{R}^d\times\mathbb{R}^m\to\mathbb{R}^d$ is some operator, given by the algorithm of the problem application, and that the *data operator* $D:\mathbb{R}^d\to\mathbb{R}^m$ models some additional input data, which may depend on the current iterate $x(t)$, required to execute the algorithm. Note that many optimization algorithms can fit the form given in (3). In Section V, the correspondence between concrete examples and the abstract algorithm formulation in (3) is demonstrated, see Section V-A for proximal gradient descent method, Section V-B for heavy ball method, and Section V-C for Newton's method.

The premise of this article is as follows: suppose that n distinct agents (called *nodes*) have the common goal of finding the solution (fixed point) x^* to an iterative algorithm of the form (1)

$$x(t+1)=\Phi(x(t))=F(x(t),Dx(t)).$$

We shall assume that this iterative algorithm is linearly convergent, i.e., that $\Phi:\mathbb{R}^d\times\mathbb{R}^d$ has a unique fixed point x^* satisfying $\|\Phi(x)-x^*\|\leq r\|x-x^*\|$, for some $0\leq r<1$ and every $x\in\mathbb{R}^d$. The exact technical assumptions used in this article are given in Section II-A.

We will furthermore assume that the data D is distributed between these n nodes in the network, such that each node $i=1,\dots,n$ has a local data operator $D_i:\mathbb{R}^d\to\mathbb{R}^m$, and that these local data operators average to D

$$Dx=\frac{1}{n}\sum_{i=1}^{n}D_ix\tag{4}$$

for every $x \in \mathbb{R}^d$. For example, $D_i x$ might be the local gradient or Hessian matrix of node i at the point x. The assumption in (4) is in general not very restrictive, see Remark 4 and the discussion following it, in Section II-A. In particular, it is assumed that no single node knows the (global) data operator $D(\cdot)$ – however, it is assumed that all of the nodes know the operator F (i.e., how to combine the local data in the algorithm).

Communication between the nodes is restricted to the edges of a fixed (constant) undirected (network) graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (which may be arbitrary), in which the vertices $\mathcal{V} = \{1, \dots, n\}$ represent the agents, and the edges in $\mathcal{E}$ represent (bidirectional) communication links. We assume that the graph is connected, but it can otherwise have arbitrary topology. Two nodes $i, j \in \mathcal{V}$ are called neighbors if and only if there is an edge $(i, j) \in \mathcal{E}$, and we write $j \in \mathcal{N}(i)$ if there is an edge from node i to j. In this network model, two agents $i, j \in \mathcal{V}$ are able to communicate if and only if they are neighbors. Since each node has access to its own data, we assume that $(i, i) \in \mathcal{E}$ for every $i \in \mathcal{V}$. Given a graph $\mathcal{G}$ with n nodes, a *mixing matrix* for the graph $\mathcal{G}$ is a doubly stochastic[1] matrix $W = (w_{ij}) \in \mathbb{R}^{n \times n}$, whose elements satisfy $w_{ij} > 0$ if and only if $(i, j) \in \mathcal{E}$. Suppose that σ is the second largest eigenvalue of W (1 is the largest one). For such matrices, one can show that

$$\|Wx - \frac{1}{n}\sum_{i=1}^{n} x_i \mathbf{1}\|_2 \leq \sigma \|x - \frac{1}{n}\sum_{i=1}^{n} x_i \mathbf{1}\|_2$$

for any (column) vector $x = (x_1, \dots, x_n) \in \mathbb{R}^{n \times 1}$. We are now ready to state the main problem (inspired by a conjecture in [35]) to which this article is devoted.

Problem 1: Develop an iterative distributed (decentralized) algorithm to find the solution (fixed point) x^* of (3) that maintains a linear convergence rate,[2] such that communication is restricted to the edges (links) in $\mathcal{E}$.

It would be interesting to pose and solve the same problem for other convergence rates, such as sub-linear. In Section V, we further explore this problem in the context of several common optimization algorithms, and how our results apply to them, such as:

1) standard (and proximal) gradient descent;
2) the heavy ball method;
3) Newton's method.

To develop distributed solutions (algorithms) to this problem, and establish their theoretical properties, we must first pin down the structures and assumptions we impose on the original algorithm (3).

A. Assumptions

In this work, we consider algorithms that converge linearly. More formally, we make the assumption.

Assumption 1: There is a constant $r \in [0, 1)$, and a unique fixed point $x^* \in \mathbb{R}^d$ of Φ, i.e., $x^* = \Phi(x^*)$, such that

$$\|\Phi(x) - x^*\|_2 \leq r\|x - x^*\|_2 \tag{5}$$

for every $x \in \mathbb{R}^d$. Moreover, let $0 \leq \delta$ be a constant satisfying

$$\|\Phi(x) - x\|_2 \leq \delta\|x - x^*\|_2 \tag{6}$$

for every $x \in \mathbb{R}^d$.

Remark 1: Note that any contraction mapping $\Phi(x)$ satisfies (5), which is therefore weaker than assuming that $\Phi(x)$ is a contraction. Moreover, the assumption (5) also implies that the underlying algorithm $\Phi(\cdot)$ has a linear convergence rate.

The more general algorithms with other convergence rates are left for the future study. Here, we conjecture that for any algorithms with a sublinear convergence rate, a similar decentralization method exists as proposed in this article.

Remark 2: Furthermore, for any mapping Φ that satisfies (5), the assumption in (6) naturally holds [3] with $\delta = 1 + r \leq 2$ (which is not sharp, in general). The reason we still indicate (6) explicitly, is because the convergence rate we obtain for our proposed algorithm depends on δ—a smaller δ gives a better convergence rate.

While we expect it to be possible to extend our results to algorithms that are convergent in the Lyapunov-sense, i.e., with respect to a Lyapunov function, that would require a substantially more complicated proof. Therefore, we leave it as a possible future extension of our work.

We also need to impose some smoothness conditions on the algorithm and data operator. Specifically, we assume that both of the functions F and D are Lipschitz continuous. Lipschitz continuity for F can be stated as follows:

Assumption 2: There are positive real numbers L_x and L_y, such that the function $F : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$ satisfies both

$$\|F(x, y) - F(z, y)\|_2 \leq L_x\|x - z\|_2$$

and

$$\|F(x, y) - F(x, u)\|_2 \leq L_y\|y - u\|_2$$

for any points $x, z \in \mathbb{R}^d$, and $y, u \in \mathbb{R}^m$.

Additionally, we need to assume that the local data operators D_i are Lipschitz continuous, to ensure that the consensus step converges.

Assumption 3: There is a positive real number L_D, such that, for every $1 \leq i \leq n$, the map $D_i : \mathbb{R}^d \to \mathbb{R}^m$ satisfies

$$\|D_i x - D_i z\|_2 \leq L_D\|x - z\|_2$$

for any points $x, z \in \mathbb{R}^d$. Moreover, (4) holds.

Remark 3: The Lipschitz condition in Assumption 3 could be stated in a weighted L^2-norm to allow more flexibility when applying our framework to algorithms with complex data dependencies. We have opted for this simpler case, to keep the notation in the proofs to a minimum.

Remark 4: If D is instead given by a weighted sum $D = \sum a_i D_i$, one may use the transformation $D_i \mapsto \frac{n}{a_i} D_i$ to bring D onto the form $D = \frac{1}{n}\sum D_i$ in (4), without changing the information content of D.

In many applications, it is natural to assume that (4) holds, e.g., in joint optimization problems in which the global objective function is the sum (or average) of local objective functions. This assumption is in general not very restrictive, since the local

[1] Every component of the matrix is nonnegative, and moreover each row's components sum to one, and each column's components sum to one.

[2] The iterative algorithm in (3) has a linear convergence since $r \in [0, 1)$. The convergence rate of the distributed algorithm may have different constants.

[3] By the triangle inequality, $\|\Phi(x) - x\|_2 \leq \|\Phi(x) - x^*\|_2 + \|x^* - x\|_2 \leq (1 + r)\|x^* - x\|_2$.

operators can in many cases be embedded into a direct product space, so that $D = \frac{1}{n}(D_1, \dots, D_s)$, and then combined inside the function F.

Lastly, we impose the following requirement on the communication graph.

Assumption 4: The network graph $\mathcal{G}$ is connected, and the second largest eigenvalue σ of the mixing matrix $W = (w_{ij})$ satisfies $0 \leq \sigma < 1$.

This is a standard assumption, and is necessary to ensure that the local data can spread to every node (and that the nodes can reach consensus).

III. Proposed Distributed Solution

The main idea to decentralize an (iterative) algorithm in a network is balancing the original iteration with the data spreading and synchronization amongst the nodes. For an algorithm given by the mapping $\Phi(\cdot)$, such a balance of two simultaneous procedures, iteration, and synchronization, is often facilitated by linear interpolation of Φ

$$\Phi_\theta = (1-\theta)\,\mathrm{I} + \theta\Phi \tag{7}$$

where I is the identity mapping, and the interpolation weight θ varies between 0 and 1, i.e., $0 < \theta \leq 1$. Note that a point x is a fixed point of Φ if and only if it is also a fixed point of Φ_θ, and that $\Phi_1 = \Phi$. Essentially, θ is a "step size"—decreasing it means taking a smaller "step" in the update direction determined by $\Phi - \mathrm{I}$, which can be used to increase the relative strength of a synchronization step (e.g., consensus) between nodes. This is also a common construction in the study of nonexpanding operators - for example, the parameter θ is the (normalized) *step size* used to ensure convergence of gradient descent methods.

The linear interpolation for the mapping $F(\cdot,\cdot)$ similarly reads

$$F_\theta(x,y) = x - \theta(x - F(x,y)) = (1-\theta)x + \theta F(x,y). \tag{8}$$

Notice that we interpolate only with respect to the first variable, but not the second one (the input data variable). Using the above interpolations, the corresponding iterative algorithm is given by

$$x(t+1) = \Phi_\theta(x(t)) = F_\theta(x(t), Dx(t))$$

which has exactly the same solutions (fixed points) as the original one in (3) using Φ.

Our proposed solution to 1, given by Algorithm 1, only relies on (parallel) local execution of the algorithms, using local estimates of the global data operators D, followed by a consensus (communication and combination) step. The consensus step, which is an extension of consensus tracking and gradient tracking in the literature, essentially "diffuses" the local data operators D_i throughout the network, to compute (delay-compensated) local estimates that converge to the global operator D. This is the reason for the name of the algorithm—*interleaved network consensus and local iteration compensation*, or *INCLIC*.

Algorithm 1: (INCLIC) Interleaved network consensus and local iteration compensation.

Input: The mixing matrix $W = (w_{ij})$, and the parameter $\theta \in (0,1]$ for the interpolation.

1: **Initialization:** Each agent $i \in [n]$ initializes its local state variables $x_i(0)$ (any initial value is allowed) and $y_i(0) = D_i x_i(0)$.[4]

2: **At each iteration** $t = 0, 1, \dots$**, each agent** i **performs the following process (in parallel):**

Step 1 (Locally) compute the intermediate variable

$$\xi_i(t+1) = F_\theta(x_i(t), y_i(t)).$$

Step 2 Send the local variable $\xi_i(t+1)$ to every network neighbor $j \in \mathcal{N}(i)$. Upon receiving the local variables of every network neighbor $j \in \mathcal{N}(i)$, (locally) compute the state variable

$$x_i(t+1) = \sum_{j \in \mathcal{N}_i} w_{ij}\xi_j(t+1).$$

Step 3 (Locally) compute the intermediate variable

$$\phi_i(t+1) = y_i(t) + D_i x_i(t+1) - D_i x_i(t).$$

Step 4 Send the local variable $\phi_i(t+1)$ to every network neighbor $j \in \mathcal{N}(i)$. Upon receiving the local variables of every network neighbor $j \in \mathcal{N}(i)$, (locally) compute the data estimate variable

$$y_i(t+1) = \sum_{j \in \mathcal{N}_i} w_{ij}\phi_j(t+1).$$

Step 5 Store the variables $x_i(t+1)$ and $y_i(t+1)$ for the next iteration.

Step 6 (Repeat): Increase the iteration counter by one, i.e., set $t := t+1$, and go to **Step 1**.[5]

3: END

To this end, introduce the local variables $x_i(t)$, representing the local iterate at time t of the node $i \in \mathcal{V}$, and the local variables $y_i(t)$, representing the local estimate of the data operator D at time t of the node i. Each node $i \in \mathcal{V}$ first initializes the local variables, in such a way that $x_i(0) \in \mathbb{R}^d$ may be chosen arbitrarily, and then setting each $y_i(0) = D_i x_i(0)$.

At each new iteration $t+1$, each node will first update their local variables, and then send these updated values to its neighbors in the communication graph $\mathcal{G}$. First, the nodes update the x-variable, as in Steps 1 and 2. The local iterate x_i of each node $i \in \mathcal{V}$ (recall that $\mathcal{V}$ is the set of nodes) will be updated as follows:

$$x_i(t+1) = \sum_{j \in \mathcal{V}} w_{ij} F_\theta(x_j(t), y_j(t)) \tag{9}$$

using the interpolated operator F_θ given in (8), where the coefficients w_{ij} are the elements of the mixing matrix $W = (w_{ij})$. Recall that each w_{ij} is nonnegative, and that w_{ij} is nonzero if and only if i and j are neighbors in $\mathcal{G}$. Moreover, for each $i = 1, \dots, n$, it holds that $\sum_{j \in \mathcal{V}} w_{ij} = 1$.

[4]The operator D_i is the local operator of agent i, meaning that the initialization step requires no communication between agents.

[5]The algorithm can be terminated at any time, upon reaching a certain number of iterations or certain accuracy.

We note that, even though we apply the consensus/combination step after evaluating the algorithm F, as in (9), our results still remain valid (but with slightly different constants) if instead using the update rule

$$x_i(t+1) = F_\theta\left(\sum_{j\in\mathcal{V}} w_{ij}x_j(t), y_j(t)\right)$$

which applies the algorithm after the consensus step. These two different update rules are in fact almost equivalent, in the sense that "shifting" an orbit generated by the second update rule by W gives the same result as using the first rule, but with the initial points $x_i(0)$ "shifted" by W.

Then, in Steps 3 and 4, the local estimates y_i of the global data operators will be similarly updated as

$$y_i(t+1) = \sum_{j\in\mathcal{V}} w_{ij}\left(y_j(t) + D_j x_j(t+1) - D_j x_j(t)\right).$$

Introducing the difference $D_j x_j(t+1) - D_j x_j(t)$, or some variation thereof, in the update step is an important approach to preserve convergence rate, e.g., as used in distributed gradient descent [34], [35].

The above steps can be more compactly expressed by introducing the stacked state (matrix)

$$x(t) = (x_1(t)^{\mathrm{T}}, \dots, x_n(t)^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{n\times d}$$

and the stacked data state estimates

$$y(t) = (y_1(t)^{\mathrm{T}}, \dots, y_n(t)^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{n\times m}.$$

We notice that the spectral properties of W carry over to the stacked state matrices; specifically, for any matrix $M \in \mathbb{R}^{n\times n'}$ with arbitrary n', it holds that

$$\begin{aligned}\|W(M - \mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}M)\|_{\mathrm{F}}^2 &= \sum_{j=1}^{n'} \|W\left(M_{\cdot,j} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}M_{\cdot,j}\right)\|_2^2 \\ &\le \sigma\sum_{j=1}^{n'} \|M_{\cdot,j} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}M_{\cdot,j}\|_2^2 \\ &= \sigma\|M - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}M\|_{\mathrm{F}}^2. \end{aligned} \tag{10}$$

We note that inequality (10) implies that the consensus-based information spreading in the network has a linear convergence rate. This is a bottleneck for any decentralization methods based on consensus, even the original centralized algorithm has a superlinear convergence rate (see Remark 1 and the conjecture following it).

We further extend the operators D and F_θ to the stacked state form, through the induced operators $\widehat{D} : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times m}$ and $\widehat{F}_\theta : \mathbb{R}^{n\times d} \times \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times d}$, given by

$$\widehat{D}x(t) = \left(Dx_1(t)^{\mathrm{T}}, \dots, Dx_n(t)^{\mathrm{T}}\right)^{\mathrm{T}}, \quad \text{and}$$

$$\widehat{F}_\theta(x(t), y(t)) = \left(F_\theta(x_1(t), y_1(t))^{\mathrm{T}}, \dots, F_\theta(x_n(t), y_n(t))^{\mathrm{T}}\right)^{\mathrm{T}}$$

respectively. We also introduce the operator

$$\widehat{\Phi}(x(t)) = \left(\Phi(x_1(t)), \dots, \Phi(x_n(t))\right)^{\mathrm{T}} = \widehat{F}(x(t), \widehat{D}x(t))$$

and the stacked local data operator

$$D_{\mathrm{loc}}x(t) = \left(D_1x_1(t)^{\mathrm{T}}, \dots, D_nx_n(t)^{\mathrm{T}}\right)^{\mathrm{T}}$$

which is simply the matrix whose rows are given by the local data operators evaluated at the local iterate, i.e., $D_i x_i(t)$.

With the above stacked states, Algorithm 1 can then be compactly expressed as

$$\begin{aligned} x(t+1) &= W\widehat{F}_\theta(x(t), y(t)) \\ y(t+1) &= W\left(y(t) + D_{\mathrm{loc}}x(t+1) - D_{\mathrm{loc}}x(t)\right). \end{aligned} \tag{11}$$

In the next section, we will show that, with proper tuning of the interpolation variable θ (the *step size*), the distributed algorithm given in (11)—that is, Algorithm 1—converges linearly to the same fixed point as the original algorithm.

IV. Main Theoretical Results

Our main results show that, given the assumptions in Section II, and with proper tuning of the interpolation parameter θ in the proposed distributed Algorithm 1 in Section III, Algorithm 1 converges exponentially (see the statement of the results for the exact definition). That is, Algorithm 1 maintains the same convergence properties as the original algorithm in (3).

We defer the proofs of the results to Section VI, to simplify the presentation in this section. These results will be further applied to important examples and applications in optimization, see Section V.

Throughout this section, we will assume that the operators F and D, that fully determine the original algorithm, are given. Moreover, we shall assume that these operators satisfy Assumptions 1–4. To state our results, we need to introduce the system matrix

$$\Gamma = \Gamma(F) := \begin{bmatrix} \sigma L_x & \sigma L_y & 0 \\ 2\sigma L_D(1+\Lambda) & \sigma(1+\Lambda) & \sigma L_D\delta \\ 2\Lambda & L_y & r \end{bmatrix} \tag{12}$$

where we have set $\Lambda = L_yL_D$, and the constants r, δ, L_x, L_y, L_D, and σ are the ones given in Assumptions 1–4. The importance of this matrix Γ will become clear in the statements of the results and their proofs. In particular, the convergence rate of Algorithm 1 is bounded from above by the spectral radius of Γ.

We note that, in general, convergence of the distributed Algorithm 1 can only be guaranteed using some degree of interpolation, i.e., that $0 < \theta < 1$. This is analogous to how the step size must be tuned for gradient descent methods, and in those cases θ actually corresponds directly to the step size. Our first result, Theorem 1, establishes a sufficient condition for the distributed Algorithm 1 to converge, and in particular achieve linear convergence, without any interpolation, i.e., with $\theta = 1$. In fact, this same result can be applied to the interpolated algorithm F_θ, to derive sufficient conditions on the interpolation parameter that ensure convergence of the distributed Algorithm 1. That is our second result, Theorem 2.

Our third result Theorem 3 is a specialization of Theorem 2, which lets us derive a better upper bound for the interpolation parameter θ. This is the result that will be used later in Section V, where we apply our results to different optimization algorithms.

Lastly, Theorem 4 shows that our results can be immediately applied to establish convergence for their proximal counterparts. This result demonstrates that convergence for the proximal versions of these algorithms can be be derived directly from their nonproximal originals, with the same convergence rate, just like in the nondistributed (centralized) case. With no further ado, we present our first result:

Theorem 1 (No interpolation case): Suppose that the maps $F, \{D_i\}_i$, and W satisfy Assumptions 1–4. If the spectral radius $\rho = \rho(\Gamma)$ of Γ is strictly less than one, i.e.,

$$\rho(\Gamma) < 1 \tag{13}$$

then Algorithm 1 with $\theta = 1$ converges to the unique solution x^* with exponentially rate ρ. In particular, there is a positive constant C, independent of the initial value of $x(0)$, such that

$$\begin{aligned}\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}} \leq C\rho^t \left(\|x(0) - \mathbf{1}_n x^*\|_{\mathrm{F}}\right. \\ \left. + \|x(0) - \mathbf{1}_n \overline{x}(0)\|_{\mathrm{F}} + \|y(0) - \mathbf{1}_n \overline{y}(0)\|_{\mathrm{F}}\right)\end{aligned} \tag{14}$$

for any $t > 0$ and initial value $x(0)$, if $y(0) = D_{\mathrm{loc}} x(0)$.

In general, the convergence condition (13) may not hold and (the distributed) Algorithm 1 may fail to converge. However, the next result shows that it is always possible to tune $\theta \in (0, 1]$ so that Algorithm 1 converges linearly. It follows from Theorem 1, by considering the interpolated operator F_θ, which satisfies the assumptions with the constants $r_\theta = (1-\theta) + \theta r, \delta_\theta = \theta\delta, L_{x;\theta} = (1-\theta) + \theta L_x, L_{y;\theta} = \theta L_y$, and $\Lambda_\theta = L_{y;\theta} L_D = \theta\Lambda$. See the proof in Section VI-B for details.

Theorem 2: Under the same assumptions as in Theorem 1, Algorithm 1 converges exponentially fast for any $\theta \in (0, 1)$ that satisfies all of the conditions

$$\begin{aligned}\theta &< \left(\frac{1-\sigma}{2\Lambda + \sigma(L_x - 1) + 2\sigma L_D\sqrt{L_y}(1+\Lambda)}\right)^2 \\ \theta &< \left(\frac{1-\sigma}{\sigma\Lambda + (1+\sigma)\sqrt{L_y}}\right)^2, \text{ and} \\ \theta &< \left(\frac{1-r}{\sigma\delta L_D\sqrt{L_y}}\right)^2.\end{aligned} \tag{15}$$

That is, for any interpolation parameter satisfying these conditions, the bound (14) in Theorem 1 holds with ρ replaced by $\rho_\theta = \rho(\Gamma(F_\theta))$, the spectral radius of $\Gamma(F_\theta)$.

We note that the conditions (15) given in Theorem 2 are in general not optimal for choosing the interpolation parameter θ, as discussed in the following remark.

Remark 5: The conditions in (15) were derived using an upper bound of the spectral radius, obtained through application of Gershgorin's circle theorem. These conditions are therefore, in general, not sharp.

With the additional assumption that $0 \leq L_x \leq 1$, which holds in many interesting examples, it is possible to obtain significantly better convergence rates, as in the following result.

Theorem 3: Under the same assumptions as in Theorem 1, assuming in addition that $0 \leq L_x \leq 1$, it follows that Algorithm 1 converges exponentially fast for any $0 < \theta \leq 1$ that further satisfies

$$\theta \leq \min\left\{1, \frac{(1-\sigma)^2}{2\Lambda + (1-\sigma)2A}\right\}$$

where

$$A = 1 - r + \sigma\left(4\Lambda + 1 - L_x + \frac{5\Lambda\delta}{2(1-r)}\right)$$

with convergence rate

$$\rho_\theta = 1 - \theta\frac{1-r}{2}.$$

That is, the bound (14) in Theorem 1 holds with ρ replaced by this ρ_θ, which is an upper bound on the spectral radius of $\Gamma(F_\theta)$.

In many applications, such as proximal algorithms for regularization in learning, or constrained optimization, the map F is a composition of a "base" map with a proximal operator, such as a projection. In many cases, the proximal operator is a nonexpanding map, for example projection onto a convex set. For the next result, we shall assume that the map Φ is a contraction, i.e., that it satisfies

$$\|\Phi(x) - \Phi(z)\|_2 \leq r\|x - z\|_2, \quad \forall x, z \in \mathbb{R}^d. \tag{16}$$

Given a nonexpanding[6] operator $P : \mathbb{R}^d \to \mathbb{R}^d$, set $G(x, y) = P \circ F(x, y)$ and consider the composition

$$\Phi_G(x) = P \circ \Phi(x) = P \circ F(x, Dx) = G(x, Dx) \tag{17}$$

which is again a contraction map with a unique fixed point $\widetilde{x}^*$ (which is in general different from the fixed point x^* of the original map Φ, e.g., as in projective gradient descent when the constraint set does not contain the stationary point). In this case, the following result established convergence of Φ_G directly from convergence of Φ.

Theorem 4: Let operators F, D, and W be given, which satisfy Assumptions 1–4. Additionally, assume that Φ is a contraction map, as in (16). Then Theorems 1–3 all hold mutatis mutandis for the operators G and Φ_G, defined in (17), in place of F and Φ, and with the same conditions on θ and convergence rate ρ.

Moreover, the maps G and Φ_G given in (17) satisfy Assumptions 1–4, with the exact same constants as the maps F and Φ, for any nonexpanding map $P : \mathbb{R}^d \to \mathbb{R}^d$.

Remark 6: The assumption that Φ is a contraction in Theorem 4 ensures that the composition $P \circ \Phi$ is also a contraction with a unique fixed point. If Φ is only assumed to satisfy the weaker Assumption 1, i.e., linear convergence, the composition may have (among other complications) more than one fixed point.

V. Applications to Distributed Optimization and Machine Learning

We now illustrate our theoretical results on distributed optimization algorithms. Consider a network with n agents, each with its own local differentiable objective function $f_i : \mathbb{R}^d \to \mathbb{R}$. Additionally, suppose that they have a common closed, properly

[6] P is nonexpanding if $\|Pu - Pv\|_2 \leq \|u - v\|_2$ for any pair of vectors $u, v \in \mathbb{R}^d$

convex function $g : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$, which may be nondifferentiable. The goal of the nodes is to jointly solve the optimization problem

$$\underset{x}{\text{minimize}} f(x) + g(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) + g(x). \tag{18}$$

Such problems are often solved using proximal gradient descent (or some variation thereof), i.e., the update rule

$$z(t+1) = \text{prox}_{\eta g}\left(z(t) - \eta\nabla f(z(t))\right)$$
$$= \arg\min_z \left\{\eta g(z) + \frac{1}{2}\left\|z - (z(t) - \eta\nabla f(z(t))\right\|_2^2\right\}.$$

The typical function of g is to represent regularization terms and/or problem constraints. In the constrained optimization case, the proximal operator would be a projection operator. When $g \equiv 0$, the proximal operator is the identity operator. Therefore, this problem formulation contains (nonconstrained) smooth optimization as a special case.

Throughout this section we assume that $f : \mathbb{R}^d \to \mathbb{R}^d$ is L-smooth and μ-strongly convex, i.e., for any pair $x, y \in \mathbb{R}^d$

$$\mu\|x - y\|_2 \leq \|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2. \tag{19}$$

We will also assume that each f_i is L-smooth and convex. However, they need not be μ-strongly convex individually, as long as their average f is. That is, only some of them, but not all, must be strongly convex.

In this strongly convex case, it turns out that the proximal operator $\text{prox}_{\eta g}$ is nonexpansive if $0 < \eta \leq 2/(\mu + L)$. This is the only property of the proximal operator that we will use in the rest of this section. The constant

$$\kappa = \frac{L}{\mu}$$

is known as the *condition number* of the problem. Under these assumptions on f, there is a unique global minimum x^* (satisfying $\nabla f(x^*) = 0$), and many (centralized) algorithms for solving Problem (18) converge linearly. We will now show how several of these algorithms can be transformed into distributed algorithms, using our framework, while also maintaining the essential convergence properties of the original algorithm. We will also compare our results to the state of the art for each specific example. Note in the following subsections how the proximal cases will follow immediately from the nonproximal version (directly using the nonexpansiveness property).

A. Proximal Gradient Descent

Standard gradient descent can be represented by setting

$$F(x, y) = F(x, y; \eta) = x - \eta y, \text{ and } D_i(x) = \nabla f_i(x) \tag{20}$$

where η is the familiar step size, and $D = \frac{1}{n}\sum_{i=1}^n D_i(x) = \nabla f(x)$. Note that there are also other ways of representing it.

If the step size η satisfies $0 \leq \eta \leq 2/(\mu + L)$, (standard) gradient descent is a contraction map with contraction factor $r = 1 - \eta\mu$, see [23]. It therefore follows that F, D, and Φ satisfy Assumptions 1–3 with $r = 1 - \eta\mu$, $L_x = 1$, $L_y = \eta$, $L_D = L$, and $\delta = 1 + r \leq 2$ (here, we simply use the worst bound for δ). Proximal gradient descent is simply the composition of F with the proximal operator $\text{prox}_{\eta g}$, which in this case is a nonexpanding map. The resulting distributed version of proximal gradient descent is then

$$x_i(t+1) = \sum_{i=1}^{k} w_{ij}\left((1-\theta)x_i(t) + \theta\,\text{prox}_{\eta g}\left(x_i(t) - \eta y_i(t)\right)\right)$$

$$y_i(t+1) = \sum_{i=1}^{k} w_{ij}\left(y_i(t) + \nabla f_i(x_i(t+1)) - \nabla f_i(x_i(t))\right).$$

Since the gradient descent operator is a contraction (for the chosen step sizes), Theorem 4 applies. Therefore, the convergence rate for this distributed proximal gradient descent is the same as we would get for the distributed nonproximal version. Therefore, we only need to analyze the special case of standard (nonproximal) gradient descent to obtain the convergence rate also for the proximal version.

Set $\eta = 2/(\mu + L)$, so that $1 - r = 2\mu/(\mu + L) \leq 2$ and $\Lambda = 2L/(\mu + L) \leq 2$. Since the conditions of Theorem 3 are satisfied, Algorithm 1 converges with rate

$$\rho_\theta = 1 - \theta\frac{\mu}{\mu + L}$$

under the condition that

$$\theta \leq \min\left\{1, \frac{(1-\sigma)^2}{2\eta L + 2(1-\sigma)\left(\eta\mu + \sigma(4\eta L + 5\kappa)\right)}\right\}.$$

As a side remark, notice that setting $\sigma = 0$ (corresponding to a fully connected network graph) recovers the centralized convergence rate. This condition is certainly met if

$$\theta = \frac{(1-\sigma)^2}{14(1+\kappa)}$$

giving the convergence rate

$$\rho_\theta = 1 - \frac{(1-\sigma)^2}{14(1+\kappa)^2}.$$

In the case, when $g \equiv 0$, i.e., smooth convex optimization, the proximal operator is the identity operator and the distributed algorithm reduces to the algorithms proposed in [33], [34], and [35] (the order of the consensus step W varies, but from a dynamical point of view they are essentially equivalent). Our results are consistent with their results, i.e., the existing state of the art, for this particular class of distributed algorithms, and static network graphs with doubly stochastic W. However, it should be noted that [34] also treats varying graphs, and that [35] also treats the nonstrongly convex case, both of which are beyond the present study.

While [32], [33], [34], and [35] do not treat treat the proximal case, distributed proximal gradient descent algorithms are also developed in [20], [39], [40], and [41], but their algorithm designs differs from ours. Note that our analysis does not explicitly involve subgradients, but directly uses on the nonexpansiveness property of the proximal operator.

B. Proximal Heavy Ball Method

The heavy ball method is given by the update rule

$$x(t+1) = x(t) - \alpha\nabla f(x(t)) + \beta(x(t) - x(t-1)) \tag{21}$$

where the constant $\beta \in [0, 1)$ represents the momentum strength, and can be tuned according to the problem [49, Sec.

3.2]. By setting $z(t) = (z_1(t), z_2(t))^{\mathrm{T}} = (x(t+1), x(t))^{\mathrm{T}} \in \mathbb{R}^{2d}$, the update rule in (21) can be alternatively expressed as

$$z(t+1) = Dz(t) := \begin{bmatrix} (1+\beta)I & -\beta I \\ I & 0 \end{bmatrix} z(t) - \begin{bmatrix} \alpha \nabla f(z_1(t)) \\ 0 \end{bmatrix} \tag{22}$$

i.e., with $F(x, y) = y$. The mean value theorem yields the inequality

$$\|Du - Dv\|_2 \leq \left\| \begin{bmatrix} (1+\beta)I - \alpha \nabla f(\zeta) & -\beta I \\ I & 0 \end{bmatrix} \right\| \cdot \|u - v\|_2$$

for some appropriate ζ. Let λ_i denote the eigenvalues of $\nabla f(\zeta)$. It can be seen that the eigenvalues of the matrix above are imaginary (nonreal) pairs with modulus β, if $\max_i (1 - \sqrt{\alpha \lambda_i})^2 \leq \beta \leq 1$. In particular, this holds if

$$\beta = \max\left(|1 - \sqrt{\alpha \mu}|, |1 - \sqrt{\alpha L}| \right).$$

Setting

$$\alpha = \frac{4}{\left(\sqrt{L} + \sqrt{\mu}\right)^2}$$

it then follows that $\|Du - Dv\|_2 \leq r \|u - v\|_2$, with $r = 1 - \sqrt{\alpha \mu} = 2/(\sqrt{\kappa} + 1) \in [0, 1)$. Thus, the Assumptions 1–3 are satisfied with $r = 1 - \sqrt{\alpha \mu}$, $L_x = 0$, $L_y = 1$, $L_D = r$, and $\delta = 1 + r \leq 2$ (we simply use the worst bound), and it follows that $\Lambda = r$. Notice that

$$\frac{r}{1-r} = \frac{\sqrt{\kappa} - 1}{2}.$$

According to the proposed decentralized algorithm, the proximal heavy ball method is given by

$$z_i(t+1) = \sum_{i=1}^{k} w_{ij} \left((1-\theta) z_i(t) + \theta \operatorname{prox}_{\eta g}\left(D z_i(t) \right) \right)$$

$$y_i(t+1) = \sum_{i=1}^{k} w_{ij} \left(y_i(t) + D z_i(t+1) - D z_i(t) \right) \tag{23}$$

where $Dz(\cdot)$ is defined in (22). Again, since the heavy ball method is a contraction for this choice of α and β, Theorem 4 gives that the proximal version of the distributed heavy ball method enjoys the exact same convergence rate as the nonproximal version. Therefore, we only need to analyze the nonproximal version. To the best of our knowledge, this is the first example of a distributed proximal heavy ball method, with linear convergence guarantees, in the literature.

Theorem 3 applies to show that the distributed heavy ball method resulting from Algorithm 1 (and its proximal version) converges with the rate

$$\rho_\theta = 1 - \frac{(1-\sigma)^2}{14\sqrt{\kappa}} \frac{2}{(\sqrt{\kappa}+1)}, \text{ for } \theta = \frac{(1-\sigma)^2}{14\sqrt{\kappa}}$$

which has the same dependence on the condition number as centralized (nondistributed) standard gradient descent. That is, distributed heavy ball method is as good as centralized standard gradient descent. This also proves conclusively that the heavy ball method performs better than, or at least as good as, standard gradient descent, also in the distributed setting. To the best of our knowledge, this is the first time this has been proved analytically.

A similar distributed heavy ball method is introduced in [37], but it does not treat the proximal case. Since the convergence rate in [37] is given by a complicated expression, it is difficult to compare it to our work. It seems as if the convergence rate in [37] is inversely proportional to the number of nodes, while ours is not. Since our convergence rate is very explicit and simple, that is another benefit of our approach. However, we should mention that [37] also treats a wide range of parameters α and β.

C. Proximal Newton's Method

Let $\mathrm{PSD}_d(\mu)$ denote the subset of positive definite matrices with smallest eigenvalue $\mu > 0$, i.e., matrices A that satisfy $x^{\mathrm{T}} A x \geq \mu x^{\mathrm{T}} x$. Note that this space is closed under taking weighted averaging, and therefore preserved by the consensus step. Define $F : \mathbb{R}^d \times \mathbb{R}^d \times \mathrm{PSD}_d(\mu) \to \mathbb{R}^d$ by

$$F(x, y_1, y_2) = x - \eta y_2^{-1} y_1$$

where the step size $\eta \in (0, 1]$. Assuming that each function f_i is also twice continuously differentiable, Newton's method can be represented by F together with the two data operators

$$D_{i,1}(x) = \nabla f_i(x), \text{ and } D_{i,2} = \nabla^2 f_i(x).$$

To simplify the rest of the analysis, and ensure that Newton's method is contractive, assume that each Hessian $\nabla^2 f_i$ is constant.[7] That is

$$\nabla^2 f_i = A_i$$

for some (strictly) positive definite matrix A_i. Note that the matrices A_i can vary between nodes. This special case corresponds to least square minimization, which is a common problem in practical applications.

To fit this example into our current framework, represent (y_1, y_2) and $(D_i^{(1)}, D_i^{(2)})$ (for each i) as a single vector variable y, and a single data operator D_i, by combining and stacking the columns of the vector y_1 and the matrix y_2 on top of one another to create a $d(d+1)$-dimensional vector (and analogously for the data operators). If we can show that F is Lipschitz continuous, it follows that it can be extended globally with the same Lipschitz coefficients, by Kirszbraun theorem (see, e.g., [51, p. 201]).

The operator F is clearly 1-Lipschitz in the x-component. Since the second variable y_2 is a matrix in $\mathrm{PSD}_d(\mu)$, F satisfies the Lipschitz conditions

$$\|F(x, y_1, y_2) - \widetilde{F}(x, u_1, y_2)\|_2 \leq \frac{\eta}{\mu} \|y_1 - u_1\|_2.$$

and

$$\begin{aligned} \|F(x, y_1, y_2) - F(x, y_1, u_2)\|_2^2 &\leq \eta \|u_2^{-1}(u_2 - y_2) y_2^{-1}\|_{\mathrm{F}}^2 \\ &\leq \frac{\eta d}{\mu^2} \|y_2 - u_2\|_{\mathrm{F}}^2. \end{aligned}$$

The last inequality follows since $\|A\|_2 \leq \|A\|_{\mathrm{F}} \leq \sqrt{\operatorname{rank}(A)} \cdot \|A\|_2$ for any matrix A.

In this case, the relevant constants are $r = 1 - \eta$, $\delta = \eta$, $L_x = 1$, $L_y = \eta/\mu \cdot \max(1, d/\mu)$, $L_D = L$. Setting $\eta = 1$, and the

[7] Note that such an assumption is an evident way to avoid the sublinear convergence regions. Because only in the neighborhood of the optimal solution, the Newton's method converges super-linearly and damped Newton's method linearly (see [50, Sec. 9.5] for reference).

distributed proximal Newton's method becomes

$$x_i(t+1) = \sum_{i=1}^{k} w_{ij}\Big((1-\theta)x_i(t) + \theta\,\mathrm{prox}_{\eta g}\Big(x_i(t) - \nabla^2 f_i(x_i(t))^{-1}\nabla f_i(x_i(t))\Big)\Big)$$

$$y_i(t+1) = \sum_{i=1}^{k} w_{ij}\left(y_i(t) + Dz_i(t+1) - Dz_i(t)\right)$$

where $Dz_i(t) = (\nabla f_i(x_i(t)), \nabla^2 f_i(x_i(t)) \in \mathbb{R}^{d\times(d+1)}$. Then applying Theorem 3, we get the convergence rate for the distributed (proximal) Newton's method as

$$\rho_\theta = 1 - \frac{(1-\sigma)^2}{44\kappa\max\left(1,\frac{d}{\mu}\right)},\ \text{for}\ \theta = \frac{(1-\sigma)^2}{22\kappa\max\left(1,\frac{d}{\mu}\right)}.$$

We remark that, by extending our results to weighted norms, or multivariable data, the factor d/μ can be removed, and replaced by a term depending on the Lipschitz constant of the Hessian.

VI. Proof of the Main Theorems

In the rest of this section, it is assumed that the operators F, D, Φ, and W satisfy Assumptions 1–4. To simplify notation, introduce the shorthand notation

$$X(t) = \|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}$$
$$Y(t) = \|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}},\ \text{and}$$
$$Z(t) = \|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}$$

where x^* is the fixed point of the mapping $\Phi(\cdot)$. We see that global convergence of the local values $x_i(t)$ to the solution is controlled by $Z(t)$. The following *three key lemmas* bound the time evolution of these variables X, Y, and Z in terms of one another. Their proofs appear in the last three subsections of this section.

Lemma 1: Suppose that Assumptions 1–3 hold. Then, for every $t \geq 0$, it holds that

$$\|x(t+1) - \mathbf{1}_n x^*\|_{\mathrm{F}} \leq r\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}} + L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} + 2\Lambda\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}. \quad (24)$$

Lemma 2: Suppose that Assumptions 2 and 4 hold. Then, for every $t \geq 0$, it holds that

$$\|x(t+1) - \mathbf{1}_n\overline{x}(t+1)\|_{\mathrm{F}} \leq \sigma L_x\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}} + \sigma L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}}. \quad (25)$$

The latter can be bounded instead as follows.

Lemma 3: Suppose that Assumptions 1–4 hold. Then, for every $t \geq 0$, it holds that

$$\|y(t+1) - \mathbf{1}_n\overline{y}(t+1)\|_{\mathrm{F}} \leq \sigma(1+\Lambda)\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} + 2\sigma L_D(1+\Lambda)\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}} + \sigma L_D\delta\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}. \quad (26)$$

Recall that the main results in this article depend on the system matrix

$$\Gamma = \Gamma(F) = \begin{bmatrix} \sigma L_x & \sigma L_y & 0 \\ 2\sigma L_D(1+\Lambda) & \sigma(1+\Lambda) & \sigma L_D\delta \\ 2\Lambda & L_y & r \end{bmatrix} \quad (27)$$

given in Section IV. Lemmas 1–3 altogether imply the recurrence relations

$$\begin{bmatrix} X(t+1) \\ Y(t+1) \\ Z(t+1) \end{bmatrix} \leq \Gamma \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} \quad (28)$$

where "$\leq$" means elementwise less than or equal to. With this recurrence relation, we are now ready to prove the results in Section IV, starting with Theorem 1.

A. Proof of Theorem 1

The recurrence relation in (28) shows that the convergence of the algorithm is controlled by the spectral radius of the matrix Γ, given in (27). Since L_y and L_D are positive, the matrix Γ is an irreducible nonnegative matrix. Therefore, the spectral radius of Γ is a simple eigenvalue of Γ, by the Perron–Frobenius Theorem (see, e.g., [52, Th. 8.4.4]). Thus, if $\rho = \rho(\Gamma)$ is its spectral radius, then there is a constant C which is independent of the initial values $X(0), Y(0)$, and $Z(0)$, such that $X(t), Y(t)$, and $Z(t)$ are all bounded by

$$\begin{pmatrix} X(t) \\ Y(t) \\ Z(t) \end{pmatrix} \leq C\rho^t \begin{pmatrix} X(0)+Y(0)+Z(0) \\ X(0)+Y(0)+Z(0) \\ X(0)+Y(0)+Z(0) \end{pmatrix}.$$

Thus, they all converge to 0 exponentially fast (with rate ρ), provided that the spectral radius is strictly less than one, i.e., $\rho < 1$. This concludes the proof.

B. Proof of Theorem 2

Suppose that F satisfies Assumptions 1 and 2, with the constants $0 \leq r < 1$, $0 \leq \delta$, $0 \leq L_x$, and $0 \leq L_y$. Consider the interpolated operator F_θ, and note that the maps $\{D_i\}_i$ and the mixing matrix W (and hence also Assumptions 3 and 4) are unaffected by the interpolation. The interpolation formula

$$\Phi_\theta(x) = \theta\Phi(x) + (1-\theta)x$$

directly shows that Φ_θ satisfies Assumption 1 with the constants $r_\theta = (1-\theta) + \theta r = 1 - \theta(1-r)$ and $\delta_\theta = \theta\delta$. Similarly, the interpolation formula

$$F_\theta(x,y) = x - \theta(x - F(x,y)) = (1-\theta)x + \theta F(x,y)$$

directly shows that F_θ satisfies Assumption 2 with the constants $L_{x;\theta} = (1-\theta) + \theta L_x = 1 + \theta(L_x - 1)$ and $L_{y;\theta} = \theta L_y$, respectively. Therefore, Theorem 1 also applies to F_θ, with $\Lambda_\theta = L_{y;\theta}L_D = \theta\Lambda$. For each θ, the system matrix Γ in (12) is then given by

$$\Gamma_\theta = \Gamma(F_\theta) = \begin{bmatrix} \sigma L_{x;\theta} & \sigma L_{y;\theta} & 0 \\ 2\sigma L_D(1+\Lambda_\theta) & \sigma(1+\Lambda_\theta) & \sigma L_D\delta_\theta \\ 2\Lambda_\theta & L_{y;\theta} & r_\theta \end{bmatrix}.$$

While it is possible to directly compute the eigenvalues of this matrix, as the roots of a third degree polynomial, the resulting expressions are impractical. Instead, we shall bound them from above.

When $L_y = 0$, notice that the nondiagonal elements of the top row are both zero, and therefore that Γ_θ has the eigenvalues $\lambda_1 = \sigma, \lambda_2 = \sigma(1+\theta(L_x-1))$, and $\lambda_3 = 1-\theta(1-r)$. These are all nonnegative, and since both σ and r are strictly less than one by assumption, both of the eigenvalues λ_1 and λ_3 are strictly less than one. Thus, if $L_y = 0$, the spectral radius is less than one only if $\lambda_2 < 1$, i.e., if either $L_x \leq 1$ ($0 \leq L_x$ by assumption) or

$$\theta < \min\left\{1, \frac{1-\sigma}{\sigma|L_x - 1|}\right\}.$$

When $0 < L_y$, we instead introduce the change of variables $\widetilde{Y} = \sqrt{L_{y;\theta}}Y$, and shall use Gershgorin's circle theorem to bound the spectral radius of the system matrix. With this coordinate change, we see that

$$\begin{bmatrix} X(t+1) \\ \widetilde{Y}(t+1) \\ Z(t+1) \end{bmatrix} \leq \widetilde{\Gamma}_\theta \begin{bmatrix} X(t) \\ \widetilde{Y}(t) \\ Z(t) \end{bmatrix}$$

with

$$\widetilde{\Gamma}_\theta = \begin{bmatrix} \sigma L_{x;\theta} & \sigma\sqrt{L_{y;\theta}} & 0 \\ 2\sigma\sqrt{L_{y;\theta}}L_D(1+\Lambda_\theta) & \sigma(1+\Lambda_\theta) & \sigma\sqrt{L_{y;\theta}}L_D\delta_\theta \\ 2\Lambda_\theta & \sqrt{L_{y;\theta}} & r_\theta \end{bmatrix}.$$

Applying Gershgorin's circle theorem, it follows that the eigenvalues of $\widetilde{\Gamma}_\theta$ (and hence also its spectral radius, since the eigenvalues are all positive) are bounded from above by the greatest of its column sums. It follows that the algorithm converges (linearly) for any $0 < \theta$ that satisfies:

$$\begin{aligned} 2\Lambda_\theta + \sigma\left(L_{x;\theta} + 2\sqrt{L_{y;\theta}}L_D(1+\Lambda_\theta)\right) &< 1 \\ \sqrt{L_{y;\theta}} + \sigma\left(\sqrt{L_{y;\theta}} + (1+\Lambda_\theta)\right) &< 1 \\ r_\theta + \sigma\sqrt{L_{y;\theta}}L_D\delta_\theta &< 1. \end{aligned}$$

After some straight-forward algebraic manipulations, recalling that $0 < \theta \leq 1$, and thus, also $\theta \leq \sqrt{\theta}$, we arrive at the weaker conditions

$$\begin{aligned} \theta &< \left(\frac{1-\sigma}{2\Lambda + \sigma(L_x-1) + 2\sigma L_D\sqrt{L_y}(1+\Lambda)}\right)^2 \\ \theta &< \left(\frac{1-\sigma}{\sigma\Lambda + (1+\sigma)\sqrt{L_y}}\right)^2 \\ \theta &< \left(\frac{1-r}{\sigma\delta L_D\sqrt{L_y}}\right)^2. \end{aligned} \tag{29}$$

Notice that the condition $\theta < \max(1, (1-\sigma)/(L_x-1))$, for the case when $L_y = 0$, is also implied by the above conditions (specifically, the first one). Therefore, the conditions in (29) ensure that the spectral radius is strictly less than 1, for all cases. The statement now follows from Theorem 1, which concludes the proof.

C. Proof of Theorem 3

The following proof involves many detailed computations, which is why we have divided it into two smaller steps.

Step 1. *Bounding the spectral radius of the matrix* Γ.

Lemma 4: Assuming that $0 \leq L_x \leq 1$, the spectral radius $\rho(\Gamma)$ of the matrix

$$\Gamma = \begin{bmatrix} \sigma L_x & \sigma L_y & 0 \\ 2\sigma L_D(1+\Lambda) & \sigma(1+\Lambda) & \sigma L_D\delta \\ 2\Lambda & L_y & r \end{bmatrix}$$

is bounded by

$$\rho(\Gamma) \leq \max\left\{\frac{1+r}{2}, \frac{1}{2}\sigma\left(1+\Lambda+L_x+\sqrt{q(\Lambda)}\right) + \frac{5\sigma\Lambda\delta}{(1-r)}\right\}$$

with

$$q(\Lambda) := (3\Lambda - L_x + 1)^2 + 4\Lambda(1+L_x).$$

Proof: Since the matrix is nonnegative, its spectral radius coincides with its largest real eigenvalue, which must be nonnegative. It is therefore sufficient to consider only nonnegative real solutions to its characteristic equation, which is given by

$$\begin{aligned} p(\lambda) := {} & (\lambda - \sigma L_x)(\lambda - \sigma(1+\Lambda))(\lambda - r) - 2\sigma^2\Lambda^2\delta \\ & - \sigma\Lambda\delta(\lambda - \sigma L_x) - 2\sigma^2\Lambda(1+\Lambda)(\lambda - r) = 0. \end{aligned}$$

Gather the terms containing the factor $\lambda - r$, and rewrite

$$\begin{aligned} p(\lambda) = {} & (\lambda - r)\left((\lambda - \sigma L_x)(\lambda - \sigma(1+\Lambda)) - 2\sigma^2\Lambda(1+\Lambda)\right) \\ & - \sigma^2\Lambda^2\delta - \sigma\Lambda\delta(\lambda - \sigma L_x + \sigma\Lambda) = 0. \end{aligned}$$

Set

$$q(\Lambda) := (3\Lambda - L_x + 1)^2 + 4\Lambda(1+L_x)$$

and notice that

$$\begin{aligned} & (\lambda - \sigma)(\lambda - \sigma(1+\Lambda)) - 2\sigma^2\Lambda(1+\Lambda) \\ & \quad = \left(\lambda - \frac{1}{2}\sigma(1+\Lambda+L_x+\sqrt{q(\Lambda)})\right) \\ & \qquad \times \left(\lambda - \frac{1}{2}\sigma(1+\Lambda+L_x-\sqrt{q(\Lambda)})\right). \end{aligned}$$

Now, let $K \geq 0$ be some parameter to be decided, and set

$$\overline{\lambda} = \max\left\{\frac{1+r}{2}, \frac{1}{2}\sigma\left(1+\Lambda+L_x+\sqrt{q(\Lambda)}\right) + K\right\} < 0.$$

Since we assume that $0 \leq L_x \leq 1$, it follows that $0 \leq 3\Lambda - L_x + 1 \leq \sqrt{q(\Lambda)}$, and therefore that:

$$\begin{aligned} p(\lambda) \geq {} & (\lambda - \sigma(L_x - \Lambda)) \\ & \cdot \left[(\lambda - r)\left(\lambda - \frac{1}{2}\sigma\left(1+\Lambda+L_x+\sqrt{q(\Lambda)}\right) - \sigma\Lambda\delta\right]\right. \\ & - \sigma^2\Lambda^2\delta. \end{aligned}$$

We shall now derive a condition to ensure that $0 < p(\lambda)$, whenever $\overline{\lambda} < \lambda$. Since the spectral radius coincides with the largest positive real root of p (see the discussion in the beginning of this proof), this would then bound the spectral radius by $\overline{\lambda}$. Since $\sigma(1+L_x)/2 \leq \overline{\lambda}$, it holds for every $\lambda > \overline{\lambda}$ that

$$p(\lambda) \geq \frac{1}{4}\sigma(1 - L_x + \Lambda)\left[(1-r)K - \sigma\Lambda\delta\right] - \sigma^2\Lambda^2\delta.$$

That is, $0 < p(\lambda)$, provided that

$$\frac{1}{(1-r)}\left[\frac{4\sigma^2\Lambda^2\delta}{\sigma(1-L_x+\Lambda))}+\sigma\Lambda\delta\right] \le K$$

which can be weakened to

$$\frac{5\sigma\Lambda\delta}{(1-r)} = \frac{1}{(1-r)}\left[\frac{4\sigma^2\Lambda^2\delta}{\sigma\Lambda}+\sigma\Lambda\delta\right] \le K$$

since $0 \le 1 - L_x$. Thus, the spectral radius is bounded by $\overline{\lambda}$, which concludes the proof. □

The next step is to derive conditions on the constants in Γ that ensure that the spectral radius is less than one. This means deriving conditions for the interpolation parameter θ.

Step 2. *Conditions to ensure that the spectral radius of* Γ *is less than one.*

In this step, we will replace F by its interpolated operator F_θ, replacing all relevant coefficients by their interpolated counterparts. Recall (for the details, see the beginning of the proof of Theorem 2) that the interpolated operator F_θ satisfies Assumptions 1–2 with $L_{x;\theta} = (1-\theta) + \theta L_x$, $L_{y;\theta} = \theta L_y$, $r_\theta = (1-\theta) + \theta r$ and $\delta_\theta = \theta\delta$. Assumptions 3 and 4 are unaffected by interpolation.

Using Lemma 4, we can bound the spectral radius of the system matrix $\Gamma_\theta = \Gamma(F_\theta)$, for any given $0 < \theta \le 1$, by

$$\max\left\{\frac{1+r_\theta}{2}, \frac{1}{2}\sigma\left(1+\Lambda_\theta+L_{x;\theta}+\sqrt{q_\theta(\Lambda_\theta)}\right)+\frac{5\sigma\Lambda_\theta\delta_\theta}{(1-r_\theta)}\right\}$$

with

$$q_\theta(\Lambda_\theta) := (3\Lambda_\theta - L_{x;\theta} + 1)^2 + 4\Lambda_\theta(1+L_{x;\theta}).$$

Now, let us impose the condition that the spectral radius is in fact bounded (from above) by $(1+r_\theta)/2$, so that

$$\sigma\left(1+\Lambda_\theta+L_{x;\theta}+\sqrt{q_\theta(\Lambda_\theta)}\right)+\frac{5\sigma\Lambda_\theta\delta_\theta}{2(1-r_\theta)} \le 1+r_\theta. \quad (30)$$

Using the interpolation formulas for the constants, we can rewrite $q_\theta(\Lambda_\theta)$ as

$$\begin{aligned} q_\theta(\Lambda_\theta) &= \theta^2(3\Lambda+1-L_x)^2 + 4\theta\Lambda(2+\theta(L_x-1)) \\ &\le \theta^2(3\Lambda+1-L_x)^2 + 8\theta\Lambda. \end{aligned}$$

Since $\sqrt{a+b} \le \sqrt{a}+\sqrt{b}$ for positive real numbers, and $L_{x;\theta} \le 1$ the condition in (30) can be weakened as

$$\begin{aligned} &\sigma\left(2+\theta\Lambda+\theta(3\Lambda+1-L_x)+\sqrt{8\theta\Lambda}+\frac{5\theta\Lambda\delta}{2(1-r)}\right) \\ &\le 2-\theta(1-r). \end{aligned}$$

Rearranging, we get

$$\frac{\theta}{2}\left(1-r+\sigma\left(4\Lambda+1-L_x+\frac{5\Lambda\delta}{2(1-r)}\right)\right)+\sqrt{\theta 2\Lambda} \le 1-\sigma$$

which is satisfied if

$$\sqrt{\theta} \le \frac{-\sqrt{2\Lambda}+\sqrt{2\Lambda+(1-\sigma)2A}}{A}$$

where we have set

$$A = 1-r+\sigma\left(4\Lambda+1-L_x+\frac{5\Lambda\delta}{2(1-r)}\right).$$

This is certainly satisfied if θ satisfies

$$\sqrt{\theta} \le \frac{1-\sigma}{\sqrt{2\Lambda+(1-\sigma)2A}} \le \frac{1}{A}\int_{2\Lambda}^{2\Lambda+(1-\sigma)2A}\frac{dx}{2\sqrt{x}}$$

$$= \frac{-\sqrt{2\Lambda}+\sqrt{2\Lambda+(1-\sigma)2A}}{A}$$

or if

$$0 < \theta \le \frac{(1-\sigma)^2}{2\Lambda+(1-\sigma)2A}.$$

That is, the spectral radius is bounded by

$$\frac{1+r_\theta}{2} = 1-\theta\frac{1-r}{2}$$

under these conditions on θ. This proves Theorem 3.

D. Proof of Theorem 4

Recall that P is a nonexpanding map, and that G is given by

$$G(x,y) = P \circ F(x,y).$$

Since P is nonexpanding, and Φ is a contraction, this G induces a contraction map

$$\Phi_G(x) = G \circ \Phi(x) = G(x, Dx)$$

i.e., an algorithm of the original form (3). Since Φ_G is a contraction, it has a unique fixed point $\widetilde{x}^*$, which in general is different from the fixed point x^* of Φ.

Since P is nonexpanding, one immediately sees that Φ_G together with this fixed point $\widetilde{x}^*$ satisfies Assumption 1 with the same constants r and δ as the original Φ (induced by F). Again, since P is nonexpanding, it also follows that G satisfies Assumption 2 with the same constants L_x and L_y as F. Assumptions 3 and 4 are unaffected by the addition of the operator P, and are therefore satisfied for G with the same constants as for F.

Therefore, the map G satisfies the conditions in Theorems 1–3, with the same constants as F. As a result, their conclusions apply mutatis mutandis to this map G.

E. Preparations for the Proofs of the Three Key Lemmas

Before proving the three key lemmas, Lemmas 1–3, we state and prove a few smaller lemmas that isolate certain computations and inequalities that appear in multiple places in their proofs.

Recall that each $y_i(0)$ is initialized to $D_i x_i(0)$ in the distributed Algorithm 1. This guarantees that the average of the estimates $y(t)$ actually equals to the average of the local data operators, as shown in the next lemma.

Lemma 5: Since each $y_i(0) = D_i x_i(0)$, it holds that

$$\overline{y}(t) = \frac{1}{n}\sum_{i=1}^{n} D_i x_i(t)$$

for every $t \ge 0$.

Proof: Since W preserves averaging, we see that

$$\begin{aligned} \overline{y}(t+1) &= \frac{1}{n}\sum_{i=1}^{n}\phi_i(t+1) \\ &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i(t)+D_i x_i(t+1)-D_i x_i(t)\right) \\ &= \overline{y}(t)+\frac{1}{n}\sum_{i=1}^{n}\left(D_i x_i(t+1)-D_i x_i(t)\right). \end{aligned}$$

By induction, it follows that:

$$\overline{y}(t+1) = \overline{y}(0) + \frac{1}{n}\sum_{i=1}^{n}\left(D_i x_i(t+1) - D_i x_i(0)\right).$$

Due to the initialization $y_i(0) = D_i x_i(0)$, it follows that $\overline{y}(0) = \frac{1}{n}\sum_{i=1}^{n} D_i x_i(0)$. Thus, the first and last terms cancel each other. □

The following lemma bounds the difference between $y_i(t)$ and the "true" (global) data operator for the local $x_i(t)$. This essentially controls "how far" the local updates are from the original update rule (applied to the local values $x_i(t)$).

Lemma 6: Under Assumption 3, the inequality

$$\begin{aligned}\|y(t) - \widehat{D}x(t)\|_{\mathrm{F}} \leq \|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\ + 2L_D\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}\end{aligned}$$

holds, where $\widehat{D}x(t) = \left(Dx_1(t)^T, \ldots, Dx_n(t)^T\right)^{\mathrm{T}}$.

Proof: We can bound the Frobenius norm of $y(t) - \widehat{D}x(t)$ by expanding

$$\begin{aligned}\|y(t) - \widehat{D}x(t)\|_{\mathrm{F}} \leq \|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\ + \|\mathbf{1}_n\overline{y}(t) - \mathbf{1}_n D\overline{x}(t)\|_{\mathrm{F}} \\ + \|\mathbf{1}_n D\overline{x}(t) - \widehat{D}x(t)\|_{\mathrm{F}}. \qquad (31)\end{aligned}$$

Since $\overline{y}(t) = \frac{1}{n}\sum_{i=1}^{n} D_i x_i(t)$ (by Lemma 5) and $D = \frac{1}{n}\sum_{i=1}^{n} D_i$, the second term can be bounded as

$$\begin{aligned}\|\mathbf{1}_n\overline{y}(t) - \mathbf{1}_n D\overline{x}(t)\|_{\mathrm{F}}^2 &= \sum_{k=1}^{n}\|\frac{1}{n}\sum_{i=1}^{n} D_i x_i(t) - D_i\overline{x}(t)\|_2^2 \\ &\leq \sum_{k=1}^{n}\frac{1}{n}\sum_{i=1}^{n} L_D^2\|x_i(t) - \overline{x}(t)\|_2^2 \\ &= L_D^2\sum_{k=1}^{n}\frac{1}{n}\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}^2 \\ &= L_D^2\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}^2.\end{aligned}$$

Taking square roots immediately yields

$$\|\mathbf{1}_n\overline{y}(t) - \mathbf{1}_n D\overline{x}(t)\|_{\mathrm{F}} \leq L_D\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.$$

As for the last term in (31), we have

$$\begin{aligned}\|\mathbf{1}_n D\overline{x}(t) - \widehat{D}x(t)\|_{\mathrm{F}}^2 &= \sum_{k=1}^{n}\|D\overline{x}(t) - Dx_k(t)\|_2^2 \\ &\leq \sum_{k=1}^{n} L_D^2\|\overline{x}(t) - x_k(t)\|_2^2 \\ &= L_D^2\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}^2.\end{aligned}$$

Again, taking square roots gives us the inequality

$$\|\mathbf{1}_n D\overline{x}(t) - \widehat{D}x(t)\|_{\mathrm{F}} = L_D\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.$$

The statement now follows. □

The next lemma controls how far the local updates are from the original update rule (applied to the local values $x_i(t)$).

Lemma 7: Under Assumption 2 and 3, it holds that

$$\begin{aligned}\|\widehat{F}(x(t), y(t)) - \widehat{\Phi}(x(t))\|_{\mathrm{F}} \leq L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\ + 2\Lambda\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.\end{aligned}$$

Proof: Writing it out, we have

$$\begin{aligned}&\|\widehat{F}(x(t), y(t)) - \widehat{\Phi}(x(t))\|_{\mathrm{F}}^2 \\ &= \sum_{k=1}^{n}\|F(x_k(t), y_k(t)) - F(x_k(t), Dx_k(t))\|_2^2 \\ &\leq L_y^2\sum_{k=1}^{n}\|y_k(t) - Dx_k(t)\|_2^2 \\ &= L_y^2\|y(t) - \widehat{D}x(t)\|_{\mathrm{F}}^2.\end{aligned}$$

Taking square roots, the conclusion follows from Lemma 6. □

F. Proofs of Lemmas 1–3

We are now ready to prove the three key lemmas.

Proof of Lemma 1: Consider first

$$x(t+1) - \mathbf{1}_n x^* = W\xi(t+1) - \mathbf{1}_n x^*.$$

Since $W\mathbf{1}_n x^* = \mathbf{1}_n x^*$ (each column lies in the subspace $\mathbb{R}\mathbf{1}_n$, which is fixed by W), and $\|W\|_2 \leq 1$ (this follows from Hölder's inequality, since all columns and rows have only positive elements and sum to one), we have

$$\|x(t+1) - \mathbf{1}_n x^*\|_{\mathrm{F}} \leq \|\xi(t+1) - \mathbf{1}_n x^*\|_{\mathrm{F}}.$$

Moreover, since $\xi(t+1) = \widehat{F}(x(t), y(t))$, we have

$$\begin{aligned}\|\xi(t+1) - \mathbf{1}_n x^*\|_{\mathrm{F}} \leq \|\widehat{\Phi}(x(t)) - \mathbf{1}_n x^*\|_{\mathrm{F}} \\ + \|\widehat{F}(x(t), y(t)) - \widehat{\Phi}(x(t))\|_{\mathrm{F}}.\end{aligned}$$

The first term satisfies

$$\begin{aligned}\|\widehat{\Phi}(x(t)) - \mathbf{1}_n x^*\|_{\mathrm{F}}^2 &= \sum_{k=1}^{n}\|\Phi(x_k(t)) - x^*\|_2^2 \\ &\leq r^2\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}^2.\end{aligned}$$

Combining this inequality with Lemma 7 in the Appendix, the assertion follows. □

Proof of Lemma 2

Recall that we want to prove that

$$\begin{aligned}\|x(t+1) - \mathbf{1}_n\overline{x}(t+1)\|_{\mathrm{F}} \leq \sigma L_x\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}} \\ + \sigma L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}}.\end{aligned}$$

Since averaging and W commute (due to the subspace $\mathbf{1}_n$ being invariant under W), it follows that $\mathbf{1}_n\overline{x}(t+1) = W\mathbf{1}_n\overline{\xi}(t+1)$, and thus:

$$x(t+1) - \mathbf{1}_n\overline{x}(t+1) = W\xi(t+1) - W\mathbf{1}_n\overline{\xi}(t+1).$$

Taking the Frobenius norm results in the inequality

$$\|x(t+1) - \mathbf{1}_n\overline{x}(t+1)\|_{\mathrm{F}} \leq \sigma\|\xi(t+1) - \mathbf{1}_n\overline{\xi}(t+1)\|_{\mathrm{F}}.$$

Note that $\overline{\xi}(t+1) - F(\overline{x}(t), \overline{y}(t))$ is the average of the rows in $\xi(t+1) - \mathbf{1}_n F(\overline{x}(t), \overline{y}(t))$. Therefore, Lemma 8 in the Appendix gives us

$$\begin{aligned}&\|\xi(t+1) - \mathbf{1}_n\overline{\xi}(t+1)\|_{\mathrm{F}} \\ &= \|\xi(t+1) - \mathbf{1}_n F\left(\overline{x}(t), \overline{y}(t)\right) \\ &\quad - \mathbf{1}_n\left(\overline{\xi}(t+1) - F(\overline{x}(t), \overline{y}(t))\right)\|_{\mathrm{F}} \\ &\leq \|\xi(t+1) - \mathbf{1}_n F\left(\overline{x}(t), \overline{y}(t)\right)\|_{\mathrm{F}}.\end{aligned}$$

Recalling that $\xi(t+1) = \widehat{F}(x(t), y(t))$, we can further expand

$$\begin{aligned}
&\|\xi(t+1) - \mathbf{1}_n\overline{\xi}(t+1)\|_{\mathrm{F}} \\
&\quad \le \|\widehat{F}(x(t), y(t)) - \widehat{F}(\mathbf{1}_n\overline{x}(t), y(t))\|_{\mathrm{F}} \\
&\qquad + \|\widehat{F}(\mathbf{1}_n\overline{x}(t), y(t)) - \mathbf{1}_n F(\overline{x}(t), \overline{y}(t))\|_{\mathrm{F}}.
\end{aligned}$$

The first of these terms satisfies

$$\begin{aligned}
&\|\widehat{F}(x(t), y(t)) - \widehat{F}(\mathbf{1}_n\overline{x}(t), y(t))\|_{\mathrm{F}}^2 \\
&\quad \le \sum_{k=1}^{n} \|F(x_k(t), y_k(t)) - F(\overline{x}(t), y_k(t))\|_2^2 \\
&\quad \le \sum_{k=1}^{n} L_x^2 \|x_k(t) - \overline{x}(t)\|_2^2 = L_x^2 \|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}^2.
\end{aligned}$$

Similarly, the second term satisfies

$$\begin{aligned}
&\|\widehat{F}(\mathbf{1}_n\overline{x}(t), y(t)) - \mathbf{1}_n F(\overline{x}(t), \overline{y}(t))\|_{\mathrm{F}}^2 \\
&\quad \le \sum_{k=1}^{n} \|F(\overline{x}(t), y_k(t)) - F(\overline{x}(t), \overline{y}(t))\|_2^2 \\
&\quad \le \sum_{k=1}^{n} L_y^2 \|y_k(t) - \overline{y}(t)\|_2^2 \\
&\quad = L_y^2 \|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}}^2.
\end{aligned}$$

Taking square roots of each of these inequalities and adding them together, the conclusion follows. □

Proof of Lemma 3: Recall that we wish to prove

$$\begin{aligned}
&\|y(t+1) - \mathbf{1}_n\overline{y}(t+1)\|_{\mathrm{F}} \\
&\quad \le \sigma(1+\Lambda)\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&\qquad + 2\sigma L_D(1+\Lambda)\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}} \\
&\qquad + \sigma L_D\delta\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}.
\end{aligned}$$

Set $g(t) = D_{\mathrm{loc}}(x(t+1)) - D_{\mathrm{loc}}(x(t))$, so that the ith row of $g(t+1)$ is given by $g_i(t+1) = D_i x_i(t+1) - D_i x_i(t)$. Since $\overline{y}(t) = \frac{1}{n}\sum_{i=1}^{n} D_i x_i(t)$, by Lemma 5, it follows that $\overline{g}(t+1) = \overline{y}(t+1) - \overline{y}(t)$. We can therefore expand

$$\begin{aligned}
y(t+1) - \mathbf{1}_n\overline{y}(t+1) &= W(y(t) - \mathbf{1}_n\overline{y}(t)) \\
&\quad + W(g(t+1) - \mathbf{1}_n\overline{g}(t+1))
\end{aligned}$$

so that

$$\begin{aligned}
\|y(t+1) - \mathbf{1}_n\overline{y}(t+1)\|_{\mathrm{F}} &\le \sigma\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&\quad + \sigma\|g(t+1) - \mathbf{1}_n\overline{g}(t+1)\|_{\mathrm{F}}.
\end{aligned}$$

Using Lemma 8, we can bound

$$\|g(t+1) - \mathbf{1}_n\overline{g}(t+1)\|_{\mathrm{F}} \le \|g(t+1)\|_{\mathrm{F}}.$$

Using Assumption 3, we can further bound

$$\begin{aligned}
\|g(t+1)\|_{\mathrm{F}}^2 &\le \sum_{k=1}^{n} \|D_k x_k(t+1) - D_k x_k(t)\|_2^2 \\
&\le \sum_{k=1}^{n} L_D^2 \|x_k(t+1) - x_k(t)\|_2^2 \\
&\le L_D^2 \|x(t+1) - x(t)\|_{\mathrm{F}}^2.
\end{aligned}$$

Since $\mathbf{1}_n\overline{x}(t)$ is preserved by W (each column of $\mathbf{1}_n\overline{x}(t)$ is in $\mathbb{R}\mathbf{1}_n$), it follows that $(W-I)\mathbf{1}_n\overline{x}(t) = 0$, and we may expand

$$\begin{aligned}
x(t+1) - x(t) &= W\xi(t+1) - x(t) \\
&= W(\xi(t+1) - x(t)) + (W-I)(x(t) - \mathbf{1}_n\overline{x}(t)).
\end{aligned}$$

Since $\|W\|_2 \le 1$ and $\|W - I\|_2 \le 2$, we have

$$\begin{aligned}
\|x(t+1) - x(t)\|_{\mathrm{F}} &\le \|\xi(t+1) - x(t)\|_{\mathrm{F}} + 2\|x(t) \\
&\quad - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.
\end{aligned}$$

Thus far, this means that

$$\begin{aligned}
&\|y(t+1) - \mathbf{1}_n\overline{y}(t+1)\|_{\mathrm{F}} \le \sigma\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&+\sigma L_D\left(\|\xi(t+1) - x(t)\|_{\mathrm{F}} + 2\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}\right).
\end{aligned}$$

Recall that $\xi(t+1) = \widehat{F}(x(t), y(t))$, and $\widehat{\Phi}(x(t)) = \widehat{F}(x(t), \widehat{D}x(t))$. Expand

$$\begin{aligned}
\|\xi(t+1) - x(t)\|_{\mathrm{F}} &\le \|\widehat{\Phi}(x(t)) - x(t)\|_{\mathrm{F}} \\
&\quad + \|\widehat{F}(x(t), y(t)) - \widehat{\Phi}(x(t))\|_{\mathrm{F}}.
\end{aligned}$$

Lemma 7 gives us the following inequality for the second term:

$$\begin{aligned}
\|\widehat{F}(x(t), y(t)) - \widehat{\Phi}(x(t))\|_{\mathrm{F}} &\le L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&\quad + 2\Lambda\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.
\end{aligned}$$

Recall that $\|\Phi(x_k(t)) - x_k(t)\|_2 \le \delta\|x_k(t) - x^*\|_2$. It therefore follows that:

$$\begin{aligned}
\|\widehat{\Phi}(x(t)) - x(t)\|_{\mathrm{F}}^2 &= \sum_{k=1}^{n} \|\Phi(x_k(t)) - x_k(t)\|_2^2 \\
&\le \delta^2 \sum_{k=1}^{n} \|x_k(t) - x^*\|_2^2 \\
&= \delta^2\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}^2.
\end{aligned}$$

Finally, after taking square roots, we have obtained the bound

$$\begin{aligned}
\|\xi(t+1) - x(t)\|_{\mathrm{F}} &\le \delta\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}} \\
&\quad + L_y\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&\quad + 2\Lambda\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}}.
\end{aligned}$$

Putting it all together, we end up with

$$\begin{aligned}
&\|y(t+1) - \mathbf{1}_n\overline{y}(t+1)\|_{\mathrm{F},L_y} \\
&\quad \le \sigma(1+\Lambda)\|y(t) - \mathbf{1}_n\overline{y}(t)\|_{\mathrm{F}} \\
&\qquad + 2\sigma L_D(1+\Lambda)\|x(t) - \mathbf{1}_n\overline{x}(t)\|_{\mathrm{F}} \\
&\qquad + \sigma L_D\delta\|x(t) - \mathbf{1}_n x^*\|_{\mathrm{F}}
\end{aligned}$$

which is the inequality in the statement. □

VII. Conclusion

In this article, we developed a general framework for decentralizing a contractive iteration for multiple connected nodes in networks, which maintains the convergence rate of the original centralized iterative algorithm. Three examples, distributed proximal gradient descent, heavy ball method, and Newton's method, are given to demonstrate the proposed framework.

For future work, we will study the general decentralization method for the algorithms with a superlinear convergence rate, and for the ones with a sublinear convergence rate.

APPENDIX A

The following lemma is a general statement for the Frobenius norms of matrices related to the averaging operator. *Lemma 8:* For any matrix $x = (x_1^{\mathrm{T}}, \ldots, x_n^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{n\times d}$, i.e., for which row i is given by the vector $x_i^{\mathrm{T}} \in \mathbb{R}^d$, it holds that

$$\|x - \mathbf{1}_n \frac{1}{n}\sum_{i=1}^{n} x_i\|_{\mathrm{F}} \leq \|x\|_{\mathrm{F}}.$$

Proof: By the definition of the Frobenius norm, it holds that

$$\begin{aligned}
&\|x - \mathbf{1}_n \frac{1}{n}\sum_{i=1}^{n} x_i\|_{\mathrm{F}}^2 \\
&= \sum_{k=1}^{n} \left\langle x_k - \frac{1}{n}\sum_{i=1}^{n} x_i, x_k - \frac{1}{n}\sum_{i=1}^{n} x_i \right\rangle \\
&= \|x\|_{\mathrm{F}}^2 - 2\sum_{k=1}^{n} \left\langle x_k, \frac{1}{n}\sum_{i=1}^{n} x_i \right\rangle + \sum_{k=1}^{n} \left\langle \frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} x_i \right\rangle \\
&= \|x\|_{\mathrm{F}}^2 - 2n \left\langle \frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} x_i \right\rangle + n\langle \frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} x_i \rangle \\
&= \|x\|_{\mathrm{F}}^2 - n \left\langle \frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} x_i \right\rangle \\
&= \|x\|_{\mathrm{F}}^2 - \|\mathbf{1}_n \frac{1}{n}\sum_{i=1}^{n} x_i\|_{\mathrm{F}}^2 \leq \|x\|_{\mathrm{F}}^2.
\end{aligned}$$

Thus, the assertion follows. □

REFERENCES

[1] M. H. DeGroot, "Reaching a consensus," *J. Amer. Statist. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.

[2] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[3] R. A. Freeman, P. Yang, and K. M. Lynch, "Stability and convergence properties of dynamic average consensus estimators," in *Proc. IEEE 45th Proc. Conf. Decis. Control*, 2006, pp. 338–343.

[4] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., MIT, Cambridge, MA, USA, 1984.

[5] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, Sep. 1986.

[6] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

[7] J. B. Predd, S. B. Kulkarni, and H. V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 56–69, Jul. 2006.

[8] J. B. Predd, S. R. Kulkarni, and H. V. Poor, "A collaborative training algorithm for distributed learning," *IEEE Trans. Inf. Theory*, vol. 55, no. 4, pp. 1856–1871, Apr. 2009.

[9] R. O. Saber and R. M. Murray, "Agreement problems in networks with directed graphs and switching topology," in *Proc. IEEE 42nd Int. Conf. Decis. Control*, 2003, vol. 4, pp. 4126–4132.

[10] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," in *Proc. Amer. Control Conf.*, 2005, pp. 1859–1864.

[11] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE Proc. IRE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[12] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[13] S. Zhang, F. He, Y. Hong, and X. Hu, "An intrinsic approach to formation control of regular polyhedra for reduced attitudes," *Automatica*, vol. 111, 2020, Art. no. 108619.

[14] S. Zhang, W. Song, F. He, Y. Hong, and X. Hu, "Intrinsic tetrahedron formation of reduced attitude," *Automatica*, vol. 87, pp. 375–382, 2018.

[15] J. A. Costa, N. Patwari, and A. O. Hero III, "Distributed weighted-multidimensional scaling for node localization in sensor networks," *ACM Trans. Sensor Netw.*, vol. 2, no. 1, pp. 39–64, 2006.

[16] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, 2005, pp. 63–70.

[17] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proc. IEEE 44th Conf. Decis. Control*, 2005, pp. 8179–8184.

[18] Q. Ling and Z. Tian, "Decentralized sparse signal recovery for compressive sleeping wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3816–3827, Jul. 2010.

[19] D. Jakovetić, "A unification and generalization of exact distributed first-order methods," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 1, pp. 31–46, Mar. 2019.

[20] J. Xu, Y. Tian, Y. Sun, and G. Scutari, "Distributed algorithms for composite optimization: Unified framework and convergence analysis," *IEEE Trans. Signal Process.*, vol. 69, pp. 3555–3570, 2021.

[21] A. S. Berahas, R. Bollapragada, N. S. Keskar, and E. Wei, "Balancing communication and computation in distributed optimization," *IEEE Trans. Autom. Control*, vol. 64, no. 8, pp. 3141–3155, Aug. 2019.

[22] A. Mahmoudi, H. S. Ghadikolaei, and C. Fischione, "Cost-efficient distributed optimization in machine learning over wireless networks," in *Proc. IEEE Int. Conf. Commun.*, Dublin, Ireland, 2020, pp. 1–7, doi: 10.1109/ICC40277.2020.9149216.

[23] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Norwell, MA, U.K.: Kluwer, 2004.

[24] D. Bertsekas, *Abstract Dynamic Programming*, Nashua, NH, USA: Athena Scientific, 2018.

[25] T. Yang et al., "A survey of distributed optimization," *Annu. Rev. Control*, vol. 47, pp. 278–305, 2019.

[26] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, 1998.

[27] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for non-differentiable optimization," *SIAM J. Optim.*, vol. 12, no. 1, pp. 109–138, 2001.

[28] B. Johansson, "On distributed optimization in networked systems," Ph.D. dissertation, *Sch. Elect. Eng.*, KTH, Stockholm, Sweden, 2008.

[29] M. Gürbüzbalaban, A. Ozdaglar, and P. A. Parrilo, "On the convergence rate of incremental aggregated gradient algorithms," *SIAM J. Optim.*, vol. 27, no. 2, pp. 1035–1048, 2017.

[30] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[31] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM J. Optim.*, vol. 26, no. 3, pp. 1835–1854, 2016.

[32] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM J. Optim.*, vol. 25, no. 2, pp. 944–966, 2015.

[33] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes," in *Proc. IEEE 54th Conf. Decis. Control*, 2015, pp. 2055–2060.

[34] A. Nedić, A. Olshevsky, and A. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, 2017.

[35] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 3, pp. 1245–1260, Sep. 2018.

[36] G. Qu and N. Li, "Accelerated distributed nesterov gradient descent," *IEEE Trans. Autom. Control*, vol. 65, no. 6, pp. 2566–2581, Jun. 2020.

[37] R. Xin and U. A. Khan, "Distributed heavy-ball: A generalization and acceleration of first-order methods with gradient tracking," *IEEE Trans. Autom. Control*, vol. 65, no. 6, pp. 2627–2633, Jun. 2020.

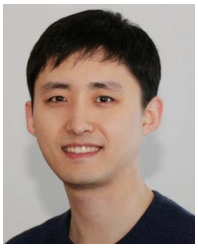
[38] D. Varagnolo, F. Zanella, A. Cenedese, G. Pillonetto, and L. Schenato, "Newton–Raphson consensus for distributed convex optimization," *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 994–1009, Apr. 2016.

[39] S. Alghunaim, K. Yuan, and A. Sayed, "A linearly convergent proximal gradient algorithm for decentralized optimization," in *Advances in Neural Information Processing Systems*, vol. 32, Red Hook, NY, USA: Curran Associates, Inc., 2019.

[40] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4494–4506, Sep. 2019.
[41] S. A. Alghunaim, E. K. Ryu, K. Yuan, and A. H. Sayed, "Decentralized proximal gradient algorithms with linear convergence rates," *IEEE Trans. Autom. Control*, vol. 66, no. 6, pp. 2787–2794, Jun. 2021.
[42] Y. Sun, M. Fazlyab, and S. Shahrampour, "On centralized and distributed mirror descent: Exponential convergence analysis using quadratic constraints," *IEEE Trans. Autom. Control*, vol. 68, no. 5, pp. 3139–3146, May 2023.
[43] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 592–606, Mar. 2012.
[44] E. Wei and A. Ozdaglar, "On the o(1/k) convergence of asynchronous distributed alternating direction method of multipliers," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2013, pp. 551–554.
[45] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Trans. Signal Process.*, vol. 62, no. 7, pp. 1750–1761, Apr. 2014.
[46] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "A Bregman splitting scheme for distributed optimization over networks," *IEEE Trans. Autom. Control*, vol. 63, no. 11, pp. 3809–3824, Nov. 2018.
[47] A. Sundararajan, B. Van Scoy, and L. Lessard, "Analysis and design of first-order distributed optimization algorithms over time-varying graphs," *IEEE Trans. Control Netw. Syst.*, vol. 7, no. 4, pp. 1597–1608, Dec. 2020.
[48] S. Han, "Systematic design of decentralized algorithms for consensus optimization," *IEEE Contr. Syst. Lett.*, vol. 3, no. 4, pp. 966–971, Oct. 2019.
[49] B. T. Polyak, "Introduction to optimization," in *Optimization Software*, vol. 1. New York, NY, USA: Publications Division, 1987.
[50] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[51] H. Federer, *General Measure Theory*. Berlin, Germany: Springer, 1996.
[52] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2012.

Thomas Ohlson Timoudas received the Ph.D. degree in mathematics from the Department of Mathematics, KTH Royal Institute of Technology, Stockholm, Sweden, in 2019.

He is currently a Researcher with RISE Research Institutes of Sweden, Stockholm, Sweden. His research interests include distributed systems, energy networks, machine learning, and artificial intelligence.



Silun Zhang received the B.Eng. and M.Sc. degrees in automation from the Harbin Institute of Technology, Harbin, China, and the Ph.D. degree in optimization and systems theory from the Department of Mathematics, KTH Royal Institute of Technology, Stockholm, Sweden, in 2019.

From 2019 to 2021, he the Wallenberg Postdoctoral Fellow. He is currently a Postdoctoral Associate with the Laboratory for Information and Decision Systems (LIDS), and the Department of Electrical Engineering and Computer Science (EECS), Massachusetts Institute of Technology, Cambridge, MA, USA.

His research interests include privacy in cyber-physical systems, large-scale systems modeling, control on manifolds, and distributional reinforcement learning.

Sindri Magnússon (Member, IEEE) received the B.Sc. degree in mathematics from the University of Iceland, Reykjavík, Iceland, in 2011, the masters degree in applied mathematics (optimization and systems theory), and the Ph.D. degree in electrical engineering from the KTH Royal Institute of Technology, Stockholm Sweden, in 2013 and 2017, respectively.

He is currently an Associate Professor with the Department of Computer and Systems Science, Stockholm University, Stockholm, Sweden. He was a Postdoctoral Researcher with Harvard University, Cambridge, MA, USA, from 2018 to 2019, where he was a Visiting Ph.D. student for nine months in 2015 and 2016. His research interests include large scale distributed/parallel optimization, machine learning, and control, both theory and applications.

Carlo Fischione (Member, IEEE) received the Ph.D. degree in electrical and information engineering (3/3 years) in 2005 and the Laurea degree in electronic engineering (Laurea, *summa cum laude*, 5/5 years) in 2001 from the University of L'Aquila, L'Aquila, Italy.

He is Full Professor with the Electrical Engineering and Computer Science, Division of Network and Systems Engineering, KTH Royal Institute of Technology, Stockholm, Sweden. He is the Director of the KTH Data Science Micro Degree Program, an advanced education program dedicated to Ericsson's researchers worldwide. He has held research positions with the Massachusetts Institute of Technology, Cambridge, MA, USA, (2015, Visiting Professor); Harvard University, Cambridge, MA, USA (2015, Associate); and University of California at Berkeley, Berkeley, CA, USA, (2004–2005, Visiting Scholar, and 2007–2008, Research Associate). He has coauthored over 200 publications, including a book, book chapters, international journals and conferences, and international patents. His research interests include applied optimization, wireless, sensor networks, Internet of things, and machine learning.

Dr. Fischione is the Chair of the IEEE Machine Learning for Communications Emerging Technology Initiative of the IEEE Communication Society. He received a number of awards, such as the "IEEE Communication Society S. O. Rice" award for the best IEEE Transactions on Communications paper of 2018. He is an Editor at Large of IEEE TRANSACTIONS ON COMMUNICATIONS (Machine Learning for Communication and Networking) and Editor of *IEEE Journal on Selected Areas on Communications - Series on Machine Learning for Communication and Networking*, and has been serving as an Associate Editor of *IFAC Automatica* (2014–2019). Meanwhile, he also has offered his advice as a consultant to numerous technology companies such as ABB Corporate Research, Berkeley Wireless Sensor Network Lab, Ericsson Research, Synopsys, and United Technology Research Center. He is cofounder and scientific Director of ELK. He is Ordinary Member of DASP (the Italian academy of history Deputazione Abruzzese di Storia Patria).