Xin *et al. EURASIP Journal on Advances in Signal Processing* (2019) 2019:1
https://doi.org/10.1186/s13634-018-0596-y

EURASIP Journal on Advances in Signal Processing

RESEARCH **Open Access**

FROST—Fast row-stochastic optimization with uncoordinated step-sizes

Ran Xin[1], Chenguang Xi[1,2] and Usman A. Khan[1*]

Abstract
In this paper, we discuss distributed optimization over directed graphs, where doubly stochastic weights cannot be constructed. Most of the existing algorithms overcome this issue by applying push-sum consensus, which utilizes column-stochastic weights. The formulation of column-stochastic weights requires each agent to know (at least) its out-degree, which may be impractical in, for example, broadcast-based communication protocols. In contrast, we describe FROST (Fast Row-stochastic-Optimization with uncoordinated STep-sizes), an optimization algorithm applicable to directed graphs that does not require the knowledge of out-degrees, the implementation of which is straightforward as each agent locally assigns weights to the incoming information and locally chooses a suitable step-size. We show that FROST converges linearly to the optimal solution for smooth and strongly convex functions given that the largest step-size is positive and sufficiently small.

Keywords: Distributed optimization, Multiagent systems, Directed graphs, Linear convergence

1 Introduction

In this paper, we study distributed optimization, where n agents are tasked to solve the following problem:

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}),$$

where each objective, $f_i : \mathbb{R}^p \rightarrow \mathbb{R}$, is private and known only to agent i. The goal of the agents is to find the global minimizer of the aggregate cost, $F(\mathbf{x})$, via local communication with their neighbors and without revealing their private objective functions. This formulation has recently received great attention due to its extensive applications in, for example, machine learning [1–6], control [7], cognitive networks, [8, 9], and source localization [10, 11].

Early work on this topic includes Distributed Gradient Descent (DGD) [12, 13], which is computationally simple but is slow due to a diminishing step-size. The convergence rates are $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$ for general convex functions and $\mathcal{O}\left(\frac{\log k}{k}\right)$ for strongly convex functions, where k is the number of iterations. With a constant step-size, DGD converges faster albeit to an inexact solution [14, 15]. Related work also includes methods based on the Lagrangian dual [16–19] to achieve faster convergence, at the expense of more computation. To achieve both fast convergence and computational simplicity, some fast distributed first-order methods have been proposed. A Nesterov-type approach [20] achieves $\mathcal{O}\left(\frac{\log k}{k^2}\right)$ for smooth convex functions with bounded gradient assumption. EXact firsT-ordeR Algorithm (EXTRA) [21] exploits the difference of two consecutive DGD iterates to achieve a linear convergence to the optimal solution. Exact diffusion [22, 23] applies an adapt-then-combine structure [24] to EXTRA and generalizes the symmetric doubly stochastic weights required in EXTRA to locally balanced row-stochastic weights over undirected graphs. Of significant relevance to this paper is a distributed gradient tracking technique built on dynamic consensus [25], which enables each agent to asymptotically learn the gradient of the global objective function. This technique was first proposed simultaneously in [26, 27]. Xu et al. and Qu and Li [26, 28] combine it with the DGD structure to achieve improved convergence for smooth and convex problems. Lorenzo and Scutari [27, 29], on the other hand, propose the NEXT framework for a more general class of non-convex problems.

*Correspondence: khan@ece.tufts.edu
[1]Electrical and Computer Engineering, Tufts University, Medford, MA, USA
Full list of author information is available at the end of the article

All of the aforementioned methods assume that the multi-agent network is undirected. In practice, it may not be possible to achieve undirected communication. It is of interest, thus, to develop optimization algorithms that are fast and are applicable to arbitrary directed graphs. The challenge here lies in the fact that doubly stochastic weights, standard in many distributed optimization algorithms, cannot be constructed over arbitrary directed graphs. In particular, the weight matrices in directed graphs can only be either row-stochastic or column-stochastic, but not both.

We now discuss related work on directed graphs. Early work based on DGD includes subgradient-push [30, 31] and Directed-Distributed Gradient Descent (D-DGD) [32, 33], with a sublinear convergence rate of $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$. Some recent work extends these methods to asynchronous networks [34–36]. To accelerate the convergence, DEXTRA [37] combines push-sum [38] and EXTRA [21] to achieve linear convergence given that the step-size lies in some non-trivial interval. This restriction on the step-size is later relaxed in ADD-OPT/Push-DIGing [39, 40], which linearly converge for a sufficiently small step-size. Of relevance is also [41], where distributed non-convex problems are considered with column-stochastic weights. More recent work [42, 43] proposes the $\mathcal{AB}$ and $\mathcal{AB}m$ algorithms, which employ both row- and column-stochastic weights to achieve (accelerated) linear convergence over arbitrary strongly connected graphs. Note that although the construction of doubly stochastic weights is avoided, all of the aforementioned methods require each agent to know its out-degree to formulate doubly or column-stochastic weights. This requirement may be impractical in situations where the agents use a broadcast-based communication protocol. In contrast, Refs. [44, 45] provide algorithms that only use row-stochastic weights. Row-stochastic weight design is simple and is further applicable to broadcast-based methods.

In this paper, we focus on optimization with row-stochastic weights following the recent work in [44, 45]. We propose a fast optimization algorithm, termed as FROST (Fast Row-stochastic Optimization with uncoordinated STep-sizes), which is applicable to both directed and undirected graphs with uncoordinated step-sizes among the agents. Distributed optimization (based on gradient tracking) with uncoordinated step-sizes has been previously studied in [26, 46, 47], over undirected graphs with doubly stochastic weights, and in [48], over directed graphs with column-stochastic weights. These works introduce a notion of heterogeneity among the step-sizes, defined respectively as the relative deviation of the step-sizes from their average in [26, 46] and as the ratio of the largest to the smallest step-size in [47, 48]. It is then shown that when the heterogeneity is small enough, i.e., the step-sizes are very close to each other, and when the largest step-size follows a bound as a function of the heterogeneity, the proposed algorithms linearly converge to the optimal solution. A challenge in this formulation is that choosing a sufficiently small, local step-size does not ensure small heterogeneity, while no step-size can be chosen to be zero. In contrast, a major contribution of this paper is that we establish linear convergence with uncoordinated step-sizes when the upper bound on the step-sizes is independent of any notion of heterogeneity. The implementation of FROST therefore is completely local, since each agent locally chooses a sufficiently small step-size, independent of other step-sizes, and locally assigns row-stochastic weights to the incoming information. In addition, our analysis shows that all step-sizes except one can be zero for the algorithm to work, which is a novel result in distributed optimization. We show that FROST converges linearly to the optimal solution for smooth and strongly convex functions.

Notation: We use lowercase bold letters to denote vectors and uppercase italic letters to denote matrices. The matrix, I_n, represents the $n \times n$ identity, whereas $\mathbf{1}_n$ ($\mathbf{0}_n$) is the n-dimensional uncoordinated vector of all 1's (0's). We further use $\mathbf{e}_i$ to denote an n-dimensional vector of all 0's except 1 at the ith location. For an arbitrary vector, $\mathbf{x}$, we denote its ith element by $[\mathbf{x}]_i$ and diag$\{\mathbf{x}\}$ is a diagonal matrix with $\mathbf{x}$ on its main diagonal. We denote by $X \otimes Y$ the Kronecker product of two matrices, X and Y. For a primitive, row-stochastic matrix, $\underline{A}$, we denote its left and right Perron eigenvectors by $\boldsymbol{\pi}_r$ and $\mathbf{1}_n$, respectively, such that $\boldsymbol{\pi}_r^\top \mathbf{1}_n = 1$; similarly, for a primitive, column-stochastic matrix, $\underline{B}$, we denote its left and right Perron eigenvectors by $\mathbf{1}_n$ and $\boldsymbol{\pi}_c$, respectively, such that $\mathbf{1}_n^\top \boldsymbol{\pi}_c = 1$ [49]. For a matrix, X, we denote $\rho(X)$ as its spectral radius and diag(X) as a diagonal matrix consisting of the corresponding diagonal elements of X. The notation $\|\cdot\|_2$ denotes the Euclidean norm of vectors and matrices, while $\|\cdot\|_F$ denotes the Frobenius norm of matrices. Depending on the argument, we denote $\|\cdot\|$ either as a particular matrix norm, the choice of which will be clear in Lemma 1, or a vector norm that is compatible with this matrix norm, i.e., $\|X\mathbf{x}\| \leq \|X\|\|\mathbf{x}\|$ for all matrices, X, and all vectors, $\mathbf{x}$ [49].

We now describe the rest of the paper. Section 2 states the problem and assumptions. Section 3 reviews related algorithms that use doubly stochastic or column-stochastic weights and shows the intuition behind the analysis of these types of algorithms. In Section 4, we provide the main algorithm, FROST, proposed in this paper. In Section 5, we develop the convergence properties of FROST. Simulation results are provided in Section 6, and Section 7 concludes the paper.

2 Problem formulation

Consider n agents communicating over a strongly connected network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \cdots, n\}$ is the set of agents and $\mathcal{E}$ is the set of edges, $(i,j), i,j \in \mathcal{V}$, such that agent j can send information to agent i, i.e., $j \rightarrow i$. Define $\mathcal{N}_i^{\text{in}}$ as the collection of in-neighbors, i.e., the set of agents that can send information to agent i. Similarly, $\mathcal{N}_i^{\text{out}}$ is the set of out-neighbors of agent i. Note that both $\mathcal{N}_i^{\text{in}}$ and $\mathcal{N}_i^{\text{out}}$ include agent i. The agents are tasked to solve the following problem:

$$\text{P1}: \quad \min_{\mathbf{x}} F(\mathbf{x}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}),$$

where $f_i : \mathbb{R}^p \rightarrow \mathbb{R}$ is a private cost function only known to agent i. We denote the optimal solution of P1 as $\mathbf{x}^*$. We will discuss different distributed algorithms related to this problem under the applicable set of assumptions, described below:

Assumption 1 *The graph, $\mathcal{G}$, is undirected and connected.*

Assumption 2 *The graph, $\mathcal{G}$, is directed and strongly connected.*

Assumption 3 *Each local objective, f_i, is convex with bounded subgradient.*

Assumption 4 *Each local objective, f_i, is smooth and strongly convex, i.e., $\forall i$ and $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$,*

i. There exists a positive constant l such that

$$\left\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\right\|_2 \leq l\|\mathbf{x} - \mathbf{y}\|_2.$$

ii. there exists a positive constant μ such that

$$f_i(\mathbf{y}) \geq f_i(\mathbf{x}) + \nabla f_i(\mathbf{x})^\top(\mathbf{y} - \mathbf{x}) + \frac{\mu}{2}\|\mathbf{x} - \mathbf{y}\|_2^2.$$

Clearly, the Lipschitz continuity and strong convexity constants for the global objective function, $F = \frac{1}{n}\sum_{i=1}^{n} f_i$, are l and μ, respectively.

Assumption 5 *Each agent in the network has and knows its unique identifier, e.g., $1, \cdots, n$.*

If this were not true, the agents may implement a finite-time distributed algorithm to assign such identifiers, e.g., with the help of task allocation algorithms, [50, 51], where the task at each agent is to pick a unique number from the set $\{1, \ldots, n\}$.

Assumption 6 *Each agent knows its out-degree in the network, i.e., the number of its out-neighbors.*

We note here that Assumptions 3 and 4 do not hold together; when applicable, the algorithms we discuss use either one of these assumptions but not both. We will discuss FROST, the algorithm proposed in this paper, under Assumptions 2, 4, 5.

3 Related work

In this section, we discuss related distributed first-order methods and provide an intuitive explanation for each one of them.

3.1 Algorithms using doubly stochastic weights

A well-known solution to distributed optimization over undirected graphs is Distributed Gradient Descent (DGD) [12, 13], which combines distributed averaging with a local gradient step. Each agent i maintains a local estimate, $\mathbf{x}_k^i$, of the optimal solution, $\mathbf{x}^*$, and implements the following iteration:

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^{n} w_{ij}\mathbf{x}_k^j - \alpha_k \nabla f_i\left(\mathbf{x}_k^i\right), \tag{1}$$

where $W = \{w_{ij}\}$ is doubly stochastic and respects the graph topology. The step-size α_k is diminishing such that $\sum_{k=0}^{\infty}\alpha_k = \infty$ and $\sum_{k=0}^{\infty}\alpha_k^2 < \infty$. Under the Assumptions 1, 3, and 6, DGD converges to $\mathbf{x}^*$ at the rate of $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$. The convergence rate is slow because of the diminishing step-size. If a constant step-size is used in DGD, i.e., $\alpha_k = \alpha$, it converges faster to an error ball, proportional to α, around $\mathbf{x}^*$ [14, 15]. This is because $\mathbf{x}^*$ is not a fixed-point of the above iteration when the step-size is a constant.

To accelerate the convergence, Refs. [26, 28] recently propose a distributed first-order method based on gradient tracking, which uses a constant step-size and replaces the local gradient, at each agent in DGD, with an asymptotic estimator of the global gradient[1]. The algorithm is updated as follows [26, 28]:

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^{n} w_{ij}\mathbf{x}_k^j - \alpha\mathbf{y}_k^i, \tag{2a}$$

$$\mathbf{y}_{k+1}^i = \sum_{j=1}^{n} w_{ij}\mathbf{y}_k^j + \nabla f_i\left(\mathbf{x}_{k+1}^i\right) - \nabla f_i\left(\mathbf{x}_k^i\right), \tag{2b}$$

initialized with $\mathbf{y}_0^i = \nabla f_i\left(\mathbf{x}_0^i\right)$ and an arbitrary $\mathbf{x}_0^i$ at each agent. The first equation is essentially a descent method, after mixing with neighboring information, where the descent direction is $\mathbf{y}_k^i$, instead of $\nabla f_i\left(\mathbf{x}_k^i\right)$ as was in Eq. (1). The second equation is a global gradient estimator when viewed as dynamic consensus [52], i.e., $\mathbf{y}_k^i$ asymptotically tracks the average of local gradients: $\frac{1}{n}\sum_{i=1}^{n}\nabla f_i\left(\mathbf{x}_k^i\right)$. It is shown in Refs. [28, 40, 46] that $\mathbf{x}_k^i$ converges linearly to $\mathbf{x}^*$ under Assumptions 1, 4, and 6, with a sufficiently small step-size, α. Note that these methods, Eqs. (1) and (2a)–(2b), are not applicable to directed graphs as they require doubly stochastic weights.

3.2 Algorithms using column-stochastic weights

We first consider the case when DGD in Eq. (1) is applied to a directed graph and the weight matrix is column-stochastic but not row-stochastic. It can be obtained that [32]:

$$\overline{\mathbf{x}}_{k+1} = \overline{\mathbf{x}}_k - \frac{\alpha_k}{n}\sum_{i=1}^{n}\nabla f_i\left(\mathbf{x}_k^i\right), \tag{3}$$

where $\overline{\mathbf{x}}_k = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_k^i$. From Eq. (3), it is clear that the average of the estimates, $\overline{\mathbf{x}}_k$, converges to $\mathbf{x}^*$, as Eq. (3) can be viewed as a centralized gradient method if each local estimate $\mathbf{x}_k^i$ converges to $\overline{\mathbf{x}}_k$. However, since the weight matrix is *not row-stochastic*, the estimates of agents will not reach an agreement [32]. This discussion motivates combining DGD with an algorithm, called push-sum, briefly discussed next, that enables agreement over directed graphs with column-stochastic weights.

3.2.1 Push-sum consensus

Push-sum [38, 53] is a technique to achieve average consensus over arbitrary digraphs. At time k, each agent maintains two state vectors, $\mathbf{x}_k^i$, $\mathbf{z}_k^i \in \mathbb{R}^p$, and an auxiliary scalar variable, v_k^i, initialized with $v_0^i = 1$. Push-sum performs the following iterations:

$$v_{k+1}^i = \sum_{j=1}^{n} b_{ij} v_k^j, \tag{4a}$$

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^{n} b_{ij}\mathbf{x}_k^j \tag{4b}$$

$$\mathbf{z}_{k+1}^i = \frac{\mathbf{x}_{k+1}^i}{v_{k+1}^i}, \tag{4c}$$

where $\underline{B} = \left\{b_{ij}\right\}$ is column-stochastic. Equation (4a) can be viewed as an independent algorithm to asymptotically learn the right Perron eigenvector of $\underline{B}$; recall that the right Perron eigenvector of $\underline{B}$ is not $\mathbf{1}_n$ because $\underline{B}$ is not row-stochastic and we denote it by $\boldsymbol{\pi}_c$. In fact, it can be verified that $\lim_{k\to\infty} v_i(k) = n[\boldsymbol{\pi}_c]_i$ and that $\lim_{k\to\infty}\mathbf{x}_i(k) = [\boldsymbol{\pi}_c]_i\sum_{i=1}^{n}\mathbf{x}_i(0)$. Therefore, the limit of $\mathbf{z}_i(k)$, as the ratio of $\mathbf{x}_i(k)$ over $v_i(k)$, is the average of the initial values:

$$\lim_{k\to\infty}\mathbf{z}_k^i = \lim_{k\to\infty}\frac{\mathbf{x}_k^i}{v_k^i} = \frac{[\boldsymbol{\pi}_c]_i\sum_{i=1}^{n}\mathbf{x}_i(0)}{n[\boldsymbol{\pi}_c]_i} = \frac{\sum_{i=1}^{n}\mathbf{x}_0^i}{n}.$$

In the next subsection, we present subgradient-push that applies push-sum to DGD, see [32, 33] for an alternate approach that does not require eigenvector estimation of Eq. (4a).

3.2.2 Subgradient-push

To solve Problem P1 over arbitrary directed graphs, Refs. [30, 31] develop subgradient-push with the following iterations:

$$v_{k+1}^i = \sum_{j=1}^{n} b_{ij} v_k^j, \tag{5a}$$

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^{n} b_{ij}\mathbf{x}_k^j - \alpha_k\nabla f_i\left(\mathbf{z}_k^i\right), \tag{5b}$$

$$\mathbf{z}_{k+1} = \frac{\mathbf{x}_{k+1}^i}{v_{k+1}^i}, \tag{5c}$$

initialized with $v_0^i = 1$ and an arbitrary $\mathbf{x}_0^i$ at each agent. The step-size, α_k, satisfies the same conditions as in DGD. To understand these iterations, note that Eqs. (5a)–(5c) are nearly the same as Eqs. (4a)–(4c), except that there is an additional gradient term in Eq. (5b), which drives the limit of $\mathbf{z}_k^i$ to $\mathbf{x}^*$. Under the Assumptions 2, 3, and 6, subgradient-push converges to $\mathbf{x}^*$ at the rate of $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$. For extensions of subgradient-push to asynchronous networks, see recent work [34–36]. We next describe an algorithm that significantly improves this convergence rate.

3.2.3 ADD-OPT/Push-DIGing

ADD-OPT [39], extended to time-varying graphs in Push-DIGing [40], is a fast algorithm over directed graphs, which converges at a linear rate to $\mathbf{x}^*$ under the Assumptions 2, 4, and 6, in contrast to the sublinear convergence of subgradient-push. The three vectors, $\mathbf{x}_k^i$, $\mathbf{z}_k^i$, and $\mathbf{y}_k^i$, and a scalar v_k^i maintained at each agent i, are updated as follows:

$$v_{k+1}^i = \sum_{j=1}^{n} b_{ij} v_k^j, \tag{6a}$$

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^{n} b_{ij}\mathbf{x}_k^j - \alpha\mathbf{y}_k^i, \tag{6b}$$

$$\mathbf{z}_{k+1}^i = \frac{\mathbf{x}_{k+1}^i}{v_{k+1}^i}, \tag{6c}$$

$$\mathbf{y}_{k+1}^i = \sum_{j=1}^{n} b_{ij}\mathbf{y}_k^i + \nabla f_i\left(\mathbf{z}_{k+1}^i\right) - \nabla f_i\left(\mathbf{z}_k^i\right), \tag{6d}$$

where each agent is initialized with $v_0^i = 1$, $\mathbf{y}_0^i = \nabla f_i\left(\mathbf{x}_0^i\right)$, and an arbitrary $\mathbf{x}_0^i$. We note here that ADD-OPT/Push-DIGing essentially applies push-sum to the algorithm in Eqs. (2a)–(2b), where the doubly stochastic weights therein are replaced by column-stochastic weights.

3.2.4 The $\mathcal{AB}$ algorithm

As we can see, subgradient-push and ADD-OPT/Push-DIGing, described before, have a nonlinear term that comes from the division by the eigenvector estimation. In contrast, the $\mathcal{AB}$ algorithm, introduced in [42] and extended to $\mathcal{AB}m$ with the addition of a heavy-ball momentum term in [43] and to time-varying graphs in [54], removes this nonlinearity and remains applicable to directed graphs by a simultaneous application of row- and column-stochastic weights[2]. Each agent i maintains two variables: $\mathbf{x}_k^i, \mathbf{y}_k^i \in \mathbb{R}^p$, where, as before, $\mathbf{x}_k^i$ is the estimate of $\mathbf{x}^*$ and $\mathbf{y}_k^i$ tracks the average gradient, $\frac{1}{n}\sum_{i=1}^n \nabla f_i\left(\mathbf{x}_k^i\right)$. The $\mathcal{AB}$ algorithm, initialized with $\mathbf{y}_0^i = \nabla f_i\left(\mathbf{x}_0^i\right)$ and arbitrary $\mathbf{x}_0^i$ at each agent, performs the following iterations:

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{x}_k^j - \alpha \mathbf{y}_k^i, \tag{7a}$$

$$\mathbf{y}_{k+1}^i = \sum_{j=1}^n b_{ij}\mathbf{y}_k^j + \nabla f_i\left(\mathbf{x}_{k+1}^i\right) - \nabla f_i\left(\mathbf{x}_k^i\right), \tag{7b}$$

where $\underline{A} = \{a_{ij}\}$ is row-stochastic and $\underline{B} = \{b_{ij}\}$ is column-stochastic. It is shown that $\mathcal{AB}$ converges linearly to $\mathbf{x}^*$ for sufficiently small step-sizes under the Assumptions 2, 4, and 6 [42]. Therefore, $\mathcal{AB}$ can be viewed as a generalization of the algorithm in Eqs. (2a)–(2b) as the doubly stochastic weights therein are replaced by row- and column-stochastic weights. Furthermore, it is shown in [43] that ADD-OPT/Push-DIGing in Eqs. (6a)–(6d) in fact can be derived from an equivalent form of $\mathcal{AB}$ after a state transformation on the $\mathbf{x}_k$-update; see [43] for details. For applications of the $\mathcal{AB}$ algorithm to distributed least squares, see, for instance, [56].

4 Algorithms using row-stochastic weights

All of the aforementioned methods require at least each agent to know its out-degree in the network in order to construct doubly or column-stochastic weights. This requirement may be infeasible, e.g., when agents use broadcast-based communication protocols. Row-stochastic weights, on the other hand, are easier to implement in a distributed manner as every agent locally assigns an appropriate weight to each incoming variable from its in-neighbors. In the next section, we describe the main contribution of this paper, i.e., a fast optimization algorithm that uses only row-stochastic weights and uncoordinated step-sizes.

To motivate the proposed algorithm, we first consider DGD in Eq. (1) over directed graphs when the weight matrix in DGD is chosen to be row-stochastic, but not column-stochastic. From consensus arguments and the fact that the step-size α_k goes to 0, it can be verified that the agents achieve agreement. However, this agreement is not on the optimal solution. This can be shown [32] by defining an accumulation state, $\widehat{\mathbf{x}}_k = \sum_{i=1}^n [\boldsymbol{\pi}_r]_i \mathbf{x}_k^i$, where $\boldsymbol{\pi}_r$ is the left Perron eigenvector of the row-stochastic weight matrix, to obtain

$$\widehat{\mathbf{x}}(k+1) = \widehat{\mathbf{x}}(k) - \alpha_k \sum_{i=1}^n [\boldsymbol{\pi}_r]_i \nabla f_i\left(\mathbf{x}_i(k)\right). \tag{8}$$

It can be verified that the agents agree to the limit of the above iteration, which is suboptimal since this iteration minimizes a weighted sum of the objective functions and not the sum. This argument leads to a modification of Eq. (8) that cancels the *imbalance* in the gradient term caused by the fact that $\boldsymbol{\pi}_r$ is not a vector of all 1's, a consequence of losing the column-stochasticity in the weight matrix. The modification, introduced in [44], is implemented as follows:

$$\mathbf{y}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{y}_k^j, \tag{9a}$$

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{x}_k^j - \alpha_k \frac{\nabla f_i\left(\mathbf{x}_k^i\right)}{\left[\mathbf{y}_k^i\right]_i}, \tag{9b}$$

where $\underline{A} = \{a_{ij}\}$ is row-stochastic and the algorithm is initialized with $\mathbf{y}_0^i = \mathbf{e}_i$ and an arbitrary $\mathbf{x}_0^i$ at each agent. Equation (9a) asymptotically learns the left Perron eigenvector of the row-stochastic weight matrix $\underline{A}$, i.e., $\lim_{k\to\infty} \mathbf{y}_k^i = \boldsymbol{\pi}_r, \forall i$. The above algorithm achieves a sublinear convergence rate of $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$ under the Assumptions 2, 3, and 5, see [44] for details.

4.1 FROST (Fast Row-stochastic Optimization with uncoordinated STep-sizes)

Based on the insights that gradient tracking and constant step-sizes provide exact and fast linear convergence, we now describe FROST that adds gradient tracking to the algorithm in Eqs. (9a)–(9b) while using constant but uncoordinated step-sizes at the agents. Each agent i at the kth iteration maintains three variables, $\mathbf{x}_k^i, \mathbf{z}_k^i \in \mathbb{R}^p$, and $\mathbf{y}_k^i \in \mathbb{R}^n$. At $k+1$-th iteration, agent i performs the following update:

$$\mathbf{y}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{y}_k^j, \tag{10a}$$

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{x}_k^j - \alpha_i \mathbf{z}_k^i, \tag{10b}$$

$$\mathbf{z}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{z}_k^j + \frac{\nabla f_i\left(\mathbf{x}_{k+1}^i\right)}{\left[\mathbf{y}_{k+1}^i\right]_i} - \frac{\nabla f_i\left(\mathbf{x}_k^i\right)}{\left[\mathbf{y}_k^i\right]_i}, \tag{10c}$$

where α_i's are the uncoordinated step-sizes locally chosen at each agent and the row-stochastic weights, $\underline{A} = \{a_{ij}\}$, respect the graph topology such that:

$$a_{ij} = \begin{cases} > 0, \; j \in \mathcal{N}_i^{\text{in}}, \\ 0, \; \text{otherwise}, \end{cases} \qquad \sum_{j=1}^{n} a_{ij} = 1, \; \forall i.$$

The algorithm is initialized with an arbitrary $\mathbf{x}_0^i$, $\mathbf{y}_0^i = \mathbf{e}_i$, and $\mathbf{z}_0^i = \nabla f_i\left(\mathbf{x}_0^i\right)$. We point out that the initial condition for Eq. (10a) and the divisions in Eq. (10c) require each agent to have a unique identifier. Clearly, Assumption 5 is applicable here. Note that Eq. (10c) is a modified gradient tracking update, first applied to optimization with row-stochastic weights in [45], where the divisions are used to eliminate the imbalance caused by the left Perron eigenvector of the (row-stochastic) weight matrix $\underline{A}$. We note that the algorithm in [45] requires identical step-sizes at the agents and thus is a special case of Eqs. (10a)–(10c).

For analysis purposes, we write Eqs. (10a)–(10c) in a compact vector-matrix form. To this aim, we introduce some notation as follows: let $\mathbf{x}_k$, $\mathbf{y}_k$, and $\nabla \mathbf{f}(\mathbf{x}_k)$ collect the local variables $\mathbf{x}_k^i$, $\mathbf{y}_k^i$, and $\nabla f_i\left(\mathbf{x}_k^i\right)$ in a vector in $\mathbb{R}^{np}$, respectively, and define

$$\begin{aligned} \underline{Y}_k &= \left[\mathbf{y}_k^1, \cdots, \mathbf{y}_k^n\right]^\top, \\ Y_k &= \underline{Y}_k \otimes I_p, \\ \widetilde{Y}_k &= \operatorname{diag}\left(Y_k\right), \\ A &= \underline{A} \otimes I_p, \\ \boldsymbol{\alpha} &= \left[\alpha_1, \cdots, \alpha_n\right]^\top, \\ D &= \operatorname{diag}\{\boldsymbol{\alpha}\} \otimes I_p. \end{aligned}$$

Since the weight matrix $\underline{A}$ is primitive with positive diagonals, it is straightforward to verify that $\widetilde{Y}_k$ is invertible for any k. Based on the notation above, Eqs. (10a)–(10c) can be written compactly as follows:

$$\begin{aligned} \underline{Y}_{k+1} &= \underline{A}\, \underline{Y}_k, && \text{(11a)} \\ \mathbf{x}_{k+1} &= A\mathbf{x}_k - D\mathbf{z}_k, && \text{(11b)} \\ \mathbf{z}_{k+1} &= A\mathbf{z}_k + \widetilde{Y}_{k+1}^{-1} \nabla \mathbf{f}\left(\mathbf{x}_{k+1}\right) - \widetilde{Y}_k^{-1} \nabla \mathbf{f}\left(\mathbf{x}_k\right), && \text{(11c)} \end{aligned}$$

where $\underline{Y}_0 = I_n$, $\mathbf{z}_0 = \nabla \mathbf{f}_0$, and $\mathbf{x}_0$ is arbitrary. We emphasize that the implementation of FROST needs no knowledge of agent's out-degree anywhere in the network in contrast to the earlier related work in [30–33, 37, 39, 40, 42, 43]. Note that Refs. [22, 23] also use row-stochastic weights but require an additional locally balanced assumption and are only applicable to undirected graphs.

5 Convergence analysis

In this section, we present the convergence analysis of FROST described in Eqs. (11a)–(11c). We first define a few additional variables as follows:

$$\begin{aligned} Y_\infty &= \lim_{k \to \infty} Y_k, \\ \widetilde{Y}_\infty &= \operatorname{diag}\left(Y_\infty\right), \\ \nabla \mathbf{f}(\mathbf{x}^*) &= \left[\nabla f_1(\mathbf{x}^*)^\top, \cdots, \nabla f_n(\mathbf{x}^*)^\top\right]^\top, \\ \tau &= \left\|A - I_{np}\right\|_2, \\ \epsilon &= \left\|I_{np} - Y_\infty\right\|_2, \\ \overline{\alpha} &= \max_i\{\alpha_i\}, \\ y &= \sup_k \left\|Y_k\right\|_2, \\ \widetilde{y} &= \sup_k \left\|\widetilde{Y}_k^{-1}\right\|_2. \end{aligned}$$

Since $\underline{A}$ is primitive and row-stochastic, from the Perron-Frobenius theorem [49], we note that $Y_\infty = \left(\mathbf{1}_n \boldsymbol{\pi}_r^\top\right) \otimes I_p$, where $\boldsymbol{\pi}_r^\top$ is the left Perron eigenvector of $\underline{A}$.

5.1 Auxiliary relations

We now start the convergence analysis with a key lemma regarding the contraction of the augmented weight matrix A under an arbitrary norm.

Lemma 1 *Let Assumption 2 hold and consider the augmented weight matrix $A = \underline{A} \otimes I_p$. There exists a vector norm, $\|\cdot\|$, such that $\forall \mathbf{a} \in \mathbb{R}^{np}$,*

$$\left\|A\mathbf{a} - Y_\infty \mathbf{a}\right\| \le \sigma \left\|\mathbf{a} - Y_\infty \mathbf{a}\right\|,$$

where $0 < \sigma < 1$ is some constant.

Proof It can be verified that $AY_\infty = Y_\infty$ and $Y_\infty Y_\infty = Y_\infty$, which leads to the following relation:

$$A\mathbf{a} - Y_\infty \mathbf{a} = \left(A - Y_\infty\right)\left(\mathbf{a} - Y_\infty \mathbf{a}\right).$$

Next, from the Perron-Frobenius theorem, we note that [49]

$$\rho\left(A - Y_\infty\right) = \rho\left(\underline{A} - \mathbf{1}_n \boldsymbol{\pi}_r^\top\right) < 1;$$

thus, there exists a matrix norm, $\|\cdot\|$, with $\|A - Y_\infty\| < 1$ and a compatible vector norm, $\|\cdot\|$, see Chapter 5 in [49], such that

$$\left\|A\mathbf{a} - Y_\infty \mathbf{a}\right\| \le \left\|A - Y_\infty\right\| \left\|\mathbf{a} - Y_\infty \mathbf{a}\right\|,$$

and the lemma follows with $\sigma = \|A - Y_\infty\|$. □

As shown above, the existence of a norm in which the consensus process with row-stochastic matrix $\underline{A}$ is a contraction does not follow the standard 2-norm argument for doubly stochastic matrices [28, 40]. The ensuing arguments built on this notion of contraction under arbitrary norms were first introduced in [39] for column-stochastic weights and in [45] for row-stochastic weights; these arguments are harmonized later to hold simultaneously for both row- and column-stochastic weights in [42, 43].

The next lemma, a direct consequence of the contraction introduced in Lemma 1, is a standard result from consensus and Markov chain theory [57].

Lemma 2 *Consider Y_k, generated from the weight matrix $\underline{A}$. We have:*

$$\|Y_k - Y_\infty\|_2 \le r\sigma^k, \qquad \forall k,$$

where r is some positive constant and σ is the contraction factor defined in Lemma 1.

Proof Note that $Y_k = \underline{A}^k \otimes I_p = A^k$ from Eq. (11a), and $Y_k - Y_\infty = A^k - Y_\infty = (A - Y_\infty)(A^{k-1} - Y_\infty) = (A - Y_\infty)^k$.

From Lemma 1, we have

$$\|Y_k - Y_\infty\| = \left\|(A - Y_\infty)^k\right\| \le \sigma^k.$$

The proof follows from the fact that all matrix norms are equivalent. □

As a consequence of Lemma 2, we next establish the linear convergence of the sequences $\left\{\widetilde{Y}_k^{-1}\right\}$ and $\left\{\widetilde{Y}_{k+1}^{-1} - \widetilde{Y}_k^{-1}\right\}$.

Lemma 3 *The following inequalities hold $\forall k$:*
(a) $\left\|\widetilde{Y}_k^{-1} - \widetilde{Y}_\infty^{-1}\right\|_2 \le \sqrt{n}r\widetilde{y}^2\sigma^k$;
(b) $\left\|\widetilde{Y}_{k+1}^{-1} - \widetilde{Y}_k^{-1}\right\|_2 \le 2\sqrt{n}r\widetilde{y}^2\sigma^k$.

Proof The proof of (a) is as follows:

$$\begin{aligned}\left\|\widetilde{Y}_k^{-1} - \widetilde{Y}_\infty^{-1}\right\|_2 &= \left\|\widetilde{Y}_k^{-1}(\widetilde{Y}_\infty - \widetilde{Y}_k)\widetilde{Y}_\infty^{-1}\right\|_2, \\ &\le \left\|\widetilde{Y}_k^{-1}\right\|_2 \left\|\widetilde{Y}_k - \widetilde{Y}_\infty\right\|_2 \left\|\widetilde{Y}_\infty^{-1}\right\|_2, \\ &\le \widetilde{y}^2 \left\|\mathrm{diag}\left(Y_k - Y_\infty\right)\right\|_2 \\ &\le \sqrt{n}r\widetilde{y}^2\sigma^k,\end{aligned}$$

where the last inequality uses Lemma 2 and the fact that $\|X\|_F \le \sqrt{n}\|X_2\|, \forall X \in \mathbb{R}^{n\times n}$. The result in (b) is straightforward by applying (a), i.e.,

$$\begin{aligned}\left\|\widetilde{Y}_{k+1}^{-1} - \widetilde{Y}_k^{-1}\right\|_2 &\le \left\|\widetilde{Y}_{k+1}^{-1} - \widetilde{Y}_\infty^{-1}\right\|_2 + \left\|\widetilde{Y}_\infty^{-1} - \widetilde{Y}_k^{-1}\right\|_2, \\ &\le \sqrt{n}r\widetilde{y}^2\sigma^{k+1} + \sqrt{n}r\widetilde{y}^2\sigma^k,\end{aligned}$$

which completes the proof. □

The next lemma presents the dynamics that govern the evolution of the weighted sum of $\mathbf{z}_k$; recall that $\mathbf{z}_k$, in Eq. (11c), asymptotically tracks the average of local gradients, $\frac{1}{n}\sum_{i=1}^n \nabla f_i\left(\mathbf{x}_k^i\right)$.

Lemma 4 *The following equation holds for all k:*

$$Y_\infty \mathbf{z}_k = Y_\infty \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k). \tag{12}$$

Proof Recall that $Y_\infty A = Y_\infty$. We obtain from Eq. (11c) that

$$Y_\infty \mathbf{z}_k = Y_\infty \mathbf{z}_{k-1} + Y_\infty \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - Y_\infty \widetilde{Y}_{k-1}^{-1}\nabla\mathbf{f}(\mathbf{x}_{k-1}).$$

Doing this iteratively, we have

$$Y_\infty \mathbf{z}_k = Y_\infty \mathbf{z}_0 + Y_\infty \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - Y_\infty \widetilde{Y}_0^{-1}\nabla\mathbf{f}(\mathbf{x}_0).$$

With the initial conditions that $\mathbf{z}_0 = \nabla\mathbf{f}(\mathbf{x}_0)$ and $\widetilde{Y}_0 = I_{np}$, we complete the proof. □

The next lemma, a standard result in convex optimization theory from [58], states that the distance to the optimal solution contracts in each step in the centralized gradient method.

Lemma 5 *Let μ and l be the strong convexity and Lipschitz continuity constants for the global objective function, $F(\mathbf{x})$, respectively. Then $\forall \mathbf{x} \in \mathbb{R}^p$ and $0 < \alpha < \frac{2}{l}$, we have*

$$\left\|\mathbf{x} - \alpha\nabla F(\mathbf{x}) - \mathbf{x}^*\right\|_2 \le \sigma_F \left\|\mathbf{x} - \mathbf{x}^*\right\|_2,$$

where $\sigma_F = \max\left(|1-\alpha\mu|, |1-\alpha l|\right)$.

With the help of the previous lemmas, we are ready to derive a crucial contraction relationship in the proposed algorithm.

5.2 Contraction relationship

Our strategy to show convergence is to bound $\|\mathbf{x}_{k+1} - Y_\infty\mathbf{x}_{k+1}\|$, $\|Y_\infty\mathbf{x}_{k+1} - \mathbf{1}_n \otimes \mathbf{x}^*\|_2$, and $\|\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_{k+1}\|$ as a linear function of their values in the last iteration and $\nabla\mathbf{f}(\mathbf{x}_k)$; this approach extends the work in [28] on doubly stochastic weights to row-stochastic weights. We will present this relationship in the next lemmas. Before we proceed, we note that since all vector norms are equivalent in $\mathbb{R}^{np}$, there exist positive constants c, d such that: $\|\cdot\|_2 \le c\|\cdot\|, \|\cdot\| \le d\|\cdot\|_2$. First, we derive a bound for $\|\mathbf{x}_{k+1} - Y_\infty\mathbf{x}_{k+1}\|$, the consensus error of the agents.

Lemma 6 *The following inequality holds, $\forall k$:*

$$\|\mathbf{x}_{k+1} - Y_\infty\mathbf{x}_{k+1}\| \le \sigma\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\| + \overline{\alpha}d\epsilon\|\mathbf{z}_k\|_2, \tag{13}$$

where d is the equivalence-norm constant such that $\|\cdot\| \le d\|\cdot\|_2$ and $\overline{\alpha}$ is the largest step-size among the agents.

Proof Note that $Y_\infty A = Y_\infty$. Using Eq. (11b) and Lemma 1, we have:

$$\begin{aligned}&\|\mathbf{x}_{k+1} - Y_\infty\mathbf{x}_{k+1}\| \\ &= \|A\mathbf{x}_k - D\mathbf{z}_k - Y_\infty\left(A\mathbf{x}_k - D\mathbf{z}_k\right)\| \le \sigma\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\| + \overline{\alpha}d\epsilon\|\mathbf{z}_k\|_2,\end{aligned}$$

which completes the proof. □

Next, we derive a bound for $\|Y_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|_2$, i.e., the optimality gap between the accumulation state of the network, $Y_\infty\mathbf{x}_{k+1}$, and the optimal solution, $\mathbf{1}_n\otimes\mathbf{x}^*$.

Lemma 7 *If $\boldsymbol{\pi}_r^\top\boldsymbol{\alpha} < \frac{2}{nl}$, the following inequality holds, $\forall k$:*

$$\begin{aligned}&\|Y_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|_2\\ &\le \overline{\alpha}nlc\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\| + \lambda\|Y_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|_2\\ &\quad+\overline{\alpha}yc\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\| + \overline{\alpha}\sqrt{n}ry\widetilde{y}^2\sigma^k\left\|\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2,\end{aligned}\tag{14}$$

where $\lambda = \max\left(\left|1 - n\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}\mu\right|, \left|1 - n\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}l\right|\right)$ and c is the equivalence-norm constant such that $\|\cdot\|_2 \le c\|\cdot\|$.

Proof Recalling that $Y_\infty = \left(\mathbf{1}_n\boldsymbol{\pi}_r^\top\right)\otimes I_p$ and $Y_\infty A = Y_\infty$, we have the following:

$$\begin{aligned}&\|Y_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|_2\\ &= \left\|Y_\infty\left(A\mathbf{x}_k - D\mathbf{z}_k + (D-D)Y_\infty\mathbf{z}_k\right) - \mathbf{1}_n\otimes\mathbf{x}^*\right\|_2,\\ &\le \left\|Y_\infty\mathbf{x}_k - Y_\infty DY_\infty\mathbf{z}_k - \mathbf{1}_n\otimes\mathbf{x}^*\right\|_2 + \overline{\alpha}yc\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\|.\end{aligned}\tag{15}$$

Since the last term in the inequality above matches the second last term in Eq. (14), we only need to handle the first term. We further note that:

$$Y_\infty DY_\infty = \left(\left(\mathbf{1}_n\boldsymbol{\pi}_r^\top\right)\otimes I_p\right)\left(\mathrm{diag}\{\boldsymbol{\alpha}\}\otimes I_p\right)\left(\left(\mathbf{1}_n\boldsymbol{\pi}_r^\top\right)\otimes I_p\right) = \left(\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}\right)Y_\infty.$$

Now, we derive a upper bound for the first term in Eq. (15)

$$\begin{aligned}&\|Y_\infty\mathbf{x}_k - Y_\infty DY_\infty\mathbf{z}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|_2\\ &\le\left\|(\mathbf{1}_n\otimes I_p)\left(\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\mathbf{x}_k - \mathbf{x}^* - n(\boldsymbol{\pi}_r^\top\boldsymbol{\alpha})\nabla F\left(\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\mathbf{x}_k\right)\right)\right\|_2\\ &\quad+\left\|n(\boldsymbol{\pi}_r^\top\boldsymbol{\alpha})(\mathbf{1}_n\otimes I_p)\nabla F\left(\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\mathbf{x}_k\right) - \left(\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}\right)Y_\infty\mathbf{z}_k\right\|_2,\\ &:= s_1 + s_2.\end{aligned}\tag{16}$$

If $\boldsymbol{\pi}_r^\top\boldsymbol{\alpha} < \frac{2}{nl}$, according to Lemma 5

$$s_1 \le \lambda\|Y_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|_2,\tag{17}$$

where $\lambda = \max\left(\left|1 - n\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}\mu\right|, \left|1 - n\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}l\right|\right)$.

Next we derive a bound for s_2

$$\begin{aligned}s_2 &= \left(\boldsymbol{\pi}_r^\top\boldsymbol{\alpha}\right)\left\|n(\mathbf{1}_n\otimes I_p)\nabla F\left(\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\mathbf{x}_k\right) - Y_\infty\mathbf{z}_k\right\|_2,\\ &\le\overline{\alpha}\left\|n(\mathbf{1}_n\otimes I_p)\nabla F\left(\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\mathbf{x}_k\right) - (\mathbf{1}_n\otimes I_p)\left(\mathbf{1}_n^\top\otimes I_p\right)\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2,\\ &\quad+\overline{\alpha}\left\|(\mathbf{1}_n\otimes I_p)\left(\mathbf{1}_n^\top\otimes I_p\right)\nabla\mathbf{f}(\mathbf{x}_k) - Y_\infty\mathbf{z}_k\right\|_2\\ &:= s_3 + s_4,\end{aligned}\tag{18}$$

where it is straightforward to bound s_3 as

$$s_3 \le \overline{\alpha}nlc\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\|.\tag{19}$$

Since $Y_\infty\widetilde{Y}_\infty^{-1} = \left(\mathbf{1}_n\mathbf{1}_n^\top\right)\otimes I_p$ and $Y_\infty\mathbf{z}_k = Y_\infty\widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)$ from Lemma 4, we have:

$$\begin{aligned}s_4 &= \overline{\alpha}\left\|Y_\infty\widetilde{Y}_\infty^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - Y_\infty\widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2\\ &\le\overline{\alpha}\sqrt{n}ry\widetilde{y}^2\sigma^k\left\|\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2,\end{aligned}\tag{20}$$

where we use Lemma 3. Combining Eqs. (15)–(20), we finish the proof. □

Next, we bound $\|\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_{k+1}\|$, the error in gradient estimation.

Lemma 8 *The following inequality holds, $\forall k$*

$$\begin{aligned}&\|\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_{k+1}\|\\ &\le\epsilon\widetilde{y}l\tau cd\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\| + \sigma\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\| + \overline{\alpha}\epsilon\widetilde{y}ld\|\mathbf{z}_k\|_2\\ &\quad+2d\sqrt{n}r\epsilon\widetilde{y}^2\sigma^k\|\nabla\mathbf{f}(\mathbf{x}_k)\|_2.\end{aligned}$$

Proof According to Eq. (11c) and Lemma 1, we have:

$$\begin{aligned}&\|\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_{k+1}\|\\ &\le\sigma\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\| + \left\|\left(\widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right) - \left(Y_\infty\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_k\right)\right\|.\end{aligned}\tag{21}$$

Note that $Y_\infty\mathbf{z}_k = Y_\infty\widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)$ from Lemma 4. Therefore,

$$\begin{aligned}&\left\|\left(\widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right) - \left(Y_\infty\mathbf{z}_{k+1} - Y_\infty\mathbf{z}_k\right)\right\|_2\\ &=\left\|\left(I_{np} - Y_\infty\right)\left(\widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right)\right\|_2,\\ &\le\epsilon\left\|\widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - \widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2 + \epsilon\left\|\widetilde{Y}_{k+1}^{-1}\nabla\mathbf{f}(\mathbf{x}_k) - \widetilde{Y}_k^{-1}\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2,\\ &\le\epsilon\widetilde{y}l\left\|\mathbf{x}_{k+1} - \mathbf{x}_k\right\|_2 + 2\sqrt{n}r\epsilon\widetilde{y}^2\sigma^k\left\|\nabla\mathbf{f}(\mathbf{x}_k)\right\|_2,\end{aligned}\tag{22}$$

where in the last inequality, we use Lemma 3. We now bound $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2$.

$$\begin{aligned}\left\|\mathbf{x}_{k+1} - \mathbf{x}_k\right\|_2 &\le\left\|(A - I_{np})\mathbf{x}_k\right\|_2 + \overline{\alpha}\left\|\mathbf{z}_k\right\|_2,\\ &\le\left\|(A - I_{np})\left(\mathbf{x}_k - Y_\infty\mathbf{x}_k\right)\right\|_2 + \overline{\alpha}\left\|\mathbf{z}_k\right\|_2,\\ &\le\tau\left\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\right\|_2 + \overline{\alpha}\left\|\mathbf{z}_k\right\|_2,\end{aligned}\tag{23}$$

where in the second inequality, we use the fact that $(A - I_{np})Y_\infty$ is a zero matrix. Combining Eqs. (21)–(23), we obtain the desired result. □

The last step is to bound $\|\mathbf{z}_k\|_2$ in terms of $\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\|$, $\|Y_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|_2$, and $\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\|$. Then, we can replace $\|\mathbf{z}_k\|_2$ in Lemmas 6 and 8 by this bound in order to develop a LTI system inequality.

Lemma 9 *The following inequality holds, $\forall k$:*

$$\begin{aligned}\|\mathbf{z}_k\|_2 \le{}& cnl\|\mathbf{x}_k - Y_\infty\mathbf{x}_k\| + nl\|Y_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|_2\\ &+c\|\mathbf{z}_k - Y_\infty\mathbf{z}_k\| + \sqrt{n}ry\widetilde{y}^2\sigma^k\|\nabla\mathbf{f}(\mathbf{x}_k)\|_2.\end{aligned}\tag{24}$$

Proof Recall that $Y_\infty \widetilde{Y}_\infty^{-1} = (\mathbf{1}_n \otimes I_p)\left(\mathbf{1}_n^\top \otimes I_p\right)$ and $Y_\infty \mathbf{z}_k = Y_\infty \widetilde{Y}_k^{-1} \nabla \mathbf{f}(\mathbf{x}_k)$ from Lemma 4. We have the following:

$$\begin{aligned}
\|\mathbf{z}_k\|_2 &\le \|\mathbf{z}_k - Y_\infty \mathbf{z}_k\|_2 + \|Y_\infty \mathbf{z}_k\|_2 \\
&\le c\|\mathbf{z}_k - Y_\infty \mathbf{z}_k\| + \|Y_\infty \widetilde{Y}_k^{-1}\nabla \mathbf{f}(\mathbf{x}_k) - Y_\infty \widetilde{Y}_\infty^{-1}\nabla \mathbf{f}(\mathbf{x}_k)\|_2 \\
&\quad + \|Y_\infty \widetilde{Y}_\infty^{-1}\nabla \mathbf{f}(\mathbf{x}_k) - (\mathbf{1}_n \otimes I_p)\left(\mathbf{1}_n^\top \otimes I_p\right)\nabla \mathbf{f}(\mathbf{x}^*)\|_2, \\
&\le c\|\mathbf{z}_k - Y_\infty \mathbf{z}_k\| + \sqrt{n}ly\tilde{y}^2\sigma^k\|\nabla \mathbf{f}(\mathbf{x}_k)\|_2 \\
&\quad + nl\|\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\|_2, \\
&\le cnl\|\mathbf{x}_k - Y_\infty \mathbf{x}_k\| + nl\|Y_\infty \mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\|_2 \\
&\quad + c\|\mathbf{z}_k - Y_\infty \mathbf{z}_k\| + \sqrt{n}ry\tilde{y}^2\sigma^k\|\nabla \mathbf{f}(\mathbf{x}_k)\|_2,
\end{aligned} \tag{25}$$

where in the second inequality, we use the fact that $\left(\mathbf{1}_n^\top \otimes I_p\right)\nabla \mathbf{f}(\mathbf{x}^*) = 0$, which is the optimality condition for Problem P1. □

Before the main result, we present an additional lemma from nonnegative matrix theory that will be helpful in establishing the linear convergence of FROST.

Lemma 10 *(Theorem 8.1.29 in [49]) Let $X \in \mathbb{R}^{n\times n}$ be a nonnegative matrix and $\mathbf{x} \in \mathbb{R}^n$ be a positive vector. If $X\mathbf{x} < \omega\mathbf{x}$, then $\rho(X) < \omega$.*

5.3 Main results

With the help of the auxiliary relationships developed in the previous subsection, we now present the main results as follows in Theorems 1 and 2. Theorem 1 states that the relationships derived in the previous subsection indeed provide a contraction when the largest step-size, $\overline{\alpha}$, is sufficiently small. Theorem 2 then establishes the linear convergence of FROST.

Theorem 1 *If $\boldsymbol{\pi}_r^\top \boldsymbol{\alpha} < \frac{2}{nl}$, the following LTI system inequality holds:*

$$\mathbf{t}_{k+1} \le J_{\boldsymbol{\alpha}} \mathbf{t}_k + H_k \mathbf{s}_k, \ \forall k, \tag{26}$$

where $\mathbf{t}_k, \mathbf{s}_k \in \mathbb{R}^3$, and $J_{\boldsymbol{\alpha}}, H_k \in \mathbb{R}^{3\times 3}$ are defined as follows:

$$\mathbf{t}_k = \begin{bmatrix} \|\mathbf{x}_k - Y_\infty \mathbf{x}_k\| \\ \|Y_\infty \mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\|_2 \\ \|\mathbf{z}_k - Y_\infty \mathbf{z}_k\| \end{bmatrix},$$

$$J_{\boldsymbol{\alpha}} = \begin{bmatrix} \sigma + a_1\overline{\alpha} & a_2\overline{\alpha} & a_3\overline{\alpha} \\ a_4\overline{\alpha} & \lambda & a_5\overline{\alpha} \\ a_6 + a_7\overline{\alpha} & a_8\overline{\alpha} & \sigma + a_9\overline{\alpha} \end{bmatrix},$$

$$H_k = \begin{bmatrix} \overline{\alpha} d\epsilon\sqrt{n}ry\tilde{y}^2 & 0 & 0 \\ \overline{\alpha}\sqrt{n}ry\tilde{y}^2 & 0 & 0 \\ d\sqrt{n}r\epsilon\tilde{y}^2\left(2 + \overline{\alpha}ry\tilde{y}\right) & 0 & 0 \end{bmatrix}\sigma^k,$$

$$\mathbf{s}_k = \begin{bmatrix} \|\nabla \mathbf{f}(\mathbf{x}_k)\|_2 \\ 0 \\ 0 \end{bmatrix},$$

and the constants a_i's are

$$\begin{aligned}
&a_1 = cd\epsilon nl, && a_4 = cnl && a_7 = cdnl^2\epsilon\tilde{y} \\
&a_2 = d\epsilon nl, && a_5 = yc, && a_8 = dnl^2\epsilon\tilde{y} \\
&a_3 = d^2\epsilon, && a_6 = \epsilon\tilde{y}l\tau cd, && a_9 = d^2\epsilon l\tilde{y}.
\end{aligned}$$

Let $[\boldsymbol{\pi}_r]_-$ be the smallest element in $\boldsymbol{\pi}_r$. When the largest step-size, $\overline{\alpha}$, satisfies

$$0 < \overline{\alpha} < \min\left\{ \frac{\delta_1(1-\sigma)}{a_1\delta_1 + a_2\delta_2 + a_3\delta_3}, \frac{(1-\sigma)\delta_3 - \delta_1 a_6}{a_7\delta_1 + a_8\delta_2 + a_9\delta_3}, \frac{1}{nl} \right\}, \tag{27}$$

with positive constants $\delta_1, \delta_2, \delta_3$ such that

$$\delta_3 > 0, \qquad \delta_1 < \frac{(1-\sigma)\delta_3}{a_6}, \qquad \delta_2 > \frac{a_4\delta_1 + a_5\delta_3}{\mu n[\boldsymbol{\pi}_r]_-}, \tag{28}$$

then the spectral radius of $J_{\boldsymbol{\alpha}}$ is strictly less than 1.

Proof Combining Lemmas 6–9, one can verify that Eq. (26) holds if $\boldsymbol{\pi}_r^\top \boldsymbol{\alpha} < \frac{2}{nl}$. Recall that $\lambda = \max\left(\left|1 - \mu n\boldsymbol{\pi}_r^\top \boldsymbol{\alpha}\right|, \left|1 - ln\boldsymbol{\pi}_r^\top \boldsymbol{\alpha}\right|\right)$. When $\boldsymbol{\pi}_r^\top \boldsymbol{\alpha} < \frac{1}{nl}$, $\lambda = 1 - \mu n\boldsymbol{\pi}_r^\top \boldsymbol{\alpha}$, since $\mu \le l$ [59]. In order to make $\boldsymbol{\pi}_r^\top \boldsymbol{\alpha} < \frac{1}{nl}$ hold, it is suffice to require $\overline{\alpha} < \frac{1}{nl}$. The next step is to find an upper bound, $\hat{\alpha}$, on the largest step-size such that $\rho(J_{\boldsymbol{\alpha}}) < 1$ when $\overline{\alpha} < \hat{\alpha}$. In the light of Lemma 10, we solve for the range of the largest step-size, $\overline{\alpha}$, and a positive vector $\boldsymbol{\delta} = [\delta_1, \delta_2, \delta_3]^\top$ from the following:

$$\begin{bmatrix} \sigma + a_1\overline{\alpha} & a_2\overline{\alpha} & a_3\overline{\alpha} \\ a_4\overline{\alpha} & 1 - \mu n(\boldsymbol{\pi}_r^\top \boldsymbol{\alpha}) & a_5\overline{\alpha} \\ a_6 + a_7\overline{\alpha} & a_8\overline{\alpha} & \sigma + a_9\overline{\alpha} \end{bmatrix} \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \end{bmatrix} < \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \end{bmatrix}, \tag{29}$$

which is equivalent to the following set of inequalities:

$$\begin{cases} (a_1\delta_1 + a_2\delta_2 + a_3\delta_3)\overline{\alpha} < \delta_1(1-\sigma), \\ (a_4\delta_1 + a_5\delta_3)\overline{\alpha} - \delta_2\mu n\boldsymbol{\pi}_r^\top \boldsymbol{\alpha} < 0, \\ (a_7\delta_1 + a_8\delta_2 + a_9\delta_3)\overline{\alpha} < (1-\sigma)\delta_3 - \delta_1 a_6. \end{cases} \tag{30}$$

Since the right hand side of the third inequality in Eq. (30) has to be positive, we have that:

$$0 < \delta_1 < \frac{(1-\sigma)\delta_3}{a_6}. \tag{31}$$

In order to find the range of δ_2 such that the second inequality holds, it suffices to solve for the range of δ_2 such that the following inequality holds:

$$(a_4\delta_1 + a_5\delta_3)\overline{\alpha} - \delta_2\mu n[\boldsymbol{\pi}_r]_-\overline{\alpha} < 0,$$

where [_r]$_-$ is the smallest entry in $\boldsymbol{\pi}_r$. Therefore, as long as

$$\delta_2 > \frac{a_4\delta_1 + a_5\delta_3}{\mu n[\boldsymbol{\pi}_r]_-}, \tag{32}$$

the second inequality in Eq. (30) holds. The next step is to solve the range of $\overline{\alpha}$ from the first and third inequalities in Eq. (30). We get

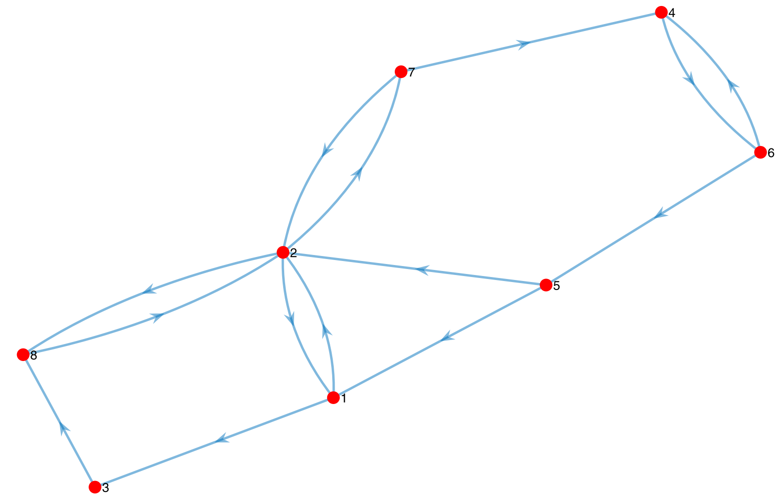

Fig. 1 A strongly connected and unbalanced directed graph

$$\overline{\alpha} < \min\left\{\frac{\delta_1(1-\sigma)}{a_1\delta_1+a_2\delta_2+a_3\delta_3}, \frac{(1-\sigma)\delta_3-\delta_1 a_6}{a_7\delta_1+a_8\delta_2+a_9\delta_3}\right\},$$

where the range of δ_1 and δ_2 is given in Eqs. (31) and (32), respectively, and δ_3 is an arbitrary positive constant and the theorem follows. □

Note that $\delta_1, \delta_2, \delta_3$ are essentially adjustable parameters that are chosen independently from the step-sizes. Specifically, according to Eq. (28), we first choose an arbitrary positive constant δ_3 and subsequently choose a constant δ_1 such that $0 < \delta_1 < \frac{(1-\sigma)\delta_3}{a_6}$ and finally we choose a constant δ_2 such that $\delta_2 > \frac{a_4\delta_1+a_5\delta_3}{\mu n[\boldsymbol{\pi}_r]_-}$.

Theorem 2 *If the largest step-size* $\overline{\alpha}$ *follows the bound in Eq.* (27)*, we have:*

$$\left\|\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\right\| \le m\left(\max\left\{\rho\left(J_{\boldsymbol{\alpha}}\right), \sigma\right\} + \xi\right)^k,$$

where ξ *is an arbitrarily small constant,* σ *is the contraction factor defined in Lemma* 1*, and* m *is some positive constant.*

Noticing that $\rho\left(J_{\boldsymbol{\alpha}}\right) < 1$ when the largest step-size, $\overline{\alpha}$, follows the bound in Eq. (27) and that H_k linearly decays at the rate of σ^k, one can intuitively verify Theorem 2. A rigorous proof follows from [45].

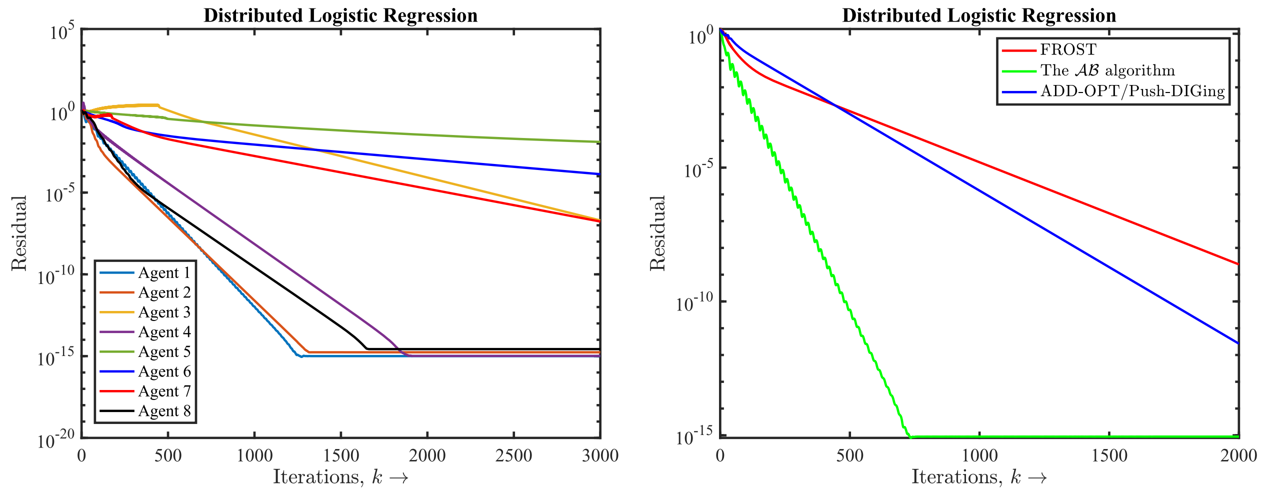

Fig. 2 (Left) Each curve represents the linear convergence of FROST when the corresponding agent uses a positive step-size, optimized manually, while every other agent uses zero step-size. (Right) Convergence comparison across different algorithms

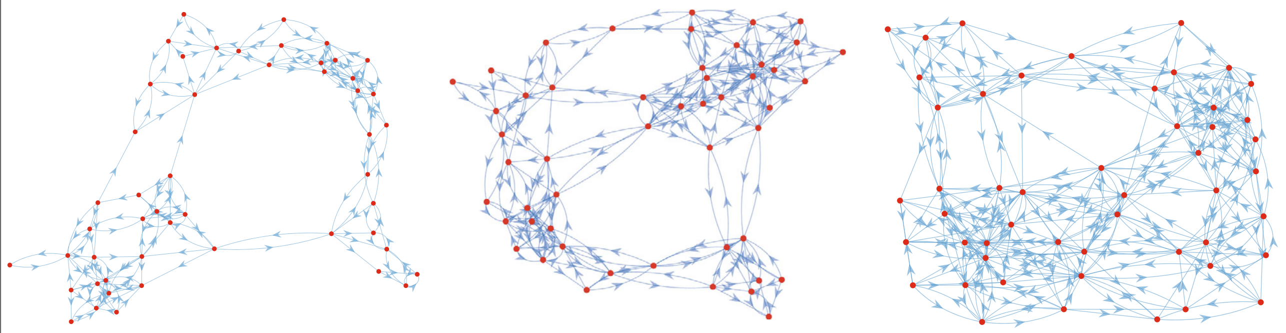
Fig. 3 Directed graphs with $n = 50$ nodes and increasing sparsity: $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$

In Theorems 1 and 2, we establish the linear convergence of FROST when the largest step-size, $\overline{\alpha}$, follows the upper bound defined in Eq. (27). Distributed optimization (based on gradient tracking) with uncoordinated step-sizes have been previously studied in [26, 46, 47], over undirected graphs with doubly stochastic weights, and in [48], over directed graphs with column-stochastic weights. These works rely on some notion of heterogeneity of the step-sizes, defined respectively as the relative deviation of the step-sizes from their average, $\frac{\|(I_n-U)\boldsymbol{\alpha}\|_2}{\|U\boldsymbol{\alpha}\|_2}$, where $U = \mathbf{1}_n\mathbf{1}_n^\top/n$, in [26, 46], and as the ratio of the largest to the smallest step-size, $\frac{\max_i\{\alpha_i\}}{\min_i\{\alpha_i\}}$, in [47, 48]. The authors then show that when the heterogeneity is small enough and when the largest step-size follows a bound that is a function of the heterogeneity, the proposed algorithms converge to the optimal solution. It is worth noting that sufficiently small step-sizes cannot guarantee sufficiently small heterogeneity in both of the aforementioned definitions. In contrast, the upper bound on the largest step-size in this paper, Eqs. (27) and (28), is independent of any notion of heterogeneity and only depends on the objective functions and the network parameters[3]. Each agent therefore locally picks a sufficiently small step-size independent of other step-sizes. Besides, this bound allows the agents to choose a zero step-size as long as at least one of them is positive and sufficiently small.

6 Numerical results

In this section, we use numerical experiments to support the theoretical results. We consider a distributed logistic regression problem. Each agent i has access to m_i training data, $(\mathbf{c}_{ij}, y_{ij}) \in \mathbb{R}^p \times \{-1,+1\}$, where $\mathbf{c}_{ij}$ contains p features of the jth training data at agent i and y_{ij} is the corresponding binary label. The network of agents cooperatively solves the following distributed logistic regression problem:

$$\min_{\mathbf{w}\in\mathbb{R}^p, b\in\mathbb{R}} F(\mathbf{w},b) = \sum_{i=1}^{n}\sum_{j=1}^{m_i} \ln\left[1+\exp\left(-\left(\mathbf{w}^\top\mathbf{c}_{ij}+b\right)y_{ij}\right)\right] + \frac{n\lambda}{2}\|\mathbf{w}\|_2^2,$$

with each private loss function being

$$f_i(\mathbf{w},b) = \sum_{j=1}^{m_i} \ln\left[1+\exp\left(-\left(\mathbf{w}^\top\mathbf{c}_{ij}+b\right)y_{ij}\right)\right] + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \tag{33}$$

where $\frac{\lambda}{2}\|\mathbf{w}\|_2^2$ is a regularization term used to prevent overfitting of the data. The feature vectors, $\mathbf{c}_{ij}$'s, are randomly generated from some Gaussian distribution with zero mean. The binary labels are randomly generated from some Bernoulli distribution. The network topology is shown in Fig. 1. We adopt a simple uniform weighting strategy to construct the row- and column-stochastic weights when needed: $a_{ij} = 1/|\mathcal{N}_i^{\text{in}}|$, $b_{ij} = 1/|\mathcal{N}_j^{\text{out}}|$, $\forall i,j$. We plot the average of residuals at each agent, $\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i(k) - \mathbf{x}^*\|_2$. In Fig. 2 (left), each curve represents the linear convergence of FROST when the corresponding agent uses a positive step-size, optimized manually, while every other agent uses zero step-size.

In Fig. 2 (right), we compare the performance of FROST, with ADD-OPT/Push-DIGing [39, 40], see Section 3.2.3, and with the $\mathcal{AB}$ algorithm in [42, 43], see Section 3.2.4. The step-size used in each algorithm is optimized. For FROST, we first manually find the optimal identical step-

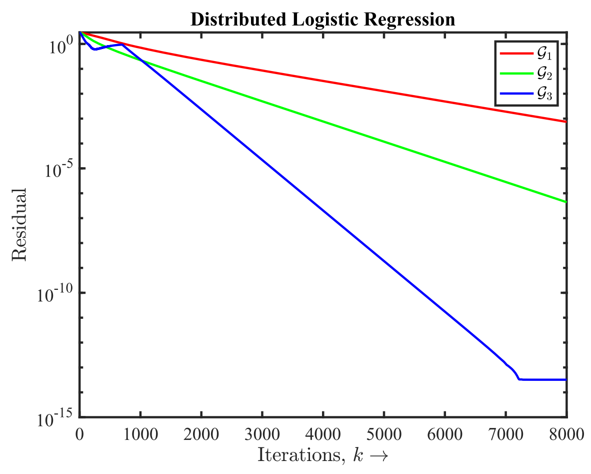

Fig. 4 Influence of network sparsity on the performance of FROST

size for all agents, which is 0.07 in our experiment, and then randomly generate uncoordinated step-sizes of FROST from the uniform distribution over the interval $[0, 0.07]$ (therefore, the convergence speed of FROST shown in this experiment is conservative). The numerical experiments thus verify our theoretical finding that as long as the largest step-size of FROST is positive and sufficiently small, FROST linearly converges to the optimal solution.

In the next experiment, we show the influence of the network sparsity on the convergence of FROST. For this purpose, we use three different graphs each with $n = 50$ nodes, where $\mathcal{G}_1$ has roughly 10% of total edges, $\mathcal{G}_2$ has roughly 13% of total edges, and $\mathcal{G}_3$ has roughly 16% of total edges. These graphs are shown in Fig. 3, and the performance of FROST over each one of them is shown in Fig. 4.

7 Conclusions

In this paper, we consider distributed optimization applicable to both directed and undirected graphs with row-stochastic weights and when the agents in the network have uncoordinated step-sizes. Most of the existing algorithms are based on column-stochastic weights, which may be infeasible to implement in many practical scenarios. Row-stochastic weights, on the other hand, are straightforward to implement as each agent locally determines the weights assigned to each incoming information. We propose a fast algorithm that we call FROST (Fast Row-stochastic Optimization with uncoordinated STep-sizes) and show that when the largest step-size is positive and sufficiently small, FROST linearly converges to the optimal solution. Simulation results further verify the theoretical analysis.

Endnotes

[1] EXTRA [21] is another related algorithm, which uses the difference between two consecutive DGD iterates to achieve linear convergence to the optimal solution.

[2] See [32, 33] for related work with sublinear rate based on surplus consensus [55].

[3] The constants δ_1, δ_2, and δ_3 in Eqs. (27) and (28) are tunable parameters that only depend on the network topology and objective functions.

Abbreviations

ADD-OPT: (Accelerated Distributed Directed OPTimization); DGD: (Distributed Gradient Descent); EXTRA: (EXact firsT-ordeR Algorithm); FROST: (Fast Row-stochastic-Optimization with uncoordinated STep-sizes)

Acknowledgments

Not applicable.

Funding

This work has been partially supported by an NSF Career Award # CCF-1350264. None of the authors have any competing interests in the manuscript.

Availability of data and materials

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Authors' contributions

All authors contributed equally to this paper. All authors read and approved the final manuscript.

Authors' information

Ran Xin received his B.S. degree in Mathematics and Applied Mathematics from Xiamen University, China, in 2016. and M.S. in Electrical Engineering from Tufts University, USA, in 2018. Currently, he is a Ph.D. student in the Electrical and Computer Engineering department at Tufts University. His research interests include optimization, control, and statistical learning theory. Chenguang Xi received his B.S. degree in Microelectronics from Shanghai JiaoTong University, China, in 2010, M.S. and Ph.D. degrees in Electrical and Computer Engineering from Tufts University, in 2012 and 2016, respectively. Currently, he is a research scientist in Applied Machine Learning at Facebook Inc. His research interests include machine learning and artificial intelligence. Dr. Usman Khan has been an Associate Professor of Electrical and Computer Engineering (ECE) at Tufts University, Medford, MA, USA, since September 2017, where he is the Director of *Signal Processing and Robotic Networks* laboratory. His research interests include statistical signal processing, network science, and distributed optimization over autonomous multi-agent systems. He has published extensively in these topics with more than 75 articles in journals and conference proceedings and holds multiple patents. Recognition of his work includes the prestigious National Science Foundation (NSF) Career award, several NSF REU awards, an IEEE journal cover, three best student paper awards in IEEE conferences, and several news articles including one on IEEE spectrum. Dr. Khan joined Tufts as an Assistant Professor in 2011 and held a Visiting Professor position at KTH, Sweden, in Spring 2015. Prior to joining Tufts, he was a postdoc in the GRASP lab at the University of Pennsylvania. He received his B.S. degree in 2002 from University of Engineering and Technology, Pakistan, M.S. degree in 2004 from University of Wisconsin-Madison, USA, and Ph.D. degree in 2009 from Carnegie Mellon University, USA, all in ECE. Dr. Khan is an IEEE senior member and has been an associate member of the Sensor Array and Multichannel Technical Committee with the IEEE Signal Processing Society since 2010. He is an elected member of the IEEE Big Data special interest group and has served on the IEEE Young Professionals Committee and on IEEE Technical Activities Board. He was an editor of the IEEE Transactions on Smart Grid from 2014 to 2017, and is currently an associate editor of the IEEE Control System Letters. He has served on the Technical Program Committees of several IEEE conferences and has organized and chaired several IEEE workshops and sessions.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details

[1]Electrical and Computer Engineering, Tufts University, Medford, MA, USA.
[2]Facebook Inc., Menlo Park, CA, USA.

Received: 5 April 2018 Accepted: 7 December 2018
Published online: 14 January 2019

References

1. P. A. Forero, A. Cano, G. B. Giannakis, Consensus-based distributed support vector machines. J. Mach. Learn. Res. **11**(May), 1663–1707 (2010)
2. S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2011). https://doi.org/10.1561/2200000016
3. H. Raja, W.U. Bajwa, Cloud K-SVD: A collaborative dictionary learning algorithm for big, distributed data. IEEE Trans. Signal Process. **64**(1), 173–188 (2016)

4. H.-T. Wai, Z. Yang, Z. Wang, M. Hong, Multi-agent reinforcement learning via double averaging primal-dual optimization. arXiv preprint arXiv:1806.00877 (2018)
5. P. Di Lorenzo, AH. Sayed, Sparse distributed learning based on diffusion adaptation. IEEE Trans. Signal rocess. **61**(6), 1419–1433 (2013)
6. S. Scardapane, R. Fierimonte, P. Di Lorenzo, M. Panella, A. Uncini, Distributed semi-supervised support vector machines. Neural Netw. **80**, 43–52 (2016)
7. A. Jadbabaie, J. Lin, A. Morse, Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans. Autom. Control. **48**(6), 988–1001 (2003)
8. G. Mateos, J. A. Bazerque, G. B. Giannakis, Distributed sparse linear regression. IEEE Trans. Signal Process. **58**(10), 5262–5276 (2010). https://doi.org/10.1109/TSP.2010.2055862
9. J. A. Bazerque, G. B. Giannakis, Distributed spectrum sensing for cognitive radio networks by exploiting sparsity. IEEE Trans. Signal Process. **58**(3), 1847–1862 (2010). https://doi.org/10.1109/TSP.2009.2038417
10. M. Rabbat, R. Nowak, in *3rd International Symposium on Information Processing in Sensor Networks*. Distributed optimization in sensor networks (IEEE, 2004), pp. 20–27. https://doi.org/10.1109/IPSN.2004.1307319
11. S. Safavi, U. A. Khan, S. Kar, J. M. F. Moura, Distributed localization: a linear theory. Proc. IEEE. **106**(7), 1204–1223 (2018). https://doi.org/10.1109/JPROC.2018.2823638
12. J. Tsitsiklis, D. P. Bertsekas, M. Athans, Distributed asynchronous deterministic and stochastic gradient optimization algorithms. IEEE Trans. Autom. Control. **31**(9), 803–812 (1986)
13. A. Nedić, A. Ozdaglar, Distributed subgradient methods for multi-agent optimization. IEEE Trans. Autom. Control. **54**(1), 48–61 (2009). https://doi.org/10.1109/TAC.2008.2009515
14. K. Yuan, Q. Ling, W. Yin, On the convergence of decentralized gradient descent. SIAM J. Optim. **26**(3), 1835–1854 (2016)
15. A. S. Berahas, R. Bollapragada, N. S. Keskar, E. Wei, Balancing communication and computation in distributed optimization. arXiv preprint arXiv:1709.02999 (2017)
16. H. Terelius, U. Topcu, R. M. Murray, Decentralized multi-agent optimization via dual decomposition. IFAC Proc. Vol. **44**(1), 11245–11251 (2011)
17. J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, M. Puschel, D-ADMM: a communication-efficient distributed algorithm for separable optimization. IEEE Trans. Signal Process. **61**(10), 2718–2723 (2013). https://doi.org/10.1109/TSP.2013.2254478
18. E. Wei, A. Ozdaglar, in *51st IEEE Annual Conference on Decision and Control*. Distributed alternating direction method of multipliers (IEEE, 2012), pp. 5445–5450. https://doi.org/10.1109/CDC.2012.6425904
19. W. Shi, Q. Ling, K. Yuan, G. Wu, W. Yin, On the linear convergence of the ADMM in decentralized consensus optimization. IEEE Trans. Signal Process. **62**(7), 1750–1761 (2014). https://doi.org/10.1109/TSP.2014.2304432
20. D. Jakovetic, J. Xavier, J. M. F. Moura, Fast distributed gradient methods. IEEE Trans. Autom. Control. **59**(5), 1131–1146 (2014)
21. W. Shi, Q. Ling, G. Wu, W. Yin, Extra: An exact first-order algorithm for decentralized consensus optimization. SIAM J. Optim. **25**(2), 944–966 (2015). https://doi.org/10.1137/14096668X. http://dx.doi.org/10.1137/14096668X
22. K. Yuan, B. Ying, X. Zhao, A. H. Sayed, Exact diffusion for distributed optimization and learning - part I: algorithm development. IEEE Trans. Signal Process, 1 (2018). https://doi.org/10.1109/TSP.2018.2875898
23. K. Yuan, B. Ying, X. Zhao, A. H. Sayed, Exact diffusion for distributed optimization and learning - part II: convergence analysis. IEEE Trans. Signal Process. https://doi.org/10.1109/TSP.2018.2875883 (2018)
24. A. H. Sayed, in *Academic Press Library in Signal Processing vol. 3*. Diffusion adaptation over networks (Elsevier, Amsterdam, 2014), pp. 323–453
25. M. Zhu, S. Martínez, Discrete-time dynamic average consensus. Automatica. **46**(2), 322–329 (2010)
26. J. Xu, S. Zhu, Y. C. Soh, L. Xie, in *IEEE 54th Annual Conference on Decision and Control*. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes (IEEE, 2015), pp. 2055–2060
27. P. D. Lorenzo, G. Scutari, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing*. Distributed nonconvex optimization over time-varying networks (IEEE, 2016), pp. 4124–4128
28. G. Qu, N. Li, Harnessing smoothness to accelerate distributed optimization. IEEE Trans. Control Netw. Syst. **5**(3), 1245–1260 (2018). https://doi.org/10.1109/TCNS.2017.2698261
29. P. Di Lorenzo, G. Scutari, Next: In-network nonconvex optimization. IEEE Trans. Signal Inf. Process. Over Netw. **2**(2), 120–136 (2016)
30. K. I. Tsianos, S. Lawlor, M. G. Rabbat, in *51st IEEE Annual Conference on Decision and Control*. Push-sum distributed dual averaging for convex optimization (IEEE, 2012), pp. 5453–5458. https://doi.org/10.1109/CDC.2012.6426375
31. A. Nedić, A. Olshevsky, Distributed optimization over time-varying directed graphs. IEEE Trans. Autom. Control. **60**(3), 601–615 (2015)
32. C. Xi, Q. Wu, U. A. Khan, On the distributed optimization over directed networks. Neurocomputing. **267**, 508–515 (2017)
33. C. Xi, U. A. Khan, Distributed subgradient projection algorithm over directed graphs. IEEE Trans. Autom. Control. **62**(8), 3986–3992 (2016)
34. M. Assran, M. Rabbat, Asynchronous subgradient-push. arXiv preprint arXiv:1803.08950 (2018)
35. J. Zhang, K. You, Asyspa: An exact asynchronous algorithm for convex optimization over digraphs. arXiv preprint arXiv:1808.04118 (2018)
36. A. Olshevsky, I. C. Paschalidis, A. Spiridonoff, Robust asynchronous stochastic gradient-push: asymptotically optimal and network-independent performance for strongly convex functions. arXiv preprint arXiv:1811.03982 (2018)
37. C. Xi, U. A. Khan, DEXTRA: a fast algorithm for optimization over directed graphs. IEEE Trans. Autom. Control. **62**(10), 4980–4993 (2017)
38. D. Kempe, A. Dobra, J. Gehrke, in *44th Annual IEEE Symposium on Foundations of Computer Science*. Gossip-based computation of aggregate information (IEEE, 2003), pp. 482–491. https://doi.org/10.1109/SFCS.2003.1238221
39. C. Xi, R. Xin, U. A. Khan, ADD-OPT: accelerated distributed directed optimization. IEEE Trans. Autom. Control. **63**(5), 1329–1339 (2017)
40. A. Nedić, A. Olshevsky, W. Shi, Achieving geometric convergence for distributed optimization over time-varying graphs. SIAM J. Optim. **27**(4), 2597–2633 (2017)
41. Y. Sun, G. Scutari, D. Palomar, in *2016 50th Asilomar Conference on Signals, Systems and Computers*. Distributed nonconvex multiagent optimization over time-varying networks (IEEE, Pacific Grove, 2016), pp. 788–794
42. R. Xin, U. A. Khan, A linear algorithm for optimization over directed graphs with geometric convergence. IEEE Control Syst. Lett. **2**(3), 325–330 (2018)
43. R. Xin, U. A. Khan, Distributed heavy-ball: a generalization and acceleration of first-order methods with gradient tracking. arXiv preprint arXiv:1808.02942 (2018)
44. V. S. Mai, E. H. Abed, in *2016 American Control Conference (ACC)*. Distributed optimization over weighted directed graphs using row stochastic matrix (IEEE, 2016), pp. 7165–7170. https://doi.org/10.1109/ACC.2016.7526803
45. C. Xi, V. S. Mai, R. Xin, E. Abed, U. A. Khan, Linear convergence in optimization over directed graphs with row-stochastic matrices. IEEE Trans. Autom. Control. (2018). in press
46. J. Xu, S. Zhu, Y. Soh, L. Xie, Convergence of asynchronous distributed gradient methods over stochastic networks. IEEE Trans. Autom. Control. **63**(2), 434–448 (2018)
47. A. Nedić, A. Olshevsky, W. Shi, C. A. Uribe, in *2017 American Control Conference (ACC)*. Geometrically convergent distributed optimization with uncoordinated step-sizes (IEEE, 2017), pp. 3950–3955. https://doi.org/10.23919/ACC.2017.7963560
48. Q. Lü, H. Li, D. Xia, Geometrical convergence rate for distributed optimization with time-varying directed graphs and uncoordinated step-sizes. Inf. Sci. **422**, 516–530 (2018)
49. R. A. Horn, C. R. Johnson, *Matrix Analysis, 2nd ed.* (Cambridge University Press, New York, NY, 2013)
50. H. W. Kuhn, The Hungarian method for the assignment problem. Nav. Res. Logist. Q. **2**(1–2), 83–97 (1955)
51. S. Safavi, U. A. Khan, in *48th IEEE Asilomar Conference on Signals, Systems, and Computers*. On the convergence rate of swap-collide algorithm for simple task assignment (IEEE, 2014), pp. 1507–1510
52. M. Zhu, S. Martínez, Discrete-time dynamic average consensus. Automatica. **46**(2), 322–329 (2010)
53. F. Benezit, V. Blondel, P. Thiran, J. Tsitsiklis, M. Vetterli, in *IEEE International Symposium on Information Theory*. Weighted gossip: distributed averaging using non-doubly stochastic matrices (IEEE, 2010), pp. 1753–1757. https://doi.org/10.1109/ISIT.2010.5513273

54. F. Saadatniaki, R. Xin, U.A. Khan, Optimization over time-varying directed graphs with row and column-stochastic matrices. arXiv preprint arXiv:1810.07393 (2018)
55. K. Cai, H. Ishii, Average consensus on general strongly connected digraphs. Automatica. **48**(11), 2750–2761 (2012). https://doi.org/10.1016/j.automatica.2012.08.003
56. T. Yang, J. George, J. Qin, X. Yi, J. Wu, Distributed finite-time least squares solver for network linear equations. arXiv preprint arXiv:1810.00156 (2018)
57. R. A. Horn, C. R. Johnson, *Matrix Analysis*. (Cambridge University Press, New York, NY, 2013)
58. D. P. Bertsekas, *Nonlinear Programming*. (Athena scientific Belmont, Belmont, 1999)
59. Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course vol. 87*. (Springer, New York, 2013)