Distributed heavy-ball: A generalization and acceleration of first-order methods with gradient tracking

Ran Xin, *Student Member, IEEE*, and Usman A. Khan, *Senior Member, IEEE*

***Abstract*—We study distributed optimization to minimize a global objective that is a sum of smooth and strongly-convex local cost functions. Recently, several algorithms over undirected and directed graphs have been proposed that use a gradient tracking method to achieve linear convergence to the global minimizer. However, a connection between these different approaches has been unclear. In this paper, we first show that many of the existing first-order algorithms are in fact related with a simple state transformation, at the heart of which lies the $\mathcal{AB}$ algorithm. We then describe *distributed heavy-ball*, denoted as $\mathcal{AB}m$, i.e., $\mathcal{AB}$ with momentum, that combines gradient tracking with a momentum term and uses nonidentical local step-sizes. By simultaneously implementing both row- and column-stochastic weights, $\mathcal{AB}m$ removes the conservatism in the related work due to doubly-stochastic weights or eigenvector estimation. $\mathcal{AB}m$ thus naturally leads to optimization and average-consensus over both undirected and directed graphs, casting a unifying framework over several well-known consensus algorithms over arbitrary strongly-connected graphs. We show that $\mathcal{AB}m$ has a global R-linear rate when the largest step-size is positive and sufficiently small. Following the standard practice in the heavy-ball literature, we numerically show that $\mathcal{AB}m$ achieves accelerated convergence especially when the objective function is ill-conditioned.**

***Index Terms*—Distributed optimization, linear convergence, first-order method, heavy ball method, momentum.**

I. Introduction

We consider distributed optimization, where n agents collaboratively solve the following problem:

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

and each local objective, $f_i : \mathbb{R}^p \to \mathbb{R}$, is smooth and strongly-convex. The goal of the agents is to find the global minimizer of the aggregate cost via only local communication with their neighbors. This formulation has recently received great interest with applications in e.g., machine learning [1–4], control [5], cognitive networks, [6, 7], and source localization [8, 9].

Early work on this topic builds on the seminal work by Tsitsiklis in [10] and includes Distributed Gradient Descent (DGD) [11] and distributed dual averaging [12] over undirected graphs. Leveraging push-sum consensus [13], Refs. [14, 15] extend the DGD framework to directed graphs. Based on a similar concept, Refs. [16, 17] propose Directed-Distributed Gradient Descent (D-DGD) for directed graphs that is based on surplus consensus [18]. In general, the DGD-based methods achieve sublinear convergence at $\mathcal{O}\left(\frac{\log k}{\sqrt{k}}\right)$, where k is the number of iterations, because of the diminishing step-size used in the iterations. The convergence rate of DGD can be improved with the help of a constant step-size but at the expense of an inexact solution [19, 20]. Follow-up work also includes augmented Lagrangians [21–24], which shows exact linear convergence for smooth and strongly-convex functions, albeit requiring higher computation at each iteration.

To improve convergence and retain computational simplicity, fast first-order methods that do not (explicitly) use a dual update have been proposed. Reference [25] describes a distributed Nesterov-type method based on multiple consensus inner loops, at $\mathcal{O}\left(\frac{\log k}{k^2}\right)$ for smooth and convex functions, with bounded gradients. EXTRA [26] uses the difference of two consecutive DGD iterates to achieve an $\mathcal{O}\left(\frac{1}{k}\right)$ rate for arbitrary convex functions and a Q-linear rate for strongly-convex functions. DEXTRA [27] combines push-sum [13] and EXTRA [26] to achieve an R-linear rate over directed graphs given that a constant step-size is carefully chosen in some interval. Refs. [28, 29] apply an adapt-then-combine structure [30] to EXTRA [26] and generalize the symmetric weights in EXTRA to row-stochastic, over undirected graphs.

Noting that DGD-type methods are faster with a constant step-size, recent work [31–40] uses a constant step-size and replaces the local gradient, at each agent in DGD, with an estimate of the global gradient. A method based on gradient tracking was first shown in [31] over undirected graphs, which proposes Aug-DGM (that uses nonidentical step-sizes at the agents) with the help of dynamic consensus [41] and shows convergence for smooth convex functions. When the step-sizes are identical, the convergence rate of Aug-DGM was derived to be $\mathcal{O}\left(\frac{1}{k}\right)$ for arbitrary convex functions and R-linear for strongly-convex functions in [32]. ADD-OPT [33] extends [32] to directed graphs by combining push-sum with gradient tracking and derives a contraction in an arbitrary norm to establish an R-linear convergence rate when the global objective is smooth and strongly-convex. Ref. [34] extends the analysis in [32, 33] to time-varying graphs and establishes an R-linear convergence using the small gain theorem [42]. In contrast to the aforementioned methods [31–34], where the weights are doubly-stochastic for undirected graphs and column-stochastic for directed graphs, FROST [35, 36] uses row-stochastic weights, which have certain advantages over column-stochastic weights. Ref. [39] unifies EXTRA [26] and gradient tracking methods [31, 32] in a primal-dual framework over static undirected graphs. More recently, Ref. [38] proposes distributed Nesterov over undirected graphs that also uses gradient tracking and shows a convergence rate of $\mathcal{O}((1-c\mathcal{Q}^{-\frac{5}{7}})^k)$ for smooth, strongly-convex functions, where $\mathcal{Q}$ is the condition number of the global objective. Refs. [43, 44], on the other hand, consider gradient tracking in distributed non-convex problems, while Ref. [40] uses second-order information to accelerate the convergence.

The authors are with the ECE Department at Tufts University, Medford, MA; ran.xin@tufts.edu, khan@ece.tufts.edu. This work has been partially supported by an NSF Career Award # CCF-1350264.

Of significant relevance here is the $\mathcal{AB}$ algorithm [37], also appeared later in [45], which can be viewed as a generalization of distributed first-order methods with gradient tracking. In particular, the algorithms over undirected graphs in Refs. [31, 32] are a special case of $\mathcal{AB}$ because the doubly-stochastic weights therein are replaced by row- and column- stochastic weights. $\mathcal{AB}$ thus is naturally applicable to arbitrary directed graphs. Moreover, the use of both row- and column-stochastic weights removes the need for eigenvector estimation[1], required earlier in [33–36]. Ref. [37] derives an R-linear rate for $\mathcal{AB}$ when the objective functions are smooth and strongly-convex. In this paper, we provide an improved understanding of $\mathcal{AB}$ and extend it to the $\mathcal{AB}m$ algorithm, a *distributed heavy-ball method*, applicable to both undirected and directed graphs. We now summarize the main contributions:

1) We show that many of the existing accelerated first-order methods are either a special case of $\mathcal{AB}$ [31, 32], or can be adapted from its equivalent forms [33–36].
2) We propose a distributed heavy-ball method, termed as $\mathcal{AB}m$, that combines $\mathcal{AB}$ with a heavy-ball (type) momentum term. To the best of our knowledge, this paper is the first to use a momentum term based on the heavy-ball method in distributed optimization.
3) $\mathcal{AB}m$ employs nonidentical step-sizes at the agents and thus its analysis naturally carries to nonidentical step-sizes in $\mathcal{AB}$ and to the related algorithms in [31–36].
4) We cast a unifying framework for consensus over arbitrary graphs that results from $\mathcal{AB}m$ and subsumes several well-known algorithms [18, 46].

On the analysis front, we show that $\mathcal{AB}$ (without momentum) converges faster as compared to the algorithms over directed graphs in [33–36], where separate iterations for eigenvector estimation are applied nonlinearly to the underlying algorithm. Towards $\mathcal{AB}m$, we establish a *global* R-linear convergence rate for smooth and strongly-convex objective functions when the largest step-size at the agents is positive and sufficiently small. This is in contrast to the earlier work on non-identical step-sizes within the framework of gradient tracking [31, 47–49], which requires the heterogeneity among the step-sizes to be sufficiently small, i.e., the step-sizes are close to each other. We also acknowledge that similar to the centralized heavy-ball method [50, 51], dating back to more than 50 years, and the recent work [52–58], a *global* acceleration can only be shown via numerical simulations. Following the standard practice, we provide simulations to verify that $\mathcal{AB}m$ has accelerated convergence, the effect of which is more pronounced when the global objective function is ill-conditioned.

We now describe the rest of the paper. Section II provides preliminaries, problem formulation, and introduces distributed heavy-ball, i.e., the $\mathcal{AB}m$ algorithm. Section III establishes the connection between $\mathcal{AB}$ and related algorithms. Section IV includes the main results on the convergence analysis, whereas Section V provides a family of average-consensus algorithms that result naturally from $\mathcal{AB}m$. Finally, Section VI provides numerical experiments and Section VII concludes the paper.

[1]Simultaneous application of both row- and column-stochastic weights was first employed for average-consensus in [18] and towards distributed optimization in [16, 17], albeit without gradient tracking.

Basic Notation: We use lowercase bold letters to denote vectors and uppercase letters for matrices. The matrix, I_n, is the $n \times n$ identity, whereas $\mathbf{1}_n$ ($\mathbf{0}_n$) is the n-dimensional column vector of all ones (zeros). For an arbitrary vector, $\mathbf{x}$, we denote its ith element by $[\mathbf{x}]_i$ and its largest and smallest element by $[\mathbf{x}]_{\max}$ and $[\mathbf{x}]_{\min}$, respectively. We use $\mathrm{diag}(\mathbf{x})$ to denote a diagonal matrix that has $\mathbf{x}$ on its main diagonal. For two matrices, X and Y, $\mathrm{diag}\,(X, Y)$ is a block-diagonal matrix with X and Y on its main diagonal, and $X \otimes Y$ denotes their Kronecker product. The spectral radius of a matrix, X, is represented by $\rho(X)$. For a primitive, row-stochastic matrix, A, we denote its left and right eigenvectors corresponding to the eigenvalue of 1 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, B, we denote its left and right eigenvectors corresponding to the eigenvalue of 1 by $\mathbf{1}_n$ and $\boldsymbol{\pi}_c$, respectively, such that $\mathbf{1}_n^\top \boldsymbol{\pi}_c = 1$. For a matrix X, we denote X_∞ as its infinite power (if it exists), i.e., $X_\infty = \lim_{k\to\infty} X^k$. From the Perron-Frobenius theorem [59], we have $A_\infty = \mathbf{1}_n \boldsymbol{\pi}_r^\top$ and $B_\infty = \boldsymbol{\pi}_c \mathbf{1}_n^\top$. We denote $\|\cdot\|_{\mathcal{A}}$ and $\|\cdot\|_{\mathcal{B}}$ as some arbitrary vector norms, the choice of which will be clear in Lemma 1, while $\|\cdot\|$ denotes the Euclidean matrix and vector norms.

II. Preliminaries and Problem Formulation

Consider n agents connected over a directed graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \cdots, n\}$ is the set of agents, and $\mathcal{E}$ is the collection of ordered pairs, $(i, j), i, j \in \mathcal{V}$, such that agent j can send information to agent i, i.e., $j \to i$. We define $\mathcal{N}_i^{\text{in}}$ as the collection of in-neighbors of agent i, 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 solve the following problem:

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

where each $f_i : \mathbb{R}^p \to \mathbb{R}$ is known only to agent i. We formalize the set of assumptions as follows.

Assumption A1. *The graph, $\mathcal{G}$, is strongly-connected.*

Assumption A2. *Each local objective, f_i, is μ_i-strongly-convex, i.e., $\forall i \in \mathcal{V}$ and $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$, we have*

$$f_i(\mathbf{y}) \geq f_i(\mathbf{x}) + \nabla f_i(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) + \frac{\mu_i}{2}\|\mathbf{x} - \mathbf{y}\|^2,$$

where $\mu_i \geq 0$ and $\sum_{i=1}^n \mu_i > 0$.

Assumption A3. *Each local objective, f_i, is l_i-smooth, i.e., its gradient is Lipschitz-continuous: $\forall i \in \mathcal{V}$ and $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$, we have, for some $l_i > 0$,*

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq l_i \|\mathbf{x} - \mathbf{y}\|.$$

Assumptions A2 and A3 ensure that the global minimizer, $\mathbf{x}^* \in \mathbb{R}^p$, of F exists and is unique [60]. In the subsequent analysis, we use $\mu \triangleq \frac{1}{n}\sum_{i=1}^n \mu_i$ and $l \triangleq \frac{1}{n}\sum_{i=1}^n l_i$, as the strong-convexity and Lipschitz-continuity constants, respectively, for the global objective, F. We define $\bar{l} \triangleq \max_i l_i$. We next describe the heavy-ball method that is credited to Polyak and then introduce the distributed heavy-ball method, termed as the $\mathcal{AB}m$ algorithm, to solve Problem P1.

A. Heavy-ball method

It is well known [51, 60] that the best achievable convergence rate of the gradient descent algorithm,

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \nabla F(\mathbf{x}_k),$$

is $\mathcal{O}((\frac{\mathcal{Q}-1}{\mathcal{Q}+1})^k)$, where $\mathcal{Q} \triangleq \frac{l}{\mu}$ is the condition number of the objective function, F. Clearly, gradient descent is quite slow when $\mathcal{Q}$ is large, i.e., when the objective function is ill-conditioned. The seminal work by Polyak [50, 51] proposes the following heavy-ball method:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha \nabla F(\mathbf{x}_k) + \beta(\mathbf{x}_k - \mathbf{x}_{k-1}), \tag{1}$$

where $\beta(\mathbf{x}_k - \mathbf{x}_{k-1})$ is interpreted as a "momentum" term, used to accelerate the convergence process. Polyak shows that with a specific choice of α and β, the heavy-ball method achieves a *local* accelerated rate of $\mathcal{O}((\frac{\sqrt{\mathcal{Q}}-1}{\sqrt{\mathcal{Q}}+1})^k)$. By local, it is meant that the acceleration can only be analytically shown when $\|\mathbf{x}_0 - \mathbf{x}^*\|$ is sufficiently small. Globally, i.e., for arbitrary initial conditions, only linear convergence is established, while an analytical characterization of the acceleration is still an open problem, see related work in [52, 53, 56–58]. Numerical analysis and simulations are often employed to show global acceleration, i.e., it is possible to tune α and β such that the heavy-ball method is faster than gradient descent [54, 55].

B. Distributed heavy-ball: The $\mathcal{AB}m$ algorithm

Recall, that our goal is to solve Problem P1 when the agents, possessing only local objectives, exchange information over a strongly-connected directed graph, $\mathcal{G}$. Each agent, $i \in \mathcal{V}$, maintains two variables: $\mathbf{x}_k^i,\ \mathbf{y}_k^i \in \mathbb{R}^p$, where $\mathbf{x}_k^i$ is the local estimate of the global minimizer and $\mathbf{y}_k^i$ is an auxiliary variable. The $\mathcal{AB}m$ algorithm, initialized with arbitrary $\mathbf{x}_0^i$'s, $\mathbf{x}_{-1}^i = \mathbf{0}_p$ and $\mathbf{y}_0^i = \nabla f_i(\mathbf{x}_0^i), \forall i \in \mathcal{V}$, is given by[2]:

$$\mathbf{x}_{k+1}^i = \sum_{j=1}^n a_{ij}\mathbf{x}_k^j - \alpha_i \mathbf{y}_k^i + \beta_i \left(\mathbf{x}_k^i - \mathbf{x}_{k-1}^i\right), \tag{2a}$$

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

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are respectively the local step-size and the momentum parameter adopted by agent i. The weights, a_{ij}'s and b_{ij}'s, are associated with the graph topology and satisfy the following conditions:

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

$$b_{ij} = \begin{cases} > 0, & i \in \mathcal{N}_j^{\text{out}}, \\ 0, & \text{otherwise}, \end{cases} \qquad \sum_{i=1}^n b_{ij} = 1, \forall j.$$

[2]We note that several variants of this algorithm can be extracted by considering an adapt-then-combine update, e.g., $\sum_{j=1}^n b_{ij}(\mathbf{y}_k^j + \nabla f_i(\mathbf{x}_{k+1}^i) - \nabla f_i\big(\mathbf{x}_k^i\big))$, see [37], instead of the combine-then-adapt update that we have used here in Eq. (2b). The momentum term in Eq. (2a) can also be integrated similarly. We choose one of the applicable forms and note that extensions to other cases follow from this exposition and the subsequent analysis.

Note that the weight matrix, $A = \{a_{ij}\}$, in Eq. (2a) is RS (row-stochastic) and the weight matrix, $B = \{b_{ij}\}$ in Eq. (2b) is CS (column-stochastic), both of which can be implemented over undirected and directed graphs alike. Intuitively, Eq. (2b) tracks the average of local gradients, $\frac{1}{n}\sum_{i=1}^n \nabla f_i(\mathbf{x}_k^i)$, see [31–39, 41], and therefore Eq. (2a) asymptotically approaches the centralized heavy-ball, Eq. (1), as the descent direction $\mathbf{y}_k^i$ becomes the gradient of the global objective.

Vector form: For the sake of analysis, we now write $\mathcal{AB}m$ in vector form. We use the following notation:

$$\mathbf{x}_k \triangleq \begin{bmatrix} \mathbf{x}_k^1 \\ \vdots \\ \mathbf{x}_k^n \end{bmatrix},\ \mathbf{y}_k \triangleq \begin{bmatrix} \mathbf{y}_k^1 \\ \vdots \\ \mathbf{y}_k^n \end{bmatrix},\ \nabla\mathbf{f}(\mathbf{x}_k) \triangleq \begin{bmatrix} \nabla f_1(\mathbf{x}_k^1) \\ \vdots \\ \nabla f_n(\mathbf{x}_k^n) \end{bmatrix},$$

all in $\mathbb{R}^{np}$. Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ define the vectors of the step-sizes and the momentum parameters, respectively. We now define augmented weight matrices, $\mathcal{A}, \mathcal{B}$, and augmented step-size and momentum matrices, $D_{\boldsymbol{\alpha}}, D_{\boldsymbol{\beta}}$:

$$\mathcal{A} \triangleq A \otimes I_p, \qquad D_{\boldsymbol{\alpha}} \triangleq \text{diag}(\boldsymbol{\alpha}) \otimes I_p,$$
$$\mathcal{B} \triangleq B \otimes I_p, \qquad D_{\boldsymbol{\beta}} \triangleq \text{diag}(\boldsymbol{\beta}) \otimes I_p,$$

all in $\mathbb{R}^{np\times np}$. Using the notation above, $\mathcal{AB}m$ can be compactly written as:

$$\mathbf{x}_{k+1} = \mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}\left(\mathbf{x}_k - \mathbf{x}_{k-1}\right), \tag{3a}$$

$$\mathbf{y}_{k+1} = \mathcal{B}\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k), . \tag{3b}$$

We note here that when $\beta_i = 0, \forall i$, $\mathcal{AB}m$ reduces to $\mathcal{AB}$ [37], albeit with two distinguishing features: (i) the algorithm in [37] uses an identical step-size, α, at each agent; and (ii) Eq. (2b) in [37] is in an adapt-then-combine form.

III. Connection with existing first-order methods

In this section, we provide a generalization of several existing methods that employ gradient tracking [31–36] and show that $\mathcal{AB}$ lies at the heart of these approaches. To proceed, we rewrite the $\mathcal{AB}$ updates below (without momentum) [37].

$$\mathbf{x}_{k+1} = \mathcal{A}\mathbf{x}_k - \alpha\mathbf{y}_k, \tag{4a}$$

$$\mathbf{y}_{k+1} = \mathcal{B}\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k). \tag{4b}$$

Since $\mathcal{AB}$ uses both RS and CS weights simultaneously, it is natural to ask how are the optimization algorithms that require the weight matrices to be doubly-stochastic (DS) [26, 31, 32, 34], or only CS [33, 34], or only RS [35, 36], are related to each other. We discuss this relationship next.

Optimization with DS weights: Refs. [31, 32, 34] consider the following updates, termed as Aug-DGM in [31] and DIGing in [34]:

$$\mathbf{x}_{k+1} = \mathcal{W}\mathbf{x}_k - \alpha\mathbf{y}_k, \tag{5a}$$

$$\mathbf{y}_{k+1} = \mathcal{W}\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k), \tag{5b}$$

where $\mathcal{W} = W \otimes I_p$, and W is a DS weight matrix. Clearly, to obtain DS weights, the underlying graph must be undirected (or balanced) and thus the algorithm in Eqs. (5) is not applicable to arbitrary directed graphs. That $\mathcal{AB}$ generalizes

Eqs. (5) is straightforward as the DS weights naturally satisfy the RS requirement in the top update and the CS requirement in the bottom update, while the reverse is not true. Similarly, we note that a related algorithm, EXTRA [26], is given by

$$\mathbf{x}_{k+1} = (I + \mathcal{W})\mathbf{x}_k - \widetilde{\mathcal{W}}\mathbf{x}_{k-1} - \alpha\left(\nabla\mathbf{f}(\mathbf{x}_k) - \nabla\mathbf{f}(\mathbf{x}_{k-1})\right),$$

where the two weight matrices, $\mathcal{W}$ and $\widetilde{\mathcal{W}}$, must be symmetric and satisfy some other stringent requirements, see [26] for details. Eliminating the $\mathbf{y}_k$-update in $\mathcal{AB}$, we note that $\mathcal{AB}$ can be written in the EXTRA format as follows:

$$\begin{aligned}\mathbf{x}_{k+1} =& (I + (\mathcal{A}+\mathcal{B}-I))\,\mathbf{x}_k \\ &- (\mathcal{B}\mathcal{A})\,\mathbf{x}_{k-1} - \alpha\left(\nabla\mathbf{f}(\mathbf{x}_k) - \nabla\mathbf{f}(\mathbf{x}_{k-1})\right). \end{aligned} \tag{6}$$

It can be seen that the linear convergence of $\mathcal{AB}$ does not follow from the analysis in [26] as $\mathcal{A}+\mathcal{B}-I$ and $\mathcal{BA}$ are not necessarily symmetric. Analysis of the $\mathcal{AB}$ algorithm, therefore, generalizes that of EXTRA to non-doubly-stochastic and non-symmetric weight matrices.

Optimization with CS weights: We now relate $\mathcal{AB}$ to ADD-OPT/Push-DIGing that only require CS weights [33, 34]. Since $\mathcal{B}$ is already CS in $\mathcal{AB}$, it suffices to seek a state transformation that transforms $\mathcal{A}$ from RS to CS, while respecting the graph topology. To this aim, let us consider the following transformation on the $\mathbf{x}_k$-update in $\mathcal{AB}$: $\widetilde{\mathbf{x}}_k \triangleq \Pi_r\mathbf{x}_k$, where $\Pi_r \triangleq \text{diag}(n\boldsymbol{\pi}_r)\otimes I_p$ and $\boldsymbol{\pi}_r$ is the left-eigenvector of the RS weight matrix, A, corresponding to the eigenvalue 1. The resulting transformed $\mathcal{AB}$ is given by

$$\widetilde{\mathbf{x}}_{k+1} = \widetilde{\mathcal{B}}\,\widetilde{\mathbf{x}}_k - \alpha\Pi_r\mathbf{y}_k, \tag{7a}$$

$$\mathbf{x}_{k+1} = \left(\text{diag}(n\boldsymbol{\pi}_r)\otimes I_p\right)^{-1}\widetilde{\mathbf{x}}_{k+1}, \tag{7b}$$

$$\mathbf{y}_{k+1} = \mathcal{B}\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k), \tag{7c}$$

where it is straightforward to show that $\widetilde{\mathcal{B}} = \Pi_r\mathcal{A}\Pi_r^{-1}$ is now CS and $\widetilde{\mathcal{B}}\left(\boldsymbol{\pi}_r\otimes I_p\right) = \boldsymbol{\pi}_r\otimes I_p$.

In order to implement the above equations, two different CS matrices ($\widetilde{\mathcal{B}}$ and $\mathcal{B}$) suffice, as long as they are primitive and respect the graph topology. The second update requires the right-eigenvector of the CS matrix used in the first update, i.e., $\widetilde{\mathcal{B}}$. Since this eigenvector is not known locally to any agent, ADD-OPT/Push-DIGing [33, 34] propose learning this eigenvector with the following iterations: $\mathbf{w}_{k+1} = \widetilde{\mathcal{B}}\mathbf{w}_k, \mathbf{w}_0 = \mathbf{1}_{np}$. The algorithms provided in [33, 34] essentially implement Eqs. (7), albeit with two differences: (i) the same CS weight matrix is used in all updates; and, (ii) the division in Eq. (7b) is replaced by the estimated component, $\mathbf{w}_{k+1}^i$, of the left-eigenvector at each agent. This nonlinearity causes stability issues in ADD-OPT/Push-DIGing, whereas their convergence compared to $\mathcal{AB}$ is slower because such an eigenvector estimation is not needed in the latter on the account of using the RS weights. Furthermore, the local step-sizes are now given by $n\alpha[\boldsymbol{\pi}_r]_i$ that shows that ADD-OPT/Push-DIGing should work with nonidentical step-sizes.

Optimization with RS weights: The state transformation technique discussed above also leads to an algorithm from $\mathcal{AB}$ that only requires RS weights. Since $\mathcal{A}$ in $\mathcal{AB}$ is RS, a transformation now is imposed on the $\mathbf{y}_k$-update and is given by $\widetilde{\mathbf{y}}_k \triangleq \Pi_c^{-1}\mathbf{y}_k$, where $\Pi_c \triangleq \text{diag}(\boldsymbol{\pi}_c)\otimes I_p$, and $\boldsymbol{\pi}_c$ is the right-eigenvector of the CS weight matrix, B, corresponding to the eigenvalue 1. Equivalently, $\mathcal{AB}$ is given by

$$\mathbf{x}_{k+1} = \mathcal{A}\mathbf{x}_k - \alpha\Pi_c\widetilde{\mathbf{y}}_k, \tag{8a}$$

$$\widetilde{\mathbf{y}}_{k+1} = \widetilde{\mathcal{A}}\widetilde{\mathbf{y}}_k + \Pi_c^{-1}\left(\nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k)\right), \tag{8b}$$

where $\widetilde{\mathcal{A}} = \Pi_c^{-1}\mathcal{B}\Pi_c$ is now RS and $\left(\boldsymbol{\pi}_c^\top\otimes I_p\right)\widetilde{\mathcal{A}} = \boldsymbol{\pi}_c^\top\otimes I_p$. Since the above form of $\mathcal{AB}$ cannot be implemented because $\boldsymbol{\pi}_c$ is not locally known, an eigenvector estimation is used in FROST [35, 36] and the division in Eq. (8b) is replaced with the appropriate estimated component of $\boldsymbol{\pi}_c$. The observations on different weight matrices in the two updates, nonidentical step-sizes, stability, and convergence made earlier for ADD-OPT/Push-DIGing are also applicable here.

In conclusion, the $\mathcal{AB}$ algorithm has various equivalent representations and several already-known protocols can in fact be derived from these representations. In a similar way, $\mathcal{AB}m$ leads to protocols that add momentum to Aug-DGM, ADD-OPT/Push-DIGing, and FROST. We will revisit the relationship and equivalence cast here in Sections V and VI. In Section V, we will show that both $\mathcal{AB}$ and $\mathcal{AB}m$ naturally provide a non-trivial class of average-consensus algorithms, a special case of which are [46] and surplus consensus [18]. In Section VI, we will compare these algorithms numerically.

IV. Convergence Analysis

We now start the convergence analysis of the proposed distributed heavy-ball method, $\mathcal{AB}m$. In the following, we first provide some auxiliary results borrowed from the literature.

A. Auxiliary Results

The following lemma establishes contractions with RS and CS matrices under arbitrary norms [37]; note thacontraction in the Euclidean norm is not applicable unless the weight matrix is DS as in [32, 34]. A similar result was first presented in [33] for CS matrices, and later in [35, 36] for RS matrices.

Lemma 1. *Consider the augmented weight matrices $\mathcal{A}$ and $\mathcal{B}$. There exist vector norms, denoted as $\|\cdot\|_{\mathcal{A}}$ and $\|\cdot\|_{\mathcal{B}}$, such that $\forall\mathbf{x}\in\mathbb{R}^{np}$,*

$$\|\mathcal{A}\mathbf{x} - \mathcal{A}_\infty\mathbf{x}\|_{\mathcal{A}} \leq \sigma_{\mathcal{A}}\|\mathbf{x} - \mathcal{A}_\infty\mathbf{x}\|_{\mathcal{A}}, \tag{9}$$

$$\|\mathcal{B}\mathbf{x} - \mathcal{B}_\infty\mathbf{x}\|_{\mathcal{B}} \leq \sigma_{\mathcal{B}}\|\mathbf{x} - \mathcal{B}_\infty\mathbf{x}\|_{\mathcal{B}}, \tag{10}$$

where $0<\sigma_{\mathcal{A}}<1$ and $0<\sigma_{\mathcal{B}}<1$ are some constants.

The next lemma from [37] states that the sum of $\mathbf{y}_k^i$'s preserves the sum of local gradients. This is a direct consequence of the dynamic consensus [41] employed with CS weights in the $\mathbf{y}_k$-update of $\mathcal{AB}m$.

Lemma 2. $(\mathbf{1}_n^\top\otimes I_p)\mathbf{y}_k = (\mathbf{1}_n^\top\otimes I_p)\nabla\mathbf{f}(\mathbf{x}_k), \forall k.$

The next lemma is standard in the convex optimization theory [61]. It states that the distance to the optimizer contracts at each step in the standard gradient descent method.

Lemma 3. *Let F be μ-strongly-convex and l-smooth. For $0<\alpha<\frac{2}{l}$, we have*

$$\|\mathbf{x} - \alpha\nabla F(\mathbf{x}) - \mathbf{x}^*\| \leq \sigma_F\|\mathbf{x}-\mathbf{x}^*\|,$$

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

Finally, we provide a result from nonnegative matrix theory.

Lemma 4. *(Theorem 8.1.29 in [59]) Let $X \in \mathbb{R}^{n\times n}$ be a nonnegative matrix and $\mathbf{x} \in \mathbb{R}^n$ be a positive vector. If $X\mathbf{x} < \omega\mathbf{x}$ with $\omega > 0$, then $\rho(X) < \omega$.*

B. Main results

The convergence analysis of $\mathcal{AB}m$ is based on deriving a contraction relationship between the following four quantities: (i) $\|\mathbf{x}_{k+1} - \mathcal{A}_\infty\mathbf{x}_{k+1}\|_\mathcal{A}$, the consensus error in the network; (ii) $\|\mathcal{A}_\infty\mathbf{x}_{k+1} - \mathbf{1}_n \otimes \mathbf{x}^*\|$, the optimality gap; (iii) $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|$, the state difference; and (iv) $\|\mathbf{y}_{k+1} - \mathcal{B}_\infty\mathbf{y}_{k+1}\|_\mathcal{B}$, the (biased) gradient estimation error. We will establish an LTI-system inequality where the state vector is the collection of these four quantities and then develop the convergence properties of the corresponding system matrix. Before we proceed, note that since all vector norms on finite-dimensional vector spaces are equivalent [59], there exist positive constants $c_{\mathcal{AB}}, c_{\mathcal{BA}}, c_{2\mathcal{A}}, c_{\mathcal{A}2}, c_{2\mathcal{B}}, c_{\mathcal{B}2}$ such that

$$\|\cdot\|_\mathcal{A} \le c_{\mathcal{AB}}\|\cdot\|_\mathcal{B}, \quad \|\cdot\| \le c_{2\mathcal{A}}\|\cdot\|_\mathcal{A}, \quad \|\cdot\|_\mathcal{A} \le c_{\mathcal{A}2}\|\cdot\|,$$
$$\|\cdot\|_\mathcal{B} \le c_{\mathcal{BA}}\|\cdot\|_\mathcal{A}, \quad \|\cdot\| \le c_{2\mathcal{B}}\|\cdot\|_\mathcal{B}, \quad \|\cdot\|_\mathcal{B} \le c_{\mathcal{B}2}\|\cdot\|.$$

We also define $\overline{\alpha} \triangleq [\boldsymbol{\alpha}]_{\max}$ and $\overline{\beta} \triangleq [\boldsymbol{\beta}]_{\max}$. In the following, we first provide an upper bound on the estimate, $\mathbf{y}_k$, of the gradient of the global objective that will be useful in deriving the aforementioned LTI system.

Lemma 5. *The following inequality holds, $\forall k$:*

$$\|\mathbf{y}_k\| \le c_{2\mathcal{A}}\bar{l}\,\|\mathcal{B}_\infty\|\,\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_\mathcal{A} + c_{2\mathcal{B}}\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_\mathcal{B} + \bar{l}\,\|\mathcal{B}_\infty\|\,\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|.$$

Proof. Recall that $\mathcal{B}_\infty = (\boldsymbol{\pi}_c \otimes I_p)(\mathbf{1}_n^\top \otimes I_p)$. We have

$$\|\mathbf{y}_k\| \le c_{2\mathcal{B}}\,\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_\mathcal{B} + \|\mathcal{B}_\infty\mathbf{y}_k\|. \tag{11}$$

We next bound $\|\mathcal{B}_\infty\mathbf{y}_k\|$:

$$\begin{aligned}\|\mathcal{B}_\infty\mathbf{y}_k\| &= \|(\boldsymbol{\pi}_c\otimes I_p)(\mathbf{1}_n^\top\otimes I_p)\nabla\mathbf{f}(\mathbf{x}_k)\|,\\ &= \|\boldsymbol{\pi}_c\|\,\left\|\textstyle\sum_{i=1}^n\nabla f_i(\mathbf{x}_k^i) - \sum_{i=1}^n\nabla f_i(\mathbf{x}^*)\right\|,\\ &\le \|\boldsymbol{\pi}_c\|\;\bar{l}\;\sqrt{n}\|\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|,\\ &\le c_{2\mathcal{A}}\;\bar{l}\;\|\mathcal{B}_\infty\|\,\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_\mathcal{A}\\ &\quad+\;\bar{l}\;\|\mathcal{B}_\infty\|\,\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|,\end{aligned} \tag{12}$$

where the first inequality uses Jensen's inequality and the last inequality uses the fact that $\|\mathcal{B}_\infty\| = \sqrt{n}\|\boldsymbol{\pi}_c\|$. The lemma follows by plugging Eq. (12) into Eq. (11). □

In the next Lemmas 6-9, we derive the relationships among the four quantities mentioned above. We start with a bound on $\|\mathbf{x}_{k+1} - \mathcal{A}_\infty\mathbf{x}_{k+1}\|_\mathcal{A}$, the consensus error in the network.

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

$$\begin{aligned}&\|\mathbf{x}_{k+1} - \mathcal{A}_\infty\mathbf{x}_{k+1}\|_\mathcal{A}\\ &\le \left(\sigma_\mathcal{A} + \overline{\alpha}c_{\mathcal{A}2}c_{2\mathcal{A}}\;\bar{l}\,\|I_{np} - \mathcal{A}_\infty\|\,\|\mathcal{B}_\infty\|\right)\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_\mathcal{A}\\ &\quad+\overline{\alpha}c_{\mathcal{A}2}\;\bar{l}\,\|I_{np} - \mathcal{A}_\infty\|\,\|B_\infty\|\,\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|\\ &\quad+\overline{\beta}c_{\mathcal{A}2}\,\|I_{np} - \mathcal{A}_\infty\|\,\|\mathbf{x}_k - \mathbf{x}_{k-1}\|\\ &\quad+\overline{\alpha}c_{\mathcal{A}2}c_{2\mathcal{B}}\,\|I_{np} - \mathcal{A}_\infty\|\,\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_\mathcal{B}.\end{aligned}$$

Proof. First, note that $\mathcal{A}_\infty\mathcal{A} = \mathcal{A}_\infty$. Following the $\mathbf{x}_k$-update of $\mathcal{AB}m$ in Eq. (3a) and using the one-step contraction property of $\mathcal{A}$ from Lemma 1, we have:

$$\begin{aligned}&\left\|\mathbf{x}_{k+1} - \mathcal{A}_\infty\mathbf{x}_{k+1}\right\|_\mathcal{A}\\ &=\left\|\mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1})\right.\\ &\quad\left.- \mathcal{A}_\infty\left(\mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1})\right)\right\|_\mathcal{A},\\ &\le \sigma_\mathcal{A}\,\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_\mathcal{A} + \overline{\alpha}\;c_{\mathcal{A}2}\,\|I_{np} - \mathcal{A}_\infty\|\,\|\mathbf{y}_k\|\\ &\quad+\overline{\beta}\;c_{\mathcal{A}2}\,\|I_{np} - \mathcal{A}_\infty\|\,\|\mathbf{x}_k - \mathbf{x}_{k-1}\|,\end{aligned}$$

and the proof follows from Lemma 5. □

Next, we derive a bound for $\|\mathcal{A}_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|$, which can be interpreted as the optimality gap between the network accumulation state, $\mathcal{A}_\infty\mathbf{x}_k$, and the global minimizer, $\mathbf{1}_n\otimes\mathbf{x}^*$.

Lemma 7. *The following inequality holds, $\forall k$, when $0 < \boldsymbol{\pi}_r^\top diag(\boldsymbol{\alpha})\boldsymbol{\pi}_c < \frac{2}{nl}$:*

$$\begin{aligned}&\|\mathcal{A}_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|\\ &\quad\le\overline{\alpha}\left(\boldsymbol{\pi}_r^\top\boldsymbol{\pi}_c\right)n\bar{l}c_{2\mathcal{A}}\,\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_\mathcal{A}\\ &\qquad+\lambda\,\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n\otimes\mathbf{x}^*\|\\ &\qquad+\beta\|\mathcal{A}_\infty\|\|\mathbf{x}_k - \mathbf{x}_{k-1}\|\\ &\qquad+\overline{\alpha}c_{2B}\|\mathcal{A}_\infty\|\,\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_\mathcal{B},\end{aligned} \tag{13}$$

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

Proof. Recall the $\mathbf{x}_k$-update of $\mathcal{AB}m$ in Eq. (3a), we have that

$$\begin{aligned}&\|\mathcal{A}_\infty\mathbf{x}_{k+1} - \mathbf{1}_n\otimes\mathbf{x}^*\|\\ &\quad=\left\|\mathcal{A}_\infty\left(\mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1})\right) - \mathbf{1}_n\otimes\mathbf{x}^*\right\|,\\ &\quad=\left\|\mathcal{A}_\infty\left(\mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + (D_{\boldsymbol{\alpha}} - D_{\boldsymbol{\alpha}})\mathcal{B}_\infty\mathbf{y}_k\right.\right.\\ &\qquad\left.\left.+\,D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1})\right) - \mathbf{1}_n\otimes\mathbf{x}^*\right\|,\\ &\quad\le\|\mathcal{A}_\infty\mathbf{x}_k - \mathcal{A}_\infty D_{\boldsymbol{\alpha}}\mathcal{B}_\infty\nabla\mathbf{f}\left(\mathbf{x}_k\right) - \left(\mathbf{1}_n\otimes I_p\right)\mathbf{x}^*\|\\ &\qquad+\overline{\beta}\|\mathcal{A}_\infty\|\|\mathbf{x}_k - \mathbf{x}_{k-1}\|\\ &\qquad+\overline{\alpha}c_{2B}\|\mathcal{A}_\infty\|\,\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_\mathcal{B},\end{aligned} \tag{14}$$

where in the last inequality, we use $\mathcal{B}_\infty\mathbf{y}_k = \mathcal{B}_\infty\nabla\mathbf{f}\left(\mathbf{x}_k\right)$ adapted from Lemma 2. Since the last two terms in Eq. (14) match the last two terms in Eq. (13), what is left is to bound the first term. Before we proceed, define

$$\widetilde{\mathbf{x}}_k \triangleq (\boldsymbol{\pi}_r^\top\otimes I_p)\mathbf{x}_k,$$
$$\nabla\mathbf{f}\left((\mathbf{1}_n\otimes I_p)\widetilde{\mathbf{x}}_k\right) \triangleq \left[\nabla f_1(\widetilde{\mathbf{x}}_k)^\top, \cdots, \nabla f_n(\widetilde{\mathbf{x}}_k)^\top\right]^\top,$$

and note that

$$\begin{aligned}&\mathcal{A}_\infty D_{\boldsymbol{\alpha}}\mathcal{B}_\infty\\ &=\left(\mathbf{1}_n\otimes I_p\right)\left(\boldsymbol{\pi}_r^\top\otimes I_p\right)\left(\mathrm{diag}(\boldsymbol{\alpha})\otimes I_p\right)\left(\boldsymbol{\pi}_c\otimes I_p\right)\left(\mathbf{1}_n^\top\otimes I_p\right)\\ &=\left(\boldsymbol{\pi}_r^\top\mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\right)\left(\mathbf{1}_n\otimes I_p\right)\left(\mathbf{1}_n^\top\otimes I_p\right).\end{aligned}$$

Now we bound the first term in Eq. (14). We have

$$\begin{aligned}&\|\mathcal{A}_\infty\mathbf{x}_k - \mathcal{A}_\infty D_{\boldsymbol{\alpha}}\mathcal{B}_\infty\nabla\mathbf{f}(\mathbf{x}_k) - (\mathbf{1}_n\otimes I_p)\,\mathbf{x}^*\|\\ &=\left\|\left(\mathbf{1}_n\otimes I_p\right)\left(\widetilde{\mathbf{x}}_k - (\boldsymbol{\pi}_r^\top\mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c)(\mathbf{1}_n^\top\otimes I_p)\nabla\mathbf{f}(\mathbf{x}_k) - \mathbf{x}^*\right)\right\|,\\ &\le\left\|\left(\mathbf{1}_n\otimes I_p\right)\left(\widetilde{\mathbf{x}}_k - n(\boldsymbol{\pi}_r^\top\mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c)\nabla F(\widetilde{\mathbf{x}}_k) - \mathbf{x}^*\right)\right\|\\ &\quad+\boldsymbol{\pi}_r^\top\mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\left\|\left(\mathbf{1}_n\otimes I_p\right)\left(n\nabla F(\widetilde{\mathbf{x}}_k) - (\mathbf{1}_n^\top\otimes I_p)\nabla\mathbf{f}(\mathbf{x}_k)\right)\right\|,\\ &\triangleq s_1 + s_2,\end{aligned}$$

and we bound s_1 and s_2 next. Using Lemma 3, we have that if $0 < \boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c < \frac{2}{nl}$,

$$\begin{aligned} s_1 &= \sqrt{n}\left\|\widetilde{\mathbf{x}}_k - n(\boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c)\nabla F(\widetilde{\mathbf{x}}_k) - \mathbf{x}^*\right\|, \\ &\leq \sqrt{n}\lambda\left\|\widetilde{\mathbf{x}}_k - \mathbf{x}^*\right\|, \\ &= \lambda\left\|\mathcal{A}_\infty \mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\right\|, \end{aligned} \tag{15}$$

where $\lambda = \max\left\{\left|1 - \mu n \boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\right|, \left|1 - ln\boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\right|\right\}$. We next bound s_2. Since $\nabla F(\widetilde{\mathbf{x}}_k) = \frac{1}{n}(\mathbf{1}_n^\top \otimes I_p)\nabla \mathbf{f}(\widetilde{\mathbf{x}}_k)$,

$$\begin{aligned} s_2 &\leq \left(\boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\right) n \left\|\nabla \mathbf{f}\left((\mathbf{1}_n \otimes I_p)\widetilde{\mathbf{x}}_k\right) - \nabla \mathbf{f}(\mathbf{x}_k)\right\|, \\ &\leq \left(\boldsymbol{\pi}_r^\top \text{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c\right) n\bar{l}c_{2\mathcal{A}} \left\|\mathbf{x}_k - \mathcal{A}_\infty \mathbf{x}_k\right\|_{\mathcal{A}}, \\ &\leq \overline{\alpha}\left(\boldsymbol{\pi}_r^\top \boldsymbol{\pi}_c\right) n\bar{l}c_{2\mathcal{A}} \left\|\mathbf{x}_k - \mathcal{A}_\infty \mathbf{x}_k\right\|_{\mathcal{A}}, \end{aligned} \tag{16}$$

and the lemma follows from Eqs. (15), (16), and (14). □

The next step is to bound the state difference, $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|$.

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

$$\begin{aligned} &\|\mathbf{x}_{k+1} - \mathbf{x}_k\| \\ &\leq \left(c_{2\mathcal{A}}\|\mathcal{A} - I_{np}\| + \overline{\alpha}c_{2\mathcal{A}}\bar{l}\|\mathcal{B}_\infty\|\right)\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_{\mathcal{A}} \\ &\quad + \overline{\alpha}\bar{l}\|\mathcal{B}_\infty\|\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\| \\ &\quad + \overline{\beta}\|\mathbf{x}_k - \mathbf{x}_{k-1}\| + \overline{\alpha}c_{2\mathcal{B}}\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_{\mathcal{B}}. \end{aligned}$$

Proof. Note that $\mathcal{A}\mathcal{A}_\infty = \mathcal{A}_\infty$ and hence $\mathcal{A}\mathcal{A}_\infty - \mathcal{A}_\infty$ is a zero matrix. Following the $\mathbf{x}_k$-update of $\mathcal{AB}m$, we have:

$$\begin{aligned} &\|\mathbf{x}_{k+1} - \mathbf{x}_k\| \\ &= \|\mathcal{A}\mathbf{x}_k - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1}) - \mathbf{x}_k\|, \\ &= \|(\mathcal{A} - I_{np})(\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k) - D_{\boldsymbol{\alpha}}\mathbf{y}_k + D_{\boldsymbol{\beta}}(\mathbf{x}_k - \mathbf{x}_{k-1})\|, \\ &\leq c_{2\mathcal{A}}\|\mathcal{A} - I_{np}\|\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_{\mathcal{A}} + \overline{\beta}\|\mathbf{x}_k - \mathbf{x}_{k-1}\| + \overline{\alpha}\|\mathbf{y}_k\|, \end{aligned}$$

and the proof follows from Lemma 5. □

The final step in formulating the LTI system is to write $\|\mathbf{y}_{k+1} - \mathcal{B}_\infty\mathbf{y}_{k+1}\|$, the biased gradient estimation error, in terms of the other three quantities. We call this biased to make a distinction with the unbiased gradient estimation error: $\|\mathbf{y}_{k+1} - \mathcal{W}_\infty\mathbf{y}_{k+1}\|$, where $\mathcal{W}$ is doubly-stochastic.

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

$$\begin{aligned} &\|\mathbf{y}_{k+1} - \mathcal{B}_\infty\mathbf{y}_{k+1}\| \\ &= \Big(c_{2\mathcal{A}}c_{\mathcal{B}2}\ \bar{l}\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{A} - I_{np}\| \\ &\quad + \overline{\alpha}c_{2\mathcal{A}}c_{\mathcal{B}2}\ \bar{l}^2\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{B}_\infty\|\Big)\|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_{\mathcal{A}} \\ &\quad + \overline{\alpha}c_{\mathcal{B}2}\ \bar{l}^2\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{B}_\infty\|\|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\| \\ &\quad + \overline{\beta}c_{\mathcal{B}2}\ \bar{l}\|I_{np} - \mathcal{B}_\infty\|\|\mathbf{x}_k - \mathbf{x}_{k-1}\| \\ &\quad + \Big(\sigma_{\mathcal{B}} + \overline{\alpha}c_{\mathcal{B}2}c_{2\mathcal{B}}\ \bar{l}\|I_{np} - \mathcal{B}_\infty\|\Big)\|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_{\mathcal{B}}. \end{aligned}$$

Proof. Note that $\mathcal{B}_\infty\mathcal{B} = \mathcal{B}_\infty$. From Eq. (3b), we have:

$$\begin{aligned} &\|\mathbf{y}_{k+1} - \mathcal{B}_\infty\mathbf{y}_{k+1}\|_{\mathcal{B}} \\ &= \big\|\mathcal{B}\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k) \\ &\quad - \mathcal{B}_\infty\big(\mathbf{y}_k + \nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k)\big)\big\|_{\mathcal{B}} \\ &\leq \sigma_{\mathcal{B}}\|\mathbf{y}(k) - \mathcal{B}_\infty\mathbf{y}(k)\|_{\mathcal{B}} + c_{\mathcal{B}2}\bar{l}\|I_{np} - \mathcal{B}_\infty\|\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2, \end{aligned}$$

where in the inequality above we use the contraction property of $\mathcal{B}$ from Lemma 1. The proof follows by applying the result of Lemma 8 to the inequality above. □

With the help of the Lemmas 6-9, we now present the main result of this paper, i.e., the $\mathcal{AB}m$ algorithm converges to the global minimizer at a global R-linear rate.

Theorem 1. *Let $0 < \boldsymbol{\pi}_r^\top diag(\boldsymbol{\alpha})\boldsymbol{\pi}_c < \frac{2}{nl}$, then the following LTI inequality holds entry-wise:*

$$\mathbf{t}_{k+1} \leq J_{\boldsymbol{\alpha},\overline{\beta}}\mathbf{t}_k, \tag{17}$$

where $\mathbf{t}_k \in \mathbb{R}^4$ and $J_{\boldsymbol{\alpha},\overline{\beta}} \in \mathbb{R}^{4\times 4}$ are respectively given by:

$$\mathbf{t}_k = \begin{bmatrix} \|\mathbf{x}_k - \mathcal{A}_\infty\mathbf{x}_k\|_{\mathcal{A}} \\ \|\mathcal{A}_\infty\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\| \\ \|\mathbf{x}_k - \mathbf{x}_{k-1}\| \\ \|\mathbf{y}_k - \mathcal{B}_\infty\mathbf{y}_k\|_{\mathcal{B}} \end{bmatrix},$$

$$J_{\boldsymbol{\alpha},\overline{\beta}} = \begin{bmatrix} \sigma_{\mathcal{A}} + a_1\overline{\alpha} & a_2\overline{\alpha} & \overline{\beta}a_3 & a_4\overline{\alpha} \\ a_5\overline{\alpha} & \lambda & \overline{\beta}a_6 & a_7\overline{\alpha} \\ a_8 + a_9\overline{\alpha} & a_{10}\overline{\alpha} & \overline{\beta} & a_{11}\overline{\alpha} \\ a_{12} + a_{13}\overline{\alpha} & a_{14}\overline{\alpha} & \overline{\beta}a_{15} & \sigma_{\mathcal{B}} + a_{16}\overline{\alpha} \end{bmatrix},$$

and the constants a_i's in the above expression are

$$\begin{aligned} a_1 &= c_{\mathcal{A}2}c_{2\mathcal{A}}\bar{l}\|I_{np} - \mathcal{A}_\infty\|\|\mathcal{B}_\infty\|, \\ a_2 &= c_{\mathcal{A}2}\bar{l}\|I_{np} - \mathcal{A}_\infty\|\|\mathcal{B}_\infty\|, \\ a_3 &= c_{\mathcal{A}2}\|I_{np} - \mathcal{A}_\infty\|, \\ a_4 &= c_{\mathcal{A}2}c_{2\mathcal{B}}\|I_{np} - \mathcal{A}_\infty\|, \\ a_5 &= nc_{2\mathcal{A}}\left(\boldsymbol{\pi}_r^\top\boldsymbol{\pi}_c\right)\bar{l}, \\ a_6 &= \|\mathcal{A}_\infty\|, \\ a_7 &= c_{2\mathcal{B}}\|\mathcal{A}_\infty\|, \\ a_8 &= c_{2\mathcal{A}}\|\mathcal{A} - I_{np}\|, \\ a_9 &= c_{2\mathcal{A}}\bar{l}\|\mathcal{B}_\infty\|, \\ a_{10} &= \bar{l}\|\mathcal{B}_\infty\|, \\ a_{11} &= c_{2\mathcal{B}}, \\ a_{12} &= c_{\mathcal{B}2}c_{2\mathcal{A}}\bar{l}\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{A} - I_{np}\|, \\ a_{13} &= c_{\mathcal{B}2}c_{2\mathcal{A}}\bar{l}^2\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{B}_\infty\|, \\ a_{14} &= c_{\mathcal{B}2}\bar{l}^2\|I_{np} - \mathcal{B}_\infty\|\|\mathcal{B}_\infty\|, \\ a_{15} &= c_{\mathcal{B}2}\bar{l}\|I_{np} - \mathcal{B}_\infty\|, \\ a_{16} &= c_{\mathcal{B}2}c_{2\mathcal{B}}\bar{l}\|I_{np} - B_\infty\|. \end{aligned}$$

When the largest step-size, $\overline{\alpha}$, satisfies

$$\begin{aligned} 0 < \overline{\alpha} < \min\Bigg\{ &\frac{1}{nl\boldsymbol{\pi}_r^\top\boldsymbol{\pi}_c}, \frac{\delta_3 - \delta_1 a_8}{a_9\delta_1 + a_{10}\delta_2 + a_{11}\delta_4}, \\ &\frac{(1-\sigma_B)\delta_4 - \delta_1 a_{12}}{a_{13}\delta_1 + a_{14}\delta_2 + a_{14}\delta_4}, \frac{(1-\sigma_B)\delta_4 - \delta_1 a_{12}}{a_{13}\delta_1 + a_{14}\delta_2 + a_{14}\delta_4}\Bigg\} \end{aligned} \tag{18}$$

and when the largest momentum parameter, $\overline{\beta}$, satisfies

$$\begin{aligned} 0 \leq \beta < \min\Bigg\{ &\frac{\delta_1(1-\sigma_A) - (a_1\delta_1 + a_2\delta_2 + a_4\delta_4)\,\overline{\alpha}}{a_3\delta_3}, \\ &\frac{\Big(\delta_2\mu[\boldsymbol{\pi}_r]_{\min}[\boldsymbol{\pi}_c]_{\min} - (a_5\delta_1 + a_7\delta_4)\Big)\overline{\alpha}}{a_6\delta_3}, \\ &\frac{\delta_3 - \delta_1 a_8 - (a_9\delta_1 + a_{10}\delta_2 + a_{11}\delta_4)\overline{\alpha}}{\delta_3}, \\ &\frac{(1-\sigma_B)\delta_4 - \delta_1 a_{12} - (a_{13}\delta_1 + a_{14}\delta_2 + a_{14}\delta_4)\overline{\alpha}}{a_{15}\delta_3}\Bigg\}, \end{aligned} \tag{19}$$

where $\delta_1, \delta_2, \delta_3, \delta_4$ are arbitrary constants such that

$$\begin{cases} \delta_1 & < & \max\left\{\frac{\delta_3}{a_8}, \frac{(1-\sigma_B)\delta_4}{a_{12}}\right\}, \\ \delta_2 & > & \frac{a_5\delta_1+a_7\delta_4}{\mu[\boldsymbol{\pi}_r]_{\min}[\boldsymbol{\pi}_c]_{\min}}, \\ \delta_3 & > & 0, \\ \delta_4 & > & 0, \end{cases}$$

then $\rho(J_{\boldsymbol{\alpha},\overline{\beta}}) < 1$ and thus $\|\mathbf{x}_k - \mathbf{1}_n \otimes \mathbf{x}^*\|$ converges to zero linearly at the rate of $\mathcal{O}(\rho(J_{\boldsymbol{\alpha},\overline{\beta}}))^k$.

Proof. It is straightforward to verify Eq. (17) by combining Lemmas 6-9. The next step is to find the range of $\overline{\alpha}$ and $\overline{\beta}$ such that $\rho(J_{\boldsymbol{\alpha},\overline{\beta}}) < 1$. In the light of Lemma 4, we solve for a positive vector $\boldsymbol{\delta} = [\delta_1, \delta_2, \delta_3, \delta_4]^\top$ and the range of $\overline{\alpha}$ and $\overline{\beta}$ such that the following inequality holds:

$$J_{\boldsymbol{\alpha},\overline{\beta}}\boldsymbol{\delta} < \boldsymbol{\delta},$$

which is equivalent to the following four conditions:

$$a_3\delta_3\beta < \delta_1(1-\sigma_A) - (a_1\delta_1 + a_2\delta_2 + a_4\delta_4)\,\overline{\alpha}, \tag{20}$$
$$a_6\delta_3\beta < \delta_2 - \delta_2\lambda - (a_5\delta_1 + a_7\delta_4)\,\overline{\alpha}, \tag{21}$$
$$\delta_3\beta < \delta_3 - \delta_1 a_8 - (a_9\delta_1 + a_{10}\delta_2 + a_{11}\delta_4)\overline{\alpha}, \tag{22}$$
$$a_{15}\delta_3\beta < (1-\sigma_B)\delta_4 - \delta_1 a_{12} - (a_{13}\delta_1 + a_{14}\delta_2 + a_{14}\delta_4)\overline{\alpha}. \tag{23}$$

Recall λ in Lemma 7, when $\overline{\alpha} < \frac{1}{nl\boldsymbol{\pi}_r^\top\boldsymbol{\pi}_c}$, we have

$$\lambda = 1 - \mu n\boldsymbol{\pi}_r^\top \mathrm{diag}(\boldsymbol{\alpha})\boldsymbol{\pi}_c \le 1 - \mu n[\boldsymbol{\pi}_r]_{\min}[\boldsymbol{\pi}_c]_{\min}\overline{\alpha}.$$

Therefore, the third condition in Eq. (21) is satisfied when

$$a_6\delta_3\beta < \delta_2\mu n[\boldsymbol{\pi}_r]_{\min}[\boldsymbol{\pi}_c]_{\min}\overline{\alpha} - (a_5\delta_1 + a_7\delta_4)\,\overline{\alpha}. \tag{24}$$

For the right hand side of the Eq. (20), (24), (22) and (23) to be positive, each one of these equations needs to satisfy the conditions we give below.

$$\text{Eq. (20)}: \ \overline{\alpha} < \frac{\delta_1(1-\sigma_A)}{a_1\delta_1 + a_2\delta_2 + a_4\delta_4}, \tag{25}$$
$$\text{Eq. (24)}: \ \delta_2 > \frac{a_5\delta_1 + a_7\delta_4}{\mu[\boldsymbol{\pi}_r]_{\min}[\boldsymbol{\pi}_c]_{\min}}, \tag{26}$$
$$\text{Eq. (22)}: \begin{cases} \delta_1 < \frac{\delta_3}{a_8}, \\ \overline{\alpha} < \frac{\delta_3 - \delta_1 a_8}{a_9\delta_1 + a_{10}\delta_2 + a_{11}\delta_4}. \end{cases} \tag{27}$$
$$\text{Eq. (23)}: \begin{cases} \delta_1 < \frac{(1-\sigma_B)\delta_4}{a_{12}}, \\ \overline{\alpha} < \frac{(1-\sigma_B)\delta_4 - \delta_1 a_{12}}{a_{13}\delta_1 + a_{14}\delta_2 + a_{14}\delta_4}. \end{cases} \tag{28}$$

We first choose arbitrary positive constants, δ_3 and δ_4, then pick δ_1 satisfying Eqs. (27) and (28), and finally choose δ_2 according to Eq. (26). Note that $\delta_1, \delta_2, \delta_3$, and δ_4 are chosen to ensure that the upper bounds on $\overline{\alpha}$ are all positive. Subsequently, from Eqs. (25), (27), and (28), together with the requirement that $\overline{\alpha} < \frac{1}{nl\boldsymbol{\pi}_r^\top\boldsymbol{\pi}_c}$, we obtain the upper bound on the largest step-size, $\overline{\alpha}$. Finally, the original four conditions in Eqs. (20), (24), (22) and (23) lead to an upper bound on $\overline{\beta}$, and the theorem follows. □

Remark 1: In Theorem 1, we have established the R-linear rate of $\mathcal{AB}m$ when the largest step-size, $\overline{\alpha}$, and the largest momentum parameter, $\overline{\beta}$, respectively follow the upper bounds described in Eq. (18) and Eq. (19). Note that $\delta_1, \delta_2, \delta_3, \delta_4$ therein are tunable parameters and only depend on the network topology and the objective functions. The upper bounds for $\overline{\alpha}$ and $\overline{\beta}$ may not be computable for arbitrary directed graphs as the contraction coefficients, $\sigma_{\mathcal{A}}$, $\sigma_{\mathcal{B}}$, and the norm equivalence constants may be unknown. However, when the graph is undirected, we can obtain computable bounds for $\overline{\alpha}$ and $\overline{\beta}$, as developed in [32, 38] for example. The upper bound on $\overline{\beta}$ also implies that if the step-sizes are relatively large, only small momentum parameters can be picked to ensure stability.

Remark 2: The nonidentical step-sizes in gradient tracking methods [31, 32] have previously been studied in [31, 47–49]. These works rely on some notion of heterogeneity among the step-sizes, defined respectively as the relative deviation of the step-sizes from their average, $\frac{\|(I-W)\boldsymbol{\alpha}\|}{\|W\boldsymbol{\alpha}\|}$, in [31, 48], and as the ratio of the largest to the smallest step-size, $[\boldsymbol{\alpha}]_{\max}/[\boldsymbol{\alpha}]_{\min}$, in [47, 49]. The authors then show that when the heterogeneity is sufficiently small and when the largest step-size follows a bound that is a function of the heterogeneity, the proposed algorithms converge to the global minimizer. It is worth noting that sufficiently small step-sizes do not guarantee sufficiently small heterogeneity in both of the above definitions. In contrast, the upper bound on the largest step-size in this paper, Eq. (18), is independent of any notion of heterogeneity and only depends on the objective functions and the network topology. Each agent therefore locally picks a sufficiently small step-size without any coordination. Based on the discussion in Section III, our approach thus improves the analysis in [31, 47–49]. Besides, Eq. (18) allows the existence of zero step-sizes among the agents as long as the largest step-size is positive and is sufficiently small.

Remark 3: To show that $\mathcal{AB}m$ has an R-linear rate for sufficiently small $\overline{\alpha}$ and $\overline{\beta}$, one can alternatively use matrix perturbation analysis as in [37] (Theorem 1). However, it does not provide explicit upper bounds on $\overline{\alpha}$ and $\overline{\beta}$ in closed form.

V. Average-Consensus from $\mathcal{AB}m$

In this section, we show that $\mathcal{AB}m$ subsumes a novel average-consensus algorithm over strongly-connected directed graphs. To show this, we choose the objective functions as

$$\widetilde{f}_i(\mathbf{x}) = \tfrac{1}{2}\|\mathbf{x} - \boldsymbol{\upsilon}_i\|^2, \quad \forall i.$$

Clearly, the minimization of $\widetilde{F} = \sum_{i=1}^n \widetilde{f}_i$ is now achieved at $\mathbf{x}^* = \frac{1}{n}\sum_{i=1}^n \boldsymbol{\upsilon}_i$. The $\mathcal{AB}m$ algorithm, Eq. (3), thus naturally leads to the following average-consensus algorithm, termed as $\mathcal{AB}m$-$\mathcal{C}$, with $\nabla\mathbf{f}(\mathbf{x}_{k+1}) - \nabla\mathbf{f}(\mathbf{x}_k) = \mathbf{x}_{k+1} - \mathbf{x}_k$; for the sake of simplicity, we choose $\alpha_i = \alpha, \beta_i = \beta, \forall i$:

$$\begin{aligned} \mathbf{x}_{k+1} &= (\mathcal{A} + \beta I)\mathbf{x}_k - \alpha\mathbf{y}_k - \beta\mathbf{x}_{k-1}, \\ \mathbf{y}_{k+1} &= (\mathcal{A} + \beta I - I)\mathbf{x}_k + (\mathcal{B} - \alpha I)\mathbf{y}_k - \beta\mathbf{x}_{k-1}. \end{aligned}$$

Its local implementation at each agent i is given by:

$$\begin{aligned} \mathbf{x}_{k+1}^i &= \sum_{j\in\mathcal{N}_i\setminus i} a_{ij}\mathbf{x}_k^j + (a_{ii}+\beta)\mathbf{x}_k^i - \alpha\mathbf{y}_k^i - \beta\mathbf{x}_{k-1}^i, \\ \mathbf{y}_{k+1}^i &= \sum_{j\in\mathcal{N}_i\setminus i} a_{ij}\mathbf{x}_k^j + (a_{ii}+\beta-1)\mathbf{x}_k^i \\ &\quad + \sum_{j\in\mathcal{N}_i\setminus i} b_{ij}\mathbf{y}_k^j + (b_{ii}-\alpha)\mathbf{y}_k^i - \beta\mathbf{x}_{k-1}^i, \end{aligned}$$

where $\mathbf{x}_0^i = \boldsymbol{\upsilon}_i$ and $\mathbf{y}_i^0 = 0, \ \forall i$.

From the analysis of $\mathcal{AB}m$, an R-linear conv
of $\mathcal{AB}m$-$\mathcal{C}$ to the average of $\boldsymbol{\upsilon}_i$'s is clear from Theor
may be possible to make concrete rate statements by s
the spectral radius of the following system matrix:

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{y}_{k+1} \\ \mathbf{x}_k \end{bmatrix} = \begin{bmatrix} \mathcal{A}+\beta I & -\alpha I & -\beta I \\ \mathcal{A}+\beta I - I & \mathcal{B}-\alpha I & -\beta I \\ I & 0 & 0 \end{bmatrix} \begin{bmatrix} : \\ : \\ \mathbf{x} \end{bmatrix}$$

However, this analysis is beyond the scope of this pa
note that when $\beta = 0$, the above equations still converge to the
average of $\boldsymbol{\upsilon}_i$'s according to Theorem 1. What is surprising is that, with $\beta = 0$, $\mathcal{AB}m$-$\mathcal{C}$ reduces to

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{y}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathcal{A} & -\alpha I \\ \mathcal{A}-I & \mathcal{B}-\alpha I \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ \mathbf{y}_k \end{bmatrix}, \tag{30}$$

which is surplus consensus [18], after a state transformation with diag $(I, -I)$; in fact, any state transformation of the form diag$(I, \widetilde{I})$ applies here as long as $\widetilde{I}$ is diagonal (to respect the graph topology) and invertible. More importantly, compared with surplus consensus [18], $\mathcal{AB}m$-$\mathcal{C}$ uses information from the past iterations. This history information is in fact the momentum from a distributed optimization perspective, which may lead to accelerated convergence as we will numerically show in Section VI.

Following this discussion, choosing the local functions as $\widetilde{f}_i$'s in [31, 32], or in ADD-OPT [33, 34], or in FROST [35, 36], we get average-consensus with only DS, CS, or RS weights. The protocol that results directly from $\mathcal{AB}$ is surplus consensus, while the one resulting directly from FROST was presented in [46]. With the analysis provided in Section III, we see that the algorithm in [46] is in fact related to surplus consensus after a state transformation. Clearly, *accelerated* average-consensus based exclusively on either row- or column-stochastic weights can be abstracted from the discussion herein, after adding a momentum term.

VI. Numerical Experiments

We now provide numerical experiments to illustrate the theoretical findings described in this paper. To this aim, we use two different graphs: an undirected graph, $\mathcal{G}_1$, and a directed graph, $\mathcal{G}_2$. Both graphs have $n = 500$ agents and are generated using nearest neighbor rules and then we add less than 0.05% random links. The number of edges in all cases is less 4% of the total possible edges. Since the graphs are randomly generated across experiments, two sample graphs are shown in Fig. 1, without the self-edges and random links for visual clarity. We generate DS weights using the Laplacian method: $W = I - \frac{1}{\max_i \deg_i + 1} L$, where L is the graph Laplacian and $\deg_i$ is the degree of node i. Additionally, we generate RS and CS weights with the uniform weighting strategy: $a_{ij} = \frac{1}{|\mathcal{N}_j^{\text{in}}|}$ and $b_{ij} = \frac{1}{|\mathcal{N}_j^{\text{out}}|}, \forall i, j$. We note that both weighting strategies are applicable to undirected graphs, while only the uniform strategy can be used over directed graphs.

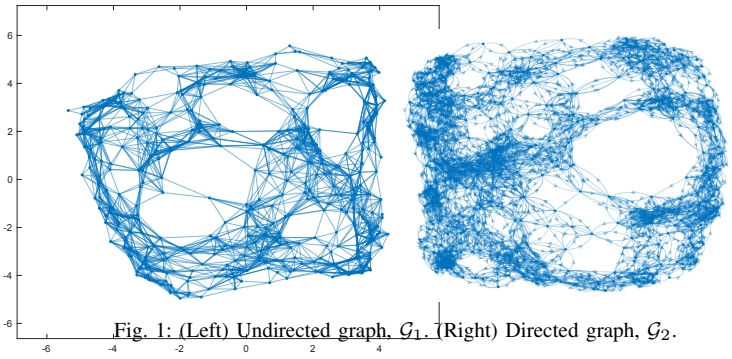

Fig. 1: (Left) Undirected graph, $\mathcal{G}_1$. (Right) Directed graph, $\mathcal{G}_2$.

A. Logistic Regression

We first consider distributed logistic regression: 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 agents cooperatively minimize $F = \sum_{i=1}^n f_i(\mathbf{b}, c)$, to find $\mathbf{b} \in \mathbb{R}^p, c \in \mathbb{R}$, with each private loss function being

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

where $\frac{\lambda}{2}\|\mathbf{b}\|_2^2$ is a regularization term used to prevent overfitting of the data. The feature vectors, $\mathbf{c}_{ij}$'s, are randomly generated from a Gaussian distribution with zero mean and the binary labels are randomly generated from a Bernoulli distribution. We plot the average of residuals at each agent, $\frac{1}{n}\sum_{i=1}^n \|\mathbf{x}_i(k) - \mathbf{x}^*\|_2$, and first compare the performance of the following over undirected graphs in Fig. 2 (Left): (i) $\mathcal{AB}m$ with RS and CS weights; (ii) $\mathcal{AB}m$ with DS weights; (iii) distributed optimization based on gradient tracking from [31, 32, 34], with DS weights; (iv) EXTRA from [26]; and, (v) centralized gradient descent.

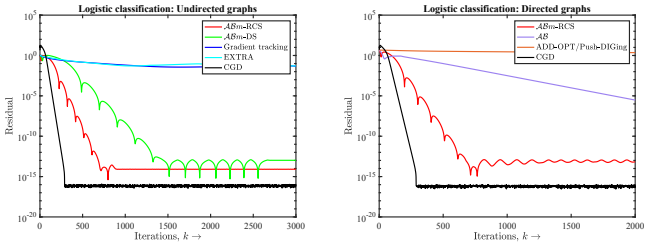

Fig. 2: Logistic regression over undirected (Left) and directed graph (Right).

Next, we compare the performance similarly over *directed graphs* in Fig. 2 (Right). Here, the algorithms with doubly-stochastic weights [26, 31, 32, 34] are not applicable, and instead we compare $\mathcal{AB}m$ with $\mathcal{AB}$ [37], ADD-OPT/Push-DIGing [33, 34], and centralized gradient descent. The weight matrices are chosen as we discussed before and the algorithm parameters are hand-tuned for best performance (except for gradient descent where the optimal step-size is given by $\alpha = \frac{2}{\mu + l}$). We note that momentum improves the convergence when compared to applicable algorithms without momentum, while ADD-OPT/Push-DIGing are much slower because of the eigenvector estimation, see Section III for details.

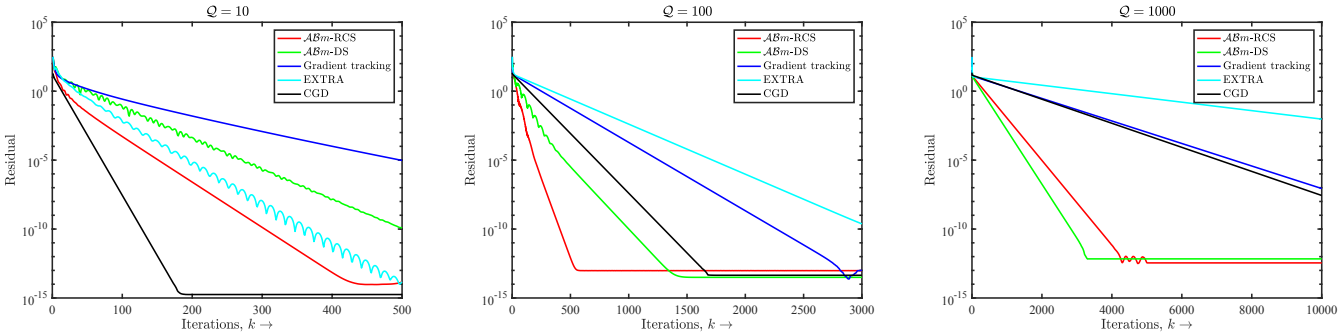

Fig. 3: Performance comparison over *undirected graph*, $\mathcal{G}_1$, as a function of the condition numbers.

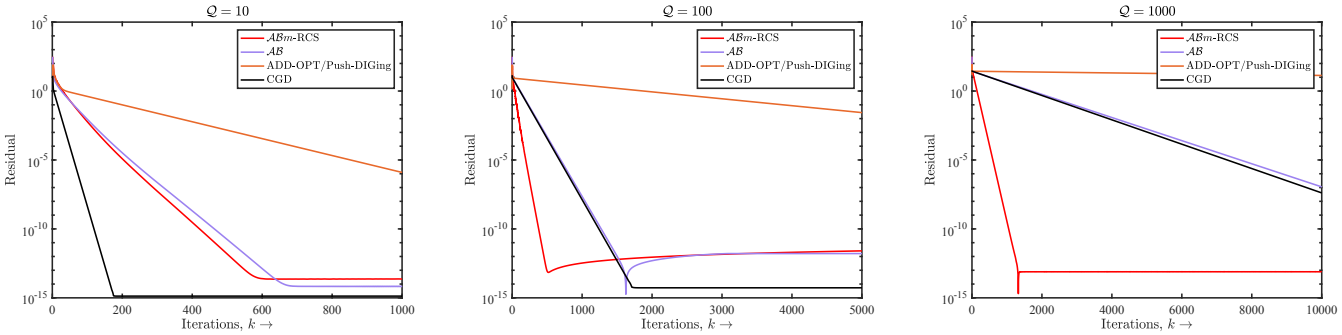

Fig. 4: Performance comparison over *directed graph*, $\mathcal{G}_2$, as a function of the condition numbers.

B. Distributed Quadratic Programming

We now compare the performance of the aforementioned algorithms over different condition numbers of the global objective function, chosen to be quadratic, i.e., $F = \sum_i \mathbf{x}^\top Q_i \mathbf{x} + \mathbf{b}_i^\top \mathbf{x}$, where $Q_i \in \mathbb{R}^{p\times p}$ is diagonal and positive-definite. The condition number $\mathcal{Q}$ of F is given by the ratio of the largest to the smallest eigenvalue of $Q \triangleq \sum_{i=1}^n Q_i$. We first provide the performance comparison over *undirected graphs* in Fig. 3, and then provide the results over *directed graphs* in Fig. 4. In all of these experiments, we have hand-tuned the algorithm parameters for best performance.

For small condition numbers, we note that gradient descent is quite fast and the distributed algorithms suffer from a relatively slower fusion over the graphs. Recall that the optimal convergence rate of gradient decent is $\mathcal{O}((\frac{\mathcal{Q}-1}{\mathcal{Q}+1})^k)$. When the condition number is large, gradient descent is quite conservative allowing fusion to catch up. Finally, we note that $\mathcal{AB}m$, with momentum, outperforms the centralized gradient descent when the condition number is large. This observation is consistent with the existing literature, see e.g., [50, 51, 53–55].

C. $\mathcal{AB}m$ and Average-Consensus

We now provide numerical analysis and simulations to show that $\mathcal{AB}m$-$\mathcal{C}$, in Eq. (29), possibly achieves acceleration when compared with surplus-consensus, in Eq. (30). To explain our choice of α and β, we first note that the power limit of the system matrix in Eq. (30), denoted as $\mathcal{H}$, is [18]:

$$\lim_{k\to\infty} \mathcal{H}^k = \mathcal{H}_\infty = \begin{bmatrix} \mathcal{W}_\infty & -\mathcal{W}_\infty \\ 0_{np\times np} & 0_{np\times np} \end{bmatrix},$$

where $\mathcal{W}_\infty = (\frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top) \otimes I_p$. It is straightforward to show that $\mathcal{H}^k - \mathcal{H}_\infty = (\mathcal{H} - \mathcal{H}_\infty)^k$. Similarly, for the augmented system matrix, $\widetilde{\mathcal{H}}$, in Eq. (29), we observe that

$$\lim_{k\to\infty} \widetilde{\mathcal{H}}^k = \widetilde{\mathcal{H}}_\infty = \begin{bmatrix} \mathcal{W}_\infty & -\mathcal{W}_\infty & 0_{np\times np} \\ 0_{np\times np} & 0_{np\times np} & 0_{np\times np} \\ \mathcal{W}_\infty & -\mathcal{W}_\infty & 0_{np\times np} \end{bmatrix},$$

and it can be verified that $\widetilde{\mathcal{H}}^k - \widetilde{\mathcal{H}}_\infty = (\widetilde{\mathcal{H}} - \widetilde{\mathcal{H}}_\infty)^k$. We therefore use grid search [60] to choose the optimal α^* in $\mathcal{H}$ and the optimal $\widetilde{\alpha}^*$ and $\widetilde{\beta}^*$ in $\widetilde{\mathcal{H}}$, which respectively minimize $\rho(\mathcal{H} - \mathcal{H}_\infty)$ and $\rho(\widetilde{\mathcal{H}} - \widetilde{\mathcal{H}}_\infty)$. Numerically, we observe that it may be possible for the minimum of $\rho(\widetilde{\mathcal{H}} - \widetilde{\mathcal{H}}_\infty)$ to be smaller than that of $\rho\left(\mathcal{H} - \mathcal{H}_\infty\right)$. The convergence speed comparison between $\mathcal{AB}m$-$\mathcal{C}$ and surplus consensus [18] is shown in Fig 5 over a directed graph, $\mathcal{G}_2$.

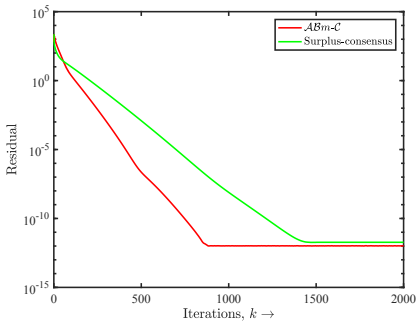

Fig. 5: Average-consensus via $\mathcal{AB}m$-$\mathcal{C}$ (with momentum) and surplus consensus (without momentum) implemented over a directed graph.

VII. Conclusions

In this paper, we provide a framework for distributed optimization that removes the need for doubly-stochastic weights and thus is naturally applicable to both undirected and directed graphs. Using a state transformation based on the non-$\mathbf{1}_n$ eigenvector, we show that the underlying algorithm, $\mathcal{AB}$, based on a simultaneous application of both RS and CS weights, lies at the heart of several algorithms studied earlier that rely on eigenvector estimation when using only CS (or only RS) weights. We then propose the distributed heavy-ball method, termed as $\mathcal{AB}m$, that combines $\mathcal{AB}$ with a heavy-ball (type) momentum term. To the best of our knowledge, this paper is the first to use a momentum term based on the heavy-ball method in distributed optimization. We show that $\mathcal{AB}m$ subsumes a novel average-consensus algorithm as a special case that unifies earlier attempts over directed graphs, with potential acceleration due to the momentum term.

References

[1] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, no. May, pp. 1663–1707, 2010.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundation and Trends in Maching Learning*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[3] H. Raja and W. U. Bajwa, "Cloud k-svd: A collaborative dictionary learning algorithm for big, distributed data," *IEEE Trans. on Signal Processing*, vol. 64, no. 1, pp. 173–188, 2016.

[4] H.-T. Wai, Z. Yang, Z. Wang, and M. Hong, "Multi-agent reinforcement learning via double averaging primal-dual optimization," *arXiv preprint arXiv:1806.00877*, 2018.

[5] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[6] G. Mateos, J. A. Bazerque, and G. B. Giannakis, "Distributed sparse linear regression," *IEEE Trans. on Signal Processing*, vol. 58, no. 10, pp. 5262–5276, Oct. 2010.

[7] J. A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Trans. on Signal Processing*, vol. 58, no. 3, pp. 1847–1862, March 2010.

[8] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *3rd International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, Apr. 2004, pp. 20–27.

[9] S. Safavi, U. A. Khan, S. Kar, and J. M. F. Moura, "Distributed localization: A linear theory," *Proceedings of the IEEE*, 2018.

[10] J. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.

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

[12] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2012.

[13] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *44th Annual IEEE Symposium on Foundations of Computer Science*, Oct. 2003, pp. 482–491.

[14] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization," in *51st IEEE Annual Conference on Decision and Control*, Maui, Hawaii, Dec. 2012, pp. 5453–5458.

[15] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. on Automatic Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015.

[16] C. Xi, Q. Wu, and U. A. Khan, "On the distributed optimization over directed networks," *Neurocomputing*, vol. 267, pp. 508–515, Dec. 2017.

[17] C. Xi and U. A. Khan, "Distributed subgradient projection algorithm over directed graphs," *IEEE Trans. on Automatic Control*, vol. 62, no. 8, pp. 3986–3992, Oct. 2016.

[18] K. Cai and H. Ishii, "Average consensus on general strongly connected digraphs," *Automatica*, vol. 48, no. 11, pp. 2750 – 2761, 2012.

[19] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, Sep. 2016.

[20] A. S. Berahas, R. Bollapragada, N. S. Keskar, and E. Wei, "Balancing communication and computation in distributed optimization," *arXiv preprint arXiv:1709.02999*, 2017.

[21] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *51st IEEE Annual Conference on Decision and Control*, Dec. 2012, pp. 5445–5450.

[22] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Puschel, "D-ADMM: A communication-efficient distributed algorithm for separable optimization," *IEEE Trans. on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, May 2013.

[23] W. Shi, Q. Ling, K Yuan, G Wu, and W Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Trans. on Signal Processing*, vol. 62, no. 7, pp. 1750–1761, April 2014.

[24] A. Mokhtari, W. Shi, Q. Ling, and A. Ribeiro, "A decentralized second-order method with exact linear convergence rate for consensus optimization," *IEEE Trans. on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 507–522, 2016.

[25] D. Jakovetić, J. Xavier, and J. M. F. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, May 2014.

[26] W. Shi, Q. Ling, G. Wu, and W Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[27] C. Xi and U. A. Khan, "DEXTRA: A fast algorithm for optimization over directed graphs," *IEEE Trans. on Automatic Control*, vol. 62, no. 10, pp. 4980–4993, Oct. 2017.

[28] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning—part i: Algorithm development," *arXiv preprint arXiv:1702.05122*, 2017.

[29] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact diffusion for distributed optimization and learning—part ii: Convergence analysis," *arXiv preprint arXiv:1702.05142*, 2017.

[30] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Library in Signal Processing*, vol. 3, pp. 323–453. Elsevier, 2014.

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

[32] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Trans. on Control of Network Systems*, Apr. 2017.

[33] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated distributed directed optimization," *IEEE Trans. on Automatic Control*, Aug. 2017, *in press*.

[34] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal of Optimization*, Dec. 2017.

[35] C. Xi, V. S. Mai, R. Xin, E. Abed, and U. A. Khan, "Linear convergence in optimization over directed graphs with row-stochastic matrices," *IEEE Trans. on Automatic Control*, Jan.

2018, *in press*.
[36] R. Xin, C. Xi, and U. A. Khan, "FROST – Fast row-stochastic optimization with uncoordinated step-sizes," *Arxiv: https://arxiv.org/abs/1803.09169*, Mar. 2018.
[37] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 325–330, Jul. 2018.
[38] G. Qu and N. Li, "Accelerated distributed Nesterov gradient descent," *Arxiv: https://arxiv.org/abs/1705.07176*, May 2017.
[39] D. Jakovetic, "A unification and generalization of exact distributed first order methods," *IEEE Transactions on Signal and Information Processing over Networks*, 2018.
[40] H.-T. Wai, N. M. Freris, A. Nedić, and A. Scaglione, "Sucag: Stochastic unbiased curvature-aided gradient method for distributed optimization," *arXiv preprint arXiv:1803.08198*, 2018.
[41] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.
[42] C. A. Desoer and M. Vidyasagar, *Feedback systems: input-output properties*, vol. 55, Siam, 1975.
[43] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Trans. on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.
[44] Y. Sun, G. Scutari, and D. Palomar, "Distributed nonconvex multiagent optimization over time-varying networks," in *Signals, Systems and Computers, 2016 50th Asilomar Conference on*. IEEE, 2016, pp. 788–794.
[45] S. Pu, W. Shi, J. Xu, and A. Nedić, "A push-pull gradient method for distributed optimization in networks," *arXiv preprint arXiv:1803.07588*, 2018.
[46] A. Priolo, A. Gasparri, E. Montijano, and C. Sagues, "A distributed algorithm for average consensus on strongly connected weighted digraphs," *Automatica*, vol. 50, no. 3, pp. 946–951, 2014.
[47] A. Nedić, A. Olshevsky, W. Shi, and C. A. Uribe, "Geometrically convergent distributed optimization with uncoordinated step-sizes," in *IEEE American Control Conference*, May 2017.
[48] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Convergence of asynchronous distributed gradient methods over stochastic networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 434–448, 2018.
[49] Q. Lü, H. Li, and D. Xia, "Geometrical convergence rate for distributed optimization with time-varying directed graphs and uncoordinated step-sizes," *Information Sciences*, vol. 422, pp. 516–530, 2018.
[50] B. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.
[51] B. Polyak, *Introduction to optimization*, Optimization Software, 1987.
[52] E. Ghadimi, H. R. Feyzmahdavian, and M. Johansson, "Global convergence of the heavy-ball method for convex optimization," in *Control Conference (ECC), 2015 European*. IEEE, 2015, pp. 310–315.
[53] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo, "On the convergence rate of incremental aggregated gradient algorithms," *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 1035–1048, 2017.
[54] L. Lessard, B. Recht, and A. Packard, "Analysis and design of optimization algorithms via integral quadratic constraints," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 57–95, 2016.
[55] Y. Drori and M. Teboulle, "Performance of first-order methods for smooth convex minimization: a novel approach," *Mathematical Programming*, vol. 145, no. 1-2, pp. 451–482, 2014.
[56] B. Polyak and P. Shcherbakov, "Lyapunov functions: An optimization theory perspective," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7456–7461, 2017.
[57] P. Tseng, "An incremental gradient (-projection) method with momentum term and adaptive stepsize rule," *SIAM Journal on Optimization*, vol. 8, no. 2, pp. 506–531, 1998.
[58] N. Loizou and P. Richtárik, "Linearly convergent stochastic heavy ball method for minimizing generalization error," *arXiv preprint arXiv:1710.10737*, 2017.
[59] R. A. Horn and C. R. Johnson, *Matrix Analysis, 2nd ed.*, Cambridge University Press, New York, NY, 2013.
[60] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media, 2013.
[61] D. P. Bertsekas, *Nonlinear programming*, Athena scientific Belmont, 1999.

Ran Xin received his B.S. degree in Mathematics and Applied Mathematics from Xiamen University, China, in 2016, and M.S. degree in Electrical and Computer Engineering from Tufts University in 2018. Currently, he is a Ph.D. student in the Electrical and Computer Engineering department at Tufts University. His research interests include optimization theory and algorithms.

Usman A. 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. 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.