SIAM J. OPTIM.
Vol. 28, No. 2, pp. 1420–1447
© 2018 Society for Industrial and Applied Mathematics

SURPASSING GRADIENT DESCENT PROVABLY: A CYCLIC INCREMENTAL METHOD WITH LINEAR CONVERGENCE RATE*

ARYAN MOKHTARI†, MERT GÜRBÜZBALABAN‡, AND ALEJANDRO RIBEIRO†

Abstract. Recently, there has been growing interest in developing optimization methods for solving large-scale machine learning problems. Most of these boil down to the problem of minimizing an average of a finite set of smooth and strongly convex functions where the number of functions n is large. The gradient descent (GD) method is successful in minimizing convex problems at a fast linear rate; however, it is not applicable to the considered large-scale optimization setting because of the high computational complexity. Incremental methods resolve this drawback of gradient methods by replacing the required gradient for the descent direction with an incremental gradient approximation. They operate by evaluating one gradient per iteration and executing the average of the n available gradients as an approximate gradient. Although incremental methods reduce the computational cost of GD, their convergence rates do not justify their advantage relative to GD in terms of the total number of gradient evaluations until convergence. In this paper, we introduce a double incremental aggregated gradient method (DIAG) that computes the gradient of only one function at each iteration, which is chosen based on a cyclic scheme, and uses the aggregated average gradient of all the functions to approximate the full gradient. The iterates of the proposed DIAG method uses averages of both iterates and gradients in contrast to classic incremental methods that utilize gradient averages but do *not* utilize iterate averages. We prove that not only does the proposed DIAG method converge linearly to the optimal solution, but also its linear convergence factor justifies the advantage of incremental methods over GD. In particular, we prove that the worst-case performance of DIAG is better than the worst-case performance of GD. Numerical experiments on quadratic programming and logistic regression problems showcase the advantage of DIAG relative to GD and other incremental methods.

Key words. incremental methods, finite sum minimization, large-scale optimization, linear convergence rate, worst-case analysis

AMS subject classifications. 90C06, 90C25, 90C30, 90C52

DOI. 10.1137/16M1101702

1. Introduction.

This paper focuses on finite sum optimization where the objective function can be written as the sum of a set of strongly convex functions. In particular, consider $\mathbf{x} \in \mathbb{R}^p$ as the optimization variable and $f_i : \mathbb{R}^p \to \mathbb{R}$ as the ith available function. We aim to find the minimizer of the average function $f(\mathbf{x}) = (1/n)\sum_{i=1}^n f_i(\mathbf{x})$, i.e., we intend to solve the optimization problem

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) := \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^n f_i(\mathbf{x}). \tag{1}$$

In this paper, we refer to f_i as the instantaneous functions and the average function f as the global objective function. This class of optimization problems arises in many

*Received by the editors November 2, 2016; accepted for publication (in revised form) January 29, 2018; published electronically May 8, 2018. This paper expands the results and presents proofs that are referenced in [20].

http://www.siam.org/journals/siopt/28-2/M110170.html

Funding: This work was supported by NSF CAREER CCF-0952867 and ONR N00014-12-1-0997.

†Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104 (aryanm@seas.upenn.edu, aribeiro@seas.upenn.edu).

‡Department of Management Science and Information Systems, Rutgers University, Piscataway, NJ 08901 (mgurbuzbalaban@business.rutgers.edu).

fields such as machine learning [5, 4, 30, 8], optimal control, [6, 7, 18], and wireless communications [26, 27]. Our focus is on problems where the instantaneous functions f_i are smooth and strongly convex.

To explain the contribution of this paper we have to discuss the rate and constants that characterize convergence of the different first-order methods that can be used to solve the problem in (1). To begin with we can neglect the specific form of f and use the conventional gradient descent (GD) method, which is known to converge linearly to the optimal argument [23]. This linear convergence rate comes from using individual iterations that are very costly when the number of functions n is large and motivates the use of stochastic and incremental methods in which only one of the instantaneous gradients ∇f_i is evaluated at each iteration. The selection is random in stochastic methods and cyclic in incremental methods. In either case the idea is that individual iterations are less efficient but since n stochastic or incremental operations have the same cost as one GD iteration, overall convergence is faster.

Although faster convergence is observed in many practical situations, it is not known if it is possible to design a stochastic or incremental method with convergence guarantees that are better than the convergence guarantees of GD. The stochastic gradient descent (SGD) method [28, 4], for instance, is known to have a sublinear convergence rate and is therefore surpassed by regular GD as the number of iterations grows. This limitation is in fact the motivation for alternative stochastic descent methods that achieve linear convergence rates by reducing the variance of stochastic descent directions. Examples of this growing and consequential literature includes stochastic averaging gradient algorithms [17, 9, 10, 21], variance reduction methods [14, 36], dual coordinate methods [31, 32], hybrid algorithms [37, 15], and majorization-minimization algorithms [19]. All of these stochastic methods are successful in achieving a linear convergence rate in expectation with individual iterations that have cost comparable to the cost of SGD iterations. However, the linear convergence constants of these methods are not necessarily better than the linear convergence constant of GD for a problem with comparable condition number. This leaves open the possibility that the worst-case performance of these methods is worse than the worst-case performance of GD—see section 2.

Given that the only difference between stochastic and incremental methods is that in the latter functions are chosen in a *cyclic* order—in contrast to the selection in stochastic methods, which is uniformly random—it is not surprising that analogous statements can be made for incremental gradient descent (IGD) methods [1, 33, 22, 24, 3, 25, 13, 2, 34, 11, 35]. Standard IGD has a slow sublinear convergence rate, which motivates the introduction of memory. This is done in the definition of the incremental aggregated gradient (IAG) method that is shown to achieve linear convergence [11] but with a constant that is not necessarily better that the GD constant. Thus, and as in the case of stochastic methods, it is possible that the worst-case performance of IAG is worse than the worst-case performance of GD—see section 2.

The contribution of this paper is to introduce a first-order incremental method that has a linear convergence rate with a constant that is better than the GD constant of a problem with comparable condition number. This means that the worst-case performance of the proposed algorithm is guaranteed to be no worse than the worst-case performance of GD. The algorithm relies on keeping memory of past variables *and* gradient evaluations and is therefore termed the double incremental aggregated gradient (DIAG) method to emphasize the difference from regular IAG methods in which only gradient histories are maintained. This major difference comes from the fact that DIAG uses a different approximation of the global function f at each iteration

from the one used in IAG. In particular, DIAG approximates each instantaneous function f_i by the sum of its first-order approximation and a proximity term, both evaluated with respect to the same iterate, whereas IAG uses different points for the first-order approximation and the proximity condition. We show that this critical difference leads to an incremental algorithm with a smaller linear convergence factor. Moreover, the linear convergence factor of the proposed DIAG method justifies the use of incremental methods to improve the performance of GD. In particular, we show that the worse case scenario of DIAG is guaranteed to be better than the worse case scenario of GD. Based on our knowledge, this is the first incremental method that is guaranteed to improve the worse case performance of GD.

We start the paper by presenting the GD and IAG methods and studying their convergence guarantees for the case in which the instantaneous functions f_i are strongly convex and their gradients ∇f_i are Lipschitz continuous (section 2). We clarify the reason that the convergence analysis of IAG cannot guarantee the advantage of incremental methods with respect to GD. Then we present the proposed DIAG method, which, in contrast to IAG, which only uses the aggregated gradient average, uses both variable and gradient averages (section 3). We explain the intuition behind this difference by comparing the function approximations used in these methods. Further, we suggest an efficient mechanism to implement the proposed DIAG algorithm that has computational complexity of the order $O(p)$, which is significantly lower than that of GD, given by $\mathcal{O}(np)$ (section 3.1). Additionally, we explain the connection between the proposed DIAG method and the majorization-minimization method (MISO) proposed in [19], and highlight the differences between these two algorithms (Remark 1).

The convergence analysis of the DIAG method is then presented (section 4). We first prove a fundamental lemma that shows that the error of DIAG at each iteration is strictly smaller than the average of the errors of the last n iterations (Lemma 1). We use this result to prove that the sequence of variables generated by DIAG converges to the optimal argument $\mathbf{x}^*$ (Proposition 2), and, in particular, the convergence rate of the iterates evaluated after each pass over the dataset is linear (Corollary 3). This linear convergence factor guarantees that one pass of DIAG is more efficient than one iteration of gradient descent, i.e., the upper bound for the error of DIAG after n gradient evaluations is strictly smaller than the one for GD. Then, we prove that the whole sequence of DIAG iterates is linearly convergent (Theorem 4) and characterize the linear convergence factor (Theorem 7). We extend our convergence results by studying the worst-case asymptotic rate of DIAG (section 5). We use the Perron–Frobenius (PF) theory to show that an upper bound for the sequence of DIAG errors has an asymptotic linear convergence rate that is strictly better than the linear convergence factor of GD (Theorem 9).

We compare the performances of DIAG, GD, and IAG in solving a quadratic programming and a binary classification problem (section 6). Numerical results for the quadratic programming confirm that DIAG outperforms GD. In particular, the relative performance of DIAG and GD does not vary by changing the problem condition number, while IAG is not preferable to GD when the problem condition number is relatively large. Moreover, DIAG outperforms IAG irrespective of the problem parameters. The convergence paths of these methods for the binary classification problem, which is a logistic regression minimization, confirm the observations for the quadratic programming problem. Finally, we close the paper with some concluding remarks (section 7).

1.1. Notation. Vectors are written as $\mathbf{x} \in \mathbb{R}^p$ and matrices as $\mathbf{A} \in \mathbb{R}^{p \times p}$. Given n vectors $\mathbf{x}_i$, the vector $\mathbf{x} = [\mathbf{x}_1; \ldots; \mathbf{x}_n]$ represents a stacking of the elements of each individual $\mathbf{x}_i$. We use $\|\mathbf{x}\|$ and $\|\mathbf{A}\|$ to denote the Euclidean norm of vector $\mathbf{x}$ and matrix $\mathbf{A}$, respectively. Given a function f its gradient $\mathbf{x}$ is denoted by $\nabla f(\mathbf{x})$.

2. Related works and preliminaries. Since the objective function in (1) is convex, descent methods can be used to find the optimal argument $\mathbf{x}^*$. In this paper, we are interested in studying methods that converge to the optimal argument of the global objective function f at a linear rate. It is customary for the linear convergence analysis of first-order methods to assume that the functions are smooth and strongly convex. We formalize these conditions in the following assumption.

Assumption 1. The functions f_i are differentiable and strongly convex with constant $\mu > 0$, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ we can write

$$(\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y}))^T(\mathbf{x} - \mathbf{y}) \geq \mu \|\mathbf{x} - \mathbf{y}\|^2. \tag{2}$$

Moreover, the gradients ∇f_i are Lipschitz continuous with constant $L < \infty$, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$ we have

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L \|\mathbf{x} - \mathbf{y}\|. \tag{3}$$

The strong convexity of the functions f_i with constant μ implies that the global objective function f is also strongly convex with constant μ. Likewise, the Lipschitz continuity of the gradients ∇f_i with constant L yields Lipschitz continuity of the global objective function gradients ∇f with constant L. Note that the conditions in Assumption 1 are mild and hold for most large-scale machine learning applications such as linear regression, logistic regression, least squares, and support vector machines.

The optimization problem in (1) can be solved using the gradient descent (GD) method [23]. The idea of GD is to update the current iterate $\mathbf{x}^k$ by descending through the negative direction of the current gradient $\nabla f(\mathbf{x}^k)$ with a proper stepsize ϵ^k. In other words, the update of GD for solving problem (1) at step k is defined as

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \epsilon^k \nabla f(\mathbf{x}^k) = \mathbf{x}^k - \frac{\epsilon^k}{n} \sum_{i=1}^{n} \nabla f_i(\mathbf{x}^k). \tag{4}$$

Convergence analysis of GD in [23] shows that the sequence of iterates $\mathbf{x}^k$ converges linearly to the optimal argument $\mathbf{x}^*$ if the stepsize is constant and satisfies $\epsilon^k = \epsilon < 2/L$. The fastest convergence rate is achieved by the stepsize $\epsilon = 2/(\mu + L)$, which leads to the linear convergence factor $(\kappa - 1)/(\kappa + 1)$, i.e.,

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{5}$$

where $\kappa := L/\mu$ is the global objective function condition number. Although GD has a fast linear convergence rate, it is not computationally affordable in large-scale applications because of its high computational complexity. To comprehend this limitation, note that each iteration of GD requires n gradient evaluations, which is not computationally affordable in large-scale applications with massive values of n. Stochastic gradient descent (SGD) arises as a natural solution in large-scale settings. SGD modifies the update of GD by approximating the gradient of the global objective function

∇f by the average of a small number of instantaneous gradients chosen uniformly at random from the set of n gradients. To be more precise, the update of SGD at step k is defined as

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\epsilon^k}{b} \sum_{i \in \mathcal{S}_b^k} \nabla f_i(\mathbf{x}^k), \tag{6}$$

where $\mathcal{S}_b^k$ is defined as a random set that contains the indices of b functions that are chosen for the update SGD at step k. Note that the components of the set $\mathcal{S}_b^k$ are chosen uniformly at random from the set of indices $\{1, 2, \ldots, n\}$. Since the stochastic gradient $(1/b) \sum_{i \in \mathcal{S}_b^k} \nabla f_i(\mathbf{x}^k)$ is an unbiased estimator of the gradient $\nabla f(\mathbf{x}^k) = (1/n) \sum_{i=1}^n \nabla f_i(\mathbf{x}^k)$, the sequence of the iterates generated by SGD converges to the optimal argument in expectation. However, the convergence rate is sublinear and slower than the linear convergence of GD. In particular, the expected error $\|\mathbf{x}^k - \mathbf{x}^*\|^2$ of SGD is bounded above as $\mathbb{E}\left[\|\mathbf{x}^k - \mathbf{x}^*\|^2\right] \leq \mathcal{O}(1/k)$ for diminishing stepsizes ϵ^k of the order $1/k$. It is worth mentioning that the expectation is taken with respect to the indices of the chosen random functions up to step k.

One may use a cyclic order instead of stochastic selection of functions in SGD, which leads to the update of the incremental gradient descent method (IGD) as in [3, 34]. Similar to the case for SGD, the sequence of iterates generated by the IGD method converges to the optimal argument at a sublinear rate of the order $\mathcal{O}(1/k)$ when the stepsize is diminishing. SGD and IGD are able to reduce the computational complexity of GD by requiring only one gradient evaluation per iteration; however, they both suffer from slow (sublinear) convergence rates.

The sublinear convergence rate of SGD has been improved recently by the stochastic average gradient method (SAG), which also can be interpreted as a stochastic incremental *aggregated* gradient method. The SAG method updates only one gradient per iteration and uses the average of the most recent version of all gradients—gradients of all functions $f_1, \ldots, f_n$—as an approximation for the full gradient [17]. To be more specific, define $\mathbf{y}_i^k$ as the copy of the decision variable $\mathbf{x}$ for the last time that the function f_i's gradient is updated. In other words, the variable $\mathbf{y}_i^k$ is updated as

$$\mathbf{y}_i^{k+1} = \begin{cases} \mathbf{x}^{k+1} & \text{if } i = i^k, \\ \mathbf{y}_i^k & \text{otherwise,} \end{cases} \tag{7}$$

where i^k is the index of the function chosen at step k. Note that in the SAG method the random index i^k is chosen uniformly at random and the gradient of its corresponding function $\nabla f_{i^k}(\mathbf{x}^k)$ is evaluated and stored as $\nabla f_{i^k}(\mathbf{y}_i^k)$. Then, the update of SAG at step k is given by

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\epsilon}{n} \sum_{i=1}^n \nabla f_i(\mathbf{y}_i^k), \tag{8}$$

which uses the gradients of all the n functions evaluated at different time steps. The sequence of iterates generated by SAG converges linearly to $\mathbf{x}^*$ in expectation with respect to the choices of random indices, i.e.,

$$\mathbb{E}\left[\|\mathbf{x}^k - \mathbf{x}^*\|^2\right] \leq \left(1 - \min\left\{\frac{1}{16\kappa}, \frac{1}{8n}\right\}\right)^k C_0, \tag{9}$$

where C_0 is a constant independent of n and κ [29]. However, the linear convergence constant of SAG in (9) is not necessarily better than the linear convergence constant of GD for a problem with comparable condition number. To be more precise, the residual $\|\mathbf{x}^k - \mathbf{x}^*\|$ of SAG in expectation decays by a factor of $(1 - \min\{\frac{1}{16\kappa}, \frac{1}{8n}\})^{n/2}$ after a pass over the dataset, which might not be better than the upper bound for the residual of GD that decays with a factor of $(\kappa - 1)/(\kappa + 1)$. As an example, for the problem in which $n = 100$ and $\kappa = 10$, the worst-case performance of GD after m passes over the set of functions (m iterations) is bounded above by $((\kappa - 1)/(\kappa + 1))^m \|\mathbf{x}^0 - \mathbf{x}^*\| \approx 0.8181^m \|\mathbf{x}^0 - \mathbf{x}^*\|$, while the worst performance of SAG after m passes over the set of functions (mn iterations) is bounded above by $C_0^{1/2}(1 - \min\{\frac{1}{16\kappa}, \frac{1}{8n}\})^{nm/2} \approx 0.9393^m C_0^{1/2}$. Note that the constants $\|\mathbf{x}^0 - \mathbf{x}^*\|$ and C_0 are negligible for sufficiently large m. For other linearly convergent first-order stochastic methods [9, 10, 14, 36, 31, 32, 37, 15, 19] we can derive similar examples. Beside this issue, the results for all these stochastic first-order methods hold in *expectation*. Thus, there is a positive probability that the sequence of iterates generated by these methods might not converge at a linear rate.

The other alternative for solving the optimization problem in (1) is the incremental aggregated gradient (IAG) method, which is a middle ground between GD and IGD. The IAG method requires one gradient evaluation per iteration, as in IG, while it approximates the gradient of the global objective function $\nabla f(\mathbf{x})$ by the average of the most recent gradient of all instantaneous functions [3], and it has a linear convergence rate, as in GD. In the IAG method, the functions are chosen in a cyclic order and it takes n iterations to have a pass over all the available functions. To introduce the update of IAG, recall the definition of $\mathbf{y}_i^k$ as the copy of the decision variable $\mathbf{x}$ for the last time that the function f_i's gradient is updated before step k, which can be updated as in (7). Then, the update of IAG is given by

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \frac{\epsilon}{n} \sum_{i=1}^{n} \nabla f_i(\mathbf{y}_i^k), \tag{10}$$

which is identical to the update of SAG in (8), and the only difference is in the scheme that the index i^k is chosen.

The convergence results in [34] provide global convergence and local linear convergence of IAG in a more general setting when each component function satisfies a local Lipschitzian error condition. More recently, a new convergence analysis of IAG has been studied in [11] that shows global linear convergence of IAG for strongly convex functions with Lipschitz continuous gradients. In particular, it has been shown that the sequence of iterates $\mathbf{x}^k$ generated by IAG satisfies the following inequality:

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \left(1 - \frac{2}{25n(2n+1)(\kappa+1)^2}\right)^k \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{11}$$

Notice that the convergence rate of IAG is linear and eventually the error of IAG will be smaller than the errors of SGD and IGD, which diminish with a sublinear rate of $\mathcal{O}(1/k)$. To compare the performance of GD and IAG it is fair to compare one iteration of GD with n iterations of IAG. This is reasonable since one iteration of GD requires n gradient evaluations, while IAG uses n gradient evaluations after n iterations. Comparing the decrement factors of GD in (5) and IAG after n gradient evaluations in (11) shows that there is no guarantee that IAG is preferable to GD for all choices of condition number κ and number of functions n, since we could face the

scenario that

$$\left(\frac{\kappa-1}{\kappa+1}\right) < \left(1-\frac{2}{25n(2n+1)(\kappa+1)^2}\right)^n. \tag{12}$$

As an example, for the problem with $n=\kappa=100$, the inequality in (12) holds and the worst-case performance of IAG is worse than the one for GD. Note that the bound for GD in (5) is strict and we can design a sequence that satisfies the equality case of the result in (5).[1] However, the bound in (11) is not necessarily tight and it could be the reason that the comparison in (12) does not justify the use of IAG instead of GD. Our goal in this paper is to come up with a first-order incremental method that has a guaranteed upper bound that is better than the one for GD in (5). We propose this algorithm in the following section.

3. Algorithm definition. In this section, we propose a novel incremental gradient method that unlike other incremental methods is able to improve upon the worst-case performance of GD. To do so, we first introduce a new interpretation of the IAG method. Recall the definition of the variable $\mathbf{y}_i^k$ as the copy of the decision variable $\mathbf{x}$ for the last time that function f_i is chosen for gradient update and its update scheme in (7). The update of IAG in (10) can be interpreted as the solution of the optimization program

$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^p} \left\{ \frac{1}{n}\sum_{i=1}^n f_i(\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^n \frac{1}{2\epsilon}\|\mathbf{x}-\mathbf{x}^k\|^2 \right\}. \tag{13}$$

This interpretation shows that in the update of IAG each instantaneous function $f_i(\mathbf{x})$ is approximated as

$$f_i(\mathbf{x}) \approx f_i(\mathbf{y}_i^k) + \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k) + \frac{1}{2\epsilon}\|\mathbf{x}-\mathbf{x}^k\|^2. \tag{14}$$

Notice that the first two terms $f_i(\mathbf{y}_i^k) + \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k)$ correspond to the first-order approximation of the function f_i around the iterate $\mathbf{y}_i^k$. The last term, which is $1/(2\epsilon)\|\mathbf{x}-\mathbf{x}^k\|^2$, is a proximal term that is added to the first-order approximation. This approximation is different from the classic approximation that is used in first-order methods, since the first-order approximation is evaluated around the point $\mathbf{y}_i^k$, which is different from the iterate $\mathbf{x}^k$ used in the proximal term. This observation verifies that the IAG algorithm performs well when the delayed variables $\mathbf{y}_i^k$ are close to the current iterate $\mathbf{x}^k$, which is true when the stepsize ϵ is very small or the iterates are all close to the optimal solution.

We resolve this issue by introducing a different approach for approximating each component function f_i. In particular, we use the approximation

$$f_i(\mathbf{x}) \approx f_i(\mathbf{y}_i^k) + \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k) + \frac{1}{2\epsilon}\|\mathbf{x}-\mathbf{y}_i^k\|^2. \tag{15}$$

As we observe, the approximation in (15) is more consistent with classic first-order methods compared to the one for IAG in (14). This is true since the first-order approximation and the proximal term in (15) are evaluated with respect to the same

[1] Consider the quadratic programming $f(\mathbf{x}) = (1/2)\mathbf{x}^T\mathbf{A}\mathbf{x}$, where $\mathbf{A} = \text{diag}[\mu, L]$, which has the optimal argument $\mathbf{x}^* = \mathbf{0} \in \mathbb{R}^2$. Then, by setting $\epsilon = 2/(\mu+L)$ the sequence of iterates generated by GD satisfies the relation $\|\mathbf{x}^m - \mathbf{x}^*\| = \rho^m\|\mathbf{x}^0 - \mathbf{x}^*\|$.

point $\mathbf{y}_i^k$. Indeed, the approximation in (15) implies that the global objective function $f(\mathbf{x})$ can be approximated by

$$f(\mathbf{x}) \approx \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2\epsilon}\|\mathbf{x}-\mathbf{y}_i^k\|^2. \tag{16}$$

We can approximate the optimal argument of the global objective function f by minimizing its approximation in (16). Thus, the updated iterate $\mathbf{x}^{k+1}$ can be computed as the minimizer of the approximated global objective function in (16), i.e.,

$$\mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^p} \left\{ \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\mathbf{y}_i^k)^T(\mathbf{x}-\mathbf{y}_i^k) + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2\epsilon}\|\mathbf{x}-\mathbf{y}_i^k\|^2 \right\}. \tag{17}$$

Considering the convex programming in (17) we can derive a closed form expression for the update of $\mathbf{x}^{k+1}$ as

$$\mathbf{x}^{k+1} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{y}_i^k - \frac{\epsilon}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{y}_i^k). \tag{18}$$

We refer to the proposed method with the update in (18) as the double incremental aggregated gradient (DIAG) method. This appellation is justified considering that the update of DIAG requires the *incremented aggregate* of *both* variables and gradients and only uses *gradient* (first-order) information.

Notice that since we use a cyclic scheme, the set of variables $\{\mathbf{y}_1^k, \mathbf{y}_2^k, \ldots, \mathbf{y}_n^k\}$ is equal to the set $\{\mathbf{x}^k, \mathbf{x}^{k-1}, \ldots, \mathbf{x}^{k-n+1}\}$. Therefore, the iterate $\mathbf{x}^{k+1}$ is a function of the last n iterates $\{\mathbf{x}^k, \mathbf{x}^{k-1}, \ldots, \mathbf{x}^{k-n+1}\}$. This observation has a fundamental role in the analysis of the proposed DIAG method—see section 4.

Remark 1. One may consider the proposed DIAG method as a cyclic version of the stochastic Finito and MISO algorithms introduced in [10] and [19], respectively. This is a valid interpretation; however, the convergence analyses and guarantees of these methods are quite different. The proposed DIAG method is designed based on the new interpretation in (15) that leads to a novel proof technique—see Lemma 1—that is different from the analysis of Finito/MISO in [10] and [19]. This analytical difference leads to different convergence guarantees. In particular, the Finito/MISO algorithm cannot improve the performance of GD for all choices of n and κ, while the established theoretical results for DIAG in section 4 guarantee that DIAG outperforms GD under any choices of n and κ.

3.1. Implementation details. Naive implementation of the update in (18) requires computation of sums of n vectors per iteration, which is computationally costly. This unnecessary computation can be avoided by tracking the sums over time. To be more precise, we can define $\mathbf{v}^k$ as the vector that tracks the first sum in (18), which is the sum of the variables. The vector $\mathbf{v}^k$ can be updated as

$$\mathbf{v}^{k+1} = \mathbf{x}^{k+1} - \mathbf{y}_{i^k}^k + \mathbf{v}^k, \tag{19}$$

where i^k is the index of the function chosen at step k. Likewise, we define the vector $\mathbf{g}^k$ as the vector that tracks the sum of gradients in (18), and it can be updated as

$$\mathbf{g}^{k+1} = \nabla f_{i^k}(\mathbf{x}^{k+1}) - \nabla f_{i^k}(\mathbf{y}_{i^k}^k) + \mathbf{g}^k. \tag{20}$$

Algorithm 1 Double incremental aggregated gradient method (DIAG)

1: **Initialization**: $\{\mathbf{y}_i^0\}_{i=1}^{i=n} = \mathbf{x}^0$, $\mathbf{v}^0 = n\mathbf{x}^0$, and $\mathbf{g}^0 = \sum_{i=1}^n \nabla f_i(\mathbf{x}^0)$.
2: **for** $k = 0, 1, \ldots$ **do**
3: Compute the function index $i^k = \text{mod}(k, n) + 1$.
4: Compute $\mathbf{x}^{k+1} = \frac{1}{n}\mathbf{v}^k - \frac{\epsilon}{n}\mathbf{g}^k$.
5: Update sum of variables $\mathbf{v}^{k+1} = \mathbf{x}^{k+1} - \mathbf{y}_{i^k}^k + \mathbf{v}^k$.
6: Compute $\nabla f_{i^k}(\mathbf{x}^{k+1})$ and update $\mathbf{g}^{k+1} = \nabla f_{i^k}(\mathbf{x}^{k+1}) - \nabla f_{i^k}(\mathbf{y}_{i^k}^k) + \mathbf{g}^k$.
7: Replace $\mathbf{y}_{i^k}^k$ and $\nabla f_{i^k}(\mathbf{y}_{i^k}^k)$ by $\mathbf{x}^{k+1}$ and $\nabla f_{i^k}(\mathbf{x}^{k+1})$, respectively. The other elements remain unchanged, i.e., $\mathbf{y}_i^{k+1} = \mathbf{y}_i^k$ and $\nabla f_i(\mathbf{y}_i^{k+1}) = \nabla f_i(\mathbf{y}_i^k)$ for $i \neq i^k$.
8: **end for**

Note that the vectors $\mathbf{v}^k$ and $\mathbf{g}^k$ are initialized as $\mathbf{v}^0 = n\mathbf{x}^0$ and $\mathbf{g}^0 = \sum_{i=1}^n \nabla f_i(\mathbf{x}^0)$.

The proposed double incremental aggregated gradient (DIAG) method is summarized in Algorithm 1. The variables for all the copies of the vector $\mathbf{x}$ are initialized by vector $\mathbf{x}^0$, i.e., $\mathbf{y}_1^0 = \cdots = \mathbf{y}_n^0 = \mathbf{x}^0$, and their corresponding gradients are stored in the memory. At each iteration k, the updated variable $\mathbf{x}^{k+1}$ is computed in step 4 using the update in (18). The sums of variables and gradients are updated in steps 5 and 6, respectively, following the recursions in (19) and (20). In step 7, the old variable $\mathbf{y}_{i^k}^k$ and gradient $\nabla f_{i^k}(\mathbf{y}_{i^k}^k)$ of the updated function f_{i^k} are replaced with their updated versions, i.e., $\mathbf{x}^{k+1}$ and $\nabla f_{i^k}(\mathbf{x}^{k+1})$, and the other components remain unchanged. In step 3, the index i^k is updated in a cycling manner.

Remark 2. Similar to other known incremental methods, e.g., IAG, SAG, SAGA, Finito/MISO, the proposed DIAG method requires a memory of order $\mathcal{O}(np)$, which might not be affordable in some large-scale optimization problems. This issue can be resolved by grouping the functions and creating new sets of functions where each one is the average of a subset of functions. If we combine m functions and use the average of them as the new function, the number of active functions reduces to n/m and the required memory decreases to $\mathcal{O}(np/m)$. On the other hand, this process increases the computational complexity of each iteration from one gradient computation to the calculation of m gradients. Indeed, there is a trade-off between the memory and computational complexity per iteration that can be optimized based on the application of interest.

4. Convergence analysis. In this section, we study the convergence properties of the proposed double incremental aggregated gradient (DIAG) method.

The following lemma characterizes an upper bound for the error $\|\mathbf{x}^{k+1} - \mathbf{x}^*\|$ in terms of the errors of the last n iterations.

LEMMA 1. *Consider the proposed DIAG method in* (18)*. If the conditions in Assumption* 1 *hold, and the stepsize* ϵ *is chosen as* $\epsilon = 2/(\mu + L)$*, the sequence of iterates* $\mathbf{x}^k$ *generated by DIAG satisfies the inequality*

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)\left[\frac{\|\mathbf{x}^k - \mathbf{x}^*\| + \cdots + \|\mathbf{x}^{k-n+1} - \mathbf{x}^*\|}{n}\right], \tag{21}$$

where $\kappa = L/\mu$ *is the objective function condition number.*

Proof. See Appendix A. ∎

The result in Lemma 1 has a significant role in the analysis of DIAG. It shows that the error at step $k+1$ is smaller than the average of the last n errors, where the

decrement factor is the ratio $(\kappa-1)/(\kappa+1)$, which is strictly smaller than 1. The cyclic scheme is critical in proving the result in (21), since it allows one to replace the sum $\sum_{i=1}^{n} \|\mathbf{y}_i^k - \mathbf{x}^*\|$ by the sum of the last n steps' errors $\|\mathbf{x}^k - \mathbf{x}^*\| + \cdots + \|\mathbf{x}^{k-n} - \mathbf{x}^*\|$. Note that If we pick functions uniformly at random, as in MISO, it is not possible to write the expression in (21), even in expectation. We also cannot write an inequality similar to the one in (21) for the IAG method, although it uses a cyclic scheme. This contrast originates from the fact that IAG only uses gradient averages, whereas DIAG uses both variable and gradient averages. In the following theorem, we use the result in Lemma 1 to show that the sequence of variables $\mathbf{x}^k$ converges to the optimal argument $\mathbf{x}^*$.

PROPOSITION 2. *Consider the proposed DIAG method in* (18) *and recall the definition* $\rho := (\kappa - 1)/(\kappa + 1)$*, where* $\kappa = L/\mu$ *is the problem condition number. If the conditions in Assumption* 1 *hold, and the stepsize* ϵ *is chosen as* $\epsilon = 2/(\mu + L)$*, then the residual* $\|\mathbf{x}^k - \mathbf{x}^*\|$ *of DIAG for iterations* $k = 1, \ldots, n$ *satisfies the inequality*

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho \left[1 - \frac{(k-1)(1-\rho)}{n}\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k = 1, \ldots, n, \tag{22}$$

and for the steps $k > n$ *we have*

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho^{\lfloor \frac{k-1}{n} \rfloor + 1} \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k > n, \tag{23}$$

where $\lfloor a \rfloor$ *indicates the floor of* a*.*

Proof. See Appendix B. ∎

The first outcome of the result in Proposition 2 is the convergence of the sequence $\|\mathbf{x}^k - \mathbf{x}^*\|$ to zero as k approaches infinity. The second result, which we formalize in the following corollary, shows that the sequence of errors converges linearly after each pass over the dataset.

COROLLARY 3. *If the conditions in Proposition* 2 *are satisfied, the error of the proposed DIAG method after* $m > 1$ *passes over the set of functions* f_i*, which requires* mn *gradient evaluations and corresponds to the iterate* $k = n(m-1) + 1$*, is bounded above by*

$$\|\mathbf{x}^{n(m-1)+1} - \mathbf{x}^*\| \leq \rho^m \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{24}$$

Proof. Since we count the initial n gradient computations, the iterate $\mathbf{x}^1$ requires n gradient computations. After the first iteration each step requires only one gradient computation. Therefore, the total number of gradient computations to evaluate $\mathbf{x}^k$ is $n+k-1$. Conversely, the variable for which exactly mn gradients have been evaluated is $\mathbf{x}^{n(m-1)+1}$. Thus, by setting $k = n(m-1)+1$ in (23) we obtain the residual of DIAG after mn passes over the set of functions and the claim in (24) follows. ∎

The result in Corollary 3 shows that the subsequence of the last iterates of each pass is linearly convergent. Moreover, the result in Corollary 3 verifies the advantage of the DIAG method versus the full gradient descent (GD) method. In particular, it shows that the error of DIAG after $m > 1$ passes over the set of functions f_i corresponding to the iterate $k = n(m-1)+1$ is bounded above by $\rho^m[1-((1-\rho)/n)\times \min\{1, ((n-1)/2)\}]\|\mathbf{x}^0 - \mathbf{x}^*\|$, which is strictly smaller than the upper bound for the

error of GD after m iterations (nm gradient computations) given by $\rho^m\|\mathbf{x}^0-\mathbf{x}^*\|$. Therefore, the DIAG method outperforms GD for any choice of κ and $n>1$; DIAG and GD are identical for $n=1$.

Notice that after the first pass over the set of functions—iteration $k=1$ for the DIAG method—the error of DIAG is upper bounded by $\rho\|\mathbf{x}^0-\mathbf{x}^*\|$ based on the result in (22). This bound is identical to the result for GD after one pass over the set of functions, since the first iterations of GD and DIAG are identical.

Although the result in Corollary 3 implies that the DIAG method is preferable with respect to GD and shows linear convergence of a subsequence of iterates, it is not sufficient to prove linear convergence of the whole sequence of iterates generated by DIAG. To be more precise, the result in Corollary 3 shows that the subsequence of errors $\{\|\mathbf{x}^{kn}-\mathbf{x}^*\|\}_{k=0}^{\infty}$, which are associated with the variables at the end of each pass over the set of functions, is linearly convergent. However, we aim to show that the whole sequence $\{\|\mathbf{x}^{k}-\mathbf{x}^*\|\}_{k=0}^{\infty}$ is linearly convergent. To be more precise, our goal is to prove that the sequence of DIAG iterates satisfies $\|\mathbf{x}^k-\mathbf{x}^*\|\leq a\gamma^k\|\mathbf{x}^0-\mathbf{x}^*\|$ for a constant $a>0$ and a positive coefficient $0\leq\gamma<1$. In the following theorem, we show that this condition is satisfied for the DIAG method.

THEOREM 4. *Consider the introduced DIAG method in* (18)*. If the conditions in Assumption* 1 *hold, and the stepsize* ϵ *is chosen as* $\epsilon=2/(\mu+L)$*, for* $k\geq 1$ *we can write*

$$\|\mathbf{x}^k-\mathbf{x}^*\|\leq a\gamma^k\|\mathbf{x}^0-\mathbf{x}^*\| \tag{25}$$

if the constants $a>0$ *and* $0\leq\gamma<1$ *satisfy the following conditions*

$$\rho\left(1-\frac{(k-1)(1-\rho)}{n}\right)\leq a\gamma^k \quad \textit{for } k=1,\dots,n, \tag{26}$$

$$\gamma^{n+1}-\left(1+\frac{\rho}{n}\right)\gamma^n+\frac{\rho}{n}\leq 0 \quad \textit{for } k>n. \tag{27}$$

Proof. See Appendix C. ∎

The result in Theorem 4 provides conditions on the constants a and γ such that the linear convergence inequality $\|\mathbf{x}^k-\mathbf{x}^*\|\leq a\gamma^k\|\mathbf{x}^0-\mathbf{x}^*\|$ holds. However, it does not gurantee that the set of constants $\{a,\gamma\}$ that satisfy the required conditions in (26) and (27) is nonempty. In the following proposition we show that there exist constants a and γ satisfying these conditions.

PROPOSITION 5. *There exist constants* $a>0$ *and* $0<\gamma<1$ *that satisfy the inequalities in* (26) *and* (27)*. In other words, the set of feasible solutions for the system of inequalities in* (26) *and* (27) *is nonempty.*

Proof. See Appendix D. ∎

The result in Proposition 5 in conjunction with the result in Theorem 4 guarantees linear convergence of the iterates generated by the DIAG method. Although there are different pairs of $\{a,\gamma\}$ that satisfy the conditions in (26) and (27) and lead to the linear convergence result in (25), we are interested in finding the pair $\{a,\gamma\}$ that leads to the smallest linear convergence factor γ, i.e., the pair that guarantees the fastest linear convergence rate. To find the smallest γ, we should pick the smallest γ that satisfies the inequality $\gamma^{n+1}-(1+\rho/n)\gamma^n+\rho/n\leq 0$. Then choose the smallest constant a that satisfies the conditions in (26) for the given γ. To do so, we first look at the properties of the function $h(\gamma):=\gamma^{n+1}-(1+\rho/n)\gamma^n+\rho/n$ in the following lemma.

LEMMA 6. *Consider the function* $h(\gamma) := \gamma^{n+1} - (1+\rho/n)\,\gamma^n + \rho/n$ *for* $\gamma \in [0,1)$. *The function* h *has only one root* γ_0 *in the interval* $[0,1)$. *Moreover,* γ_0 *is the smallest choice of* γ *that satisfies the condition in* (27) .

Proof. The derivative of the function h is given by

$$\frac{d}{d\gamma}h = (n+1)\gamma^n - (n+\rho)\gamma^{n-1}. \tag{28}$$

Therefore, the only critical point of the function h in the interval $(0,1)$ is $\gamma^* = (n+\rho)/(n+1)$. The point γ^* is a local minimum for the function h, since the second derivative of the function h is positive at γ^*. Notice that the objective function value $h(\gamma^*) < 0$ is negative. Moreover, we know that $h(0) > 0$ and $h(1) = 0$. This observation shows that the function h has a root γ_0 between 0 and γ^* and this is the only root of the function h in the interval $(0,1)$. Thus, γ_0 is the smallest value of γ in the interval $(0,1)$ that satisfies the condition in (27). ∎

The result in Lemma 6 shows that the unique root of the function $h(\gamma) := \gamma^{n+1} - (1+\rho/n)\gamma^n + \rho/n$ in the interval $[0,1)$ is the smallest γ that satisfies the condition in (27). We use this result to formalize the pair $\{a,\gamma\}$ with the smallest choice of γ that satisfies the conditions in (26) and (27).

THEOREM 7. *Consider the DIAG method in* (18). *Let the conditions in Assumption* 1 *hold and set the stepsize as* $\epsilon = 2/(\mu+L)$. *Then, the sequence of iterates generated by DIAG is linearly convergent as*

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq a_0 \gamma_0^k \|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{29}$$

where γ_0 *is the unique root of the equation*

$$\gamma^{n+1} - \left(1 + \frac{\rho}{n}\right)\gamma^n + \frac{\rho}{n} = 0 \tag{30}$$

in the interval $[0,1)$ *and* a_0 *is given by*

$$a_0 = \max_{i\in\{1,\ldots,n\}} \rho \left(1 - \frac{(i-1)(1-\rho)}{n}\right)\gamma_0^{-i}. \tag{31}$$

Proof. This follows from the results in Theorem 4 and Lemma 6. ∎

The result in Theorem 7 shows R-linear convergence of the DIAG iterates with the linear convergence factor γ_0; however, it does not show that γ_0^n is smaller than the linear convergence factor of GD. In the following section, we aim to show that the linear convergence factor of DIAG after n iterations, which is γ_0^n, is strictly smaller than the linear factor of GD.

5. Worst-case asymptotic rate of DIAG. In the previous section, we proved that the DIAG method outperforms GD *after each pass* (Corollary 3), but this result does not characterize the linear convergence factor for the sequence of errors $\|\mathbf{x}^k - \mathbf{x}^*\|$ generated by DIAG. The result in Theorem 7 shows R-Linear convergence of the sequence $\|\mathbf{x}^k - \mathbf{x}^*\|$ to zero; however, it does not show that γ_0^n is smaller than the linear convergence factor of GD. In this section, we aim to derive a result that shows the sequence $\|\mathbf{x}^k - \mathbf{x}^*\|$ has a linear convergence rate with constant γ_0 such that γ_0^n is strictly smaller than ρ, which is the linear convergence factor of GD. To do so, we define the sequence d^k as

$$d^{k+1} = \rho\,\frac{d^k + d^{k-1} + \cdots + d^{k-n+1}}{n}, \tag{32}$$

where $\rho = (\kappa - 1)/(\kappa + 1)$ and $d^j := \|\mathbf{x}^j - \mathbf{x}^*\|$ for $j = 0, 1, 2, \ldots, n-1$. It follows directly from (21) that the sequence d^k provides an upper bound for the sequence of errors $\|\mathbf{x}^k - \mathbf{x}^*\|$ for all $k \geq 0$. In other words, we have the relation

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq d^k \quad \text{for all } k \geq 0. \tag{33}$$

Next, we characterize the convergence properties of the sequence of d^k, which provides an upper bound for the desired error sequence $\|\mathbf{x}^k - \mathbf{x}^*\|$. To do so, we rewrite the update of the sequence d^k in matrix form, which is more suitable for the analysis. Define the column vector $\mathbf{d}^k = [d^{k-1}; \ldots; d^{k-n}] \in \mathbb{R}^n$ as the concatenation of the last n values of the sequence d^k up to step k. Considering the definition of $\mathbf{d}^k$ and the update of the sequence d^k we can write

$$\mathbf{d}^{k+1} = \mathbf{M}_\rho \mathbf{d}^k, \quad \text{where} \quad \mathbf{M}_\rho := \begin{bmatrix} \frac{\rho}{n} & \frac{\rho}{n} & \cdots & \frac{\rho}{n} \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{34}$$

We observe that the matrix $\mathbf{M}_\rho \in \mathbb{R}^{n \times n}$ is a nonnegative matrix whose eigenvalues determine the asymptotic growth rate of the sequence $\mathbf{d}^k$ and hence of d^k. It is straightforward to check that the characteristic polynomial of $\mathbf{M}_\rho$ is

$$T(\lambda) = \lambda^n - \frac{\rho}{n}\lambda^{n-1} - \frac{\rho}{n}\lambda^{n-2} - \cdots - \frac{\rho}{n} = \frac{\lambda^{n+1} - \left(1 + \frac{\rho}{n}\right)\lambda^n + \frac{\rho}{n}}{\lambda - 1}, \tag{35}$$

whose roots are the eigenvalues of the matrix $\mathbf{M}_\rho$. In the remainder of this section, we will infer information about the eigenvalues of $\mathbf{M}_\rho$ using Perron–Frobenius (PF) theory. This theory is well developed for positive matrices where all the entries are strictly positive but $\mathbf{M}_\rho$ has zero entries and is therefore not positive. Nevertheless, PF theory has been successfully extended to certain nonnegative matrices called *irreducible* matrices. A square matrix $\mathbf{A}$ is called irreducible if for every i and j there exists an r such that $\mathbf{A}^r(i,j) > 0$. In the next lemma, we prove that the matrix $\mathbf{M}_\rho$ is irreducible, which will justify our use of the PF theory developed for irreducible matrices.

LEMMA 8. *The matrix $\mathbf{M}_\rho$ is irreducible for any $\rho > 0$.*

Proof. By the definition of irreducibility, we need to show that for every i and j there exists an r such that the rth power of the matrix $\mathbf{M}_\rho$ is entrywise positive, i.e., $\mathbf{M}_\rho^r(i,j) > 0$. Let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$ be the standard basis for $\mathbb{R}^n$. It suffices to show that we can choose $r = n$ for all i and j, i.e.,

$$\mathbf{M}_\rho^n(i,j) = \mathbf{e}_i^T \mathbf{M}_\rho^n \mathbf{e}_j > 0 \quad \text{for all } i, j. \tag{36}$$

By the definition of the recurrence (34), we have

$$\begin{bmatrix} d^{2n-1} \\ d^{2n-2} \\ \cdots \\ d^n \end{bmatrix} = \mathbf{d}^{2n} = (\mathbf{M}_\rho)^n \mathbf{d}^n = (\mathbf{M}_\rho)^n \begin{bmatrix} d^{n-1} \\ d^{n-2} \\ \cdots \\ d^0 \end{bmatrix}.$$

Fix any $j \in \{1, \ldots, n\}$ and choose $\mathbf{d}^n = \mathbf{e}_j$ (this would correspond to the initalization $d_{n-\ell} = 1$ for $\ell = j$ and $d_{n-\ell} = 0$ for $1 \leq \ell \leq n$ and $\ell \neq j$). Then, using the definition

of $\mathbf{M}_\sigma$, it is easy to check that such an initialization of $\mathbf{d}^n$ leads to $d^n = \rho/n > 0$, $d^{n+1} > 0, \ldots, d^{2n-1} > 0$. Therefore, for every i and j, we have

$$(\mathbf{M}_\rho)^n(i,j) = \mathbf{e}_i^T(\mathbf{M}_\rho)^n\mathbf{e}_j = d^{2n-i} > 0, \tag{37}$$

which proves (36) and completes the proof. ∎

The following result shows that the sequence d^k converges to zero linearly with a constant γ_0, where γ_0 is defined by (30). We also derive upper and lower bounds on γ_0.

THEOREM 9. *Consider the constant $\rho = (\kappa-1)/(\kappa+1) \in (0,1)$ and let $\lambda^*(\rho)$ be the spectral radius of the matrix $\mathbf{M}_\rho$. Then the following hold.*

(i) *$\lambda^*(\rho)$ is the largest real root of the characteristic polynomial $T(\lambda)$. Furthermore, it is a simple root.*

(ii) *We have the limit*

$$\lim_{k\to\infty} d^{k+1}/d^k = \lambda^*(\rho). \tag{38}$$

(iii) *For integer numbers $n > 1$ the constant $\lambda^*(\rho)$ is bounded below and above as*

$$\rho \le \lambda^*(\rho) < \sqrt[n]{\rho}. \tag{39}$$

(iv) *We have $\lambda^*(\rho) = \gamma_0$, where γ_0 is the largest real root of the polynomial $h(\lambda) := \lambda^{n+1} - (1+(\rho/n))\lambda^n + (\rho/n)$ in the interval $[0,1)$.*

Proof. See Appendix E. ∎

The results in Theorem 9 study the convergence properties of the sequence d^k defined in (32), which is an upper bound for the sequence of DIAG errors $\|\mathbf{x}^k - \mathbf{x}^*\|$. The last result in Theorem 9 shows that the constant $\lambda^*(\rho)$, which is the spectral radius of the matrix $\mathbf{M}_\rho$, is equal to the linear convergence factor γ_0 of DIAG defined as the root of the polynomial in (30). The second result indicates that the sequence d^k has an asymptotic linear convergence rate with the constant $\lambda^*(\rho) = \gamma_0$. This convergence result is not stronger than the result in Theorem 7, since it holds asymptotically, but we report this result since it shows that there exists a sequence that achieves the theoretical upper bound proven for the DIAG method. Also, this result is interesting since it proves the R-linear convergence of DIAG from an entirely different approach based on PF theory. The most important result in Theorem 9 is the third result, which shows that $\lambda^*(\rho)$, which is equal to γ_0 according to the last result, is strictly smaller than $\sqrt[n]{\rho}$. Based on the inequality in (39), we obtain that the linear convergence factor of DIAG after running for n iterations, which is γ_0^n, is strictly smaller than ρ, the decrement factor of GD after one pass over the set of functions.

It is worth mentioning that the upper bound sequence d_k achieving the asymptotic Q-linear convergence with ratio γ_0 doesn't necessarily mean the sequence $\|\mathbf{x}^k - \mathbf{x}^*\|$ cannot achieve a better rate, but it implies the rate cannot be worse than γ_0. Therefore, we refer to this result as the asymptotic *worst-case scenario* analysis of DIAG.

Remark 3. Note that the right-hand inequality in (39) can also be achieved using the results in section 4. To be more specific, first one may show $\rho = (1-\frac{1-\rho}{1}) < (1-\frac{1-\rho}{2})^2 < \cdots < (1-\frac{1-\rho}{n})^n$. This sequence of inequalities implies that $\rho < (1-\frac{1-\rho}{n})^n$, which is equivalent to the inequality $\rho^{\frac{n+1}{n}} - (1+\frac{\rho}{n})\rho + \frac{\rho}{n} < 0$, and, therefore, we obtain that $h(\rho^{1/n}) < 0$. Further, according to the result in Lemma 6, $\lambda^*(\rho) = \gamma_0$ is

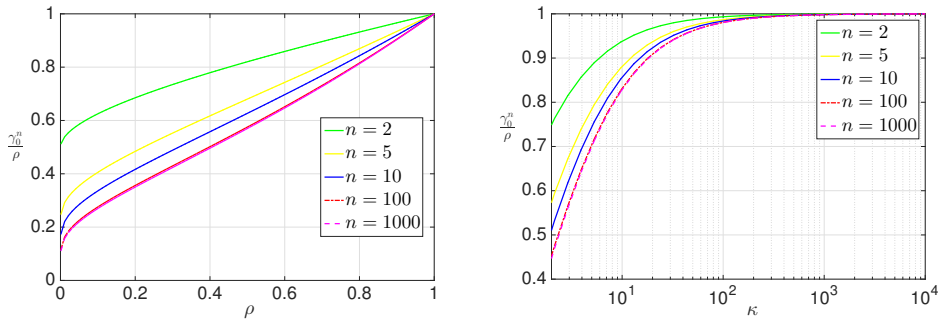

FIG. 1. *Comparison of the linear convergence factors of DIAG and GD via the ratio γ_0^n/ρ in terms of ρ (left) and κ (right) for different choices of n.*

the unique solution to $h(\lambda) = 0$, i.e., $h(\lambda^*(\rho)) = 0$, and $\lambda^*(\rho) < \lambda$ for all $\lambda \in [0, 1)$ with $h(\lambda) < 0$. Combining these two results leads to the conclusion that $\lambda^*(\rho) < \rho^{1/n}$.

Indeed, formalizing the gap between the linear convergence factors of DIAG and GD requires access to an explicit expression for the largest root of the polynomial in (35). However, for specific choices of n and κ one can evaluate the DIAG linear convergence factor γ_0 using polynomial solvers and compare it with the linear factor of GD. We, therefore, compare the ratio γ_0^n/ρ for some choices of ρ and n by finding the root of the polynomial using a MATLAB solver. The outcome of the comparison is illustrated in Figure 1 (see color figures in the online version). As we observe in the left-hand plot in Figure 1, for the case in which $n = 2$, the variable γ_0^n/ρ, which is the ratio between the linear convergence factors of DIAG and GD after one pass over the functions, is close to 0.5 for small choices of ρ, while it approaches 1 as ρ becomes closer to 1. Therefore, for smaller choices of ρ the gain in using DIAG instead of GD is more significant compared to the cases in which ρ is close to 1. A similar pattern can be observed for other choices of n. Conversely, for a fixed choice of ρ, when the number of functions n increases the ratio γ_0^n/ρ becomes smaller. This behavior shows that by increasing the number of functions n the gap between the performances of DIAG and GD increases and DIAG becomes more favorable. Since $\rho = (k-1)/(k+1)$ is an increasing function of the problem condition number κ, similar conclusions can be achieved by comparing the ratio γ_0^n/ρ for different choices of n and κ, as demonstrated in the right-hand plot in Figure 1. It is also worth mentioning that in all the illustrated curves the ratio γ_0^n/ρ is smaller than 1, which verifies our theoretical conclusion that DIAG outperforms GD for all choices of n and κ.

6. Numerical experiments. In this section, we study the performance of the proposed DIAG method and compare it with existing alternative first-order methods. To do so, we first apply DIAG to solve a family of quadratic programming problems. Then, we evaluate the performance of DIAG and other first-order methods in solving a logistic regression minimization problem.

6.1. Quadratic programming example. To study the effect of the number of functions n and problem condition number κ on the performance of the GD, IAG, and DIAG methods, we first apply these algorithms in solving a quadratic programming problem, where we can tune the problem condition number. In particular, consider

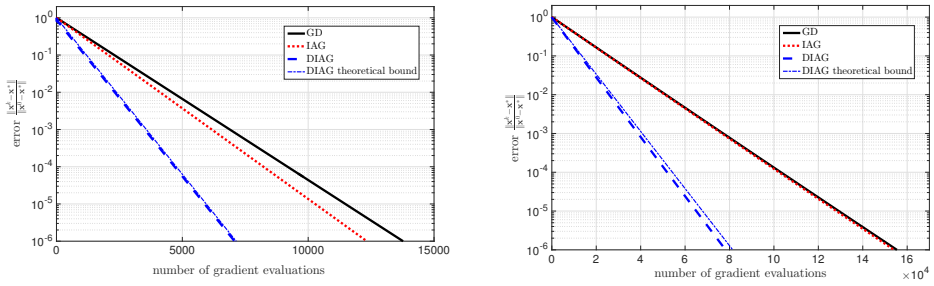

FIG. 2. *Relative error of GD, IAG, and DIAG versus number of gradient evaluations for the quadratic programming in* (40) *with* $n = 200$ *and* $\kappa = 10$ *(left) and* $n = 200$ *and* $\kappa = 117$ *(right). When the condition number is small, the IAG method performs slightly better than GD, while DIAG has the fastest convergence path. For the case in which the condition number is larger, IAG and GD have similar convergence paths, and the best performance belongs to DIAG.*

the optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\mathbf{x}^T\mathbf{A}_i\mathbf{x} + \mathbf{b}_i^T\mathbf{x}, \tag{40}$$

where each matrix $\mathbf{A}_i \in \mathbb{R}^{p\times p}$ is a positive definite diagonal matrix and each vector $\mathbf{b}_i \in \mathbb{R}^p$ is randomly chosen from the box $[0,1]^p$. To control the problem condition number, the first $p/2$ diagonal elements of $\mathbf{A}_i$ are chosen uniformly at random from the interval $[1, 10^1, \ldots, 10^{\eta/2}]$ and its last $p/2$ elements are chosen from the interval $[1, 10^{-1}, \ldots, 10^{-\eta/2}]$. This selection results in the sum matrix $\sum_{i=1}^n \mathbf{A}_i$ having eigenvalues in the range $[n10^{-\eta/2}, n10^{\eta/2}]$. In our experiments, we fix the variable dimension as $p = 20$ and the number of functions as $n = 200$. Moreover, the stepsizes of GD and DIAG are set as their best theoretical stepsizes, which are $\epsilon_{GD} = 2/(\mu+L)$ and $\epsilon_{DIAG} = 2/(\mu+L)$, respectively. Note that the stepsize suggested in [11] for IAG is $\epsilon_{IAG} = (0.32\mu)/((nL)(L+\mu))$; however, this choice of stepsize leads to slow convergence of IAG in practice. Thus, we use the stepsize $\epsilon_{IAG} = 2/(nL)$, which performs better than the one suggested in [11].

We compare these methods in terms of the total number of gradient evaluations. Note that comparing these methods in terms of the total number of iterations would not be fair since each iteration of GD requires n gradient evaluations, while IAG and DIAG only require one gradient computation per iteration.

We first consider the case in which $\eta = 1$ and use the realization with condition number $\kappa = 10$ so that we have a relatively small condition number. The left-hand plot in Figure 2 (see color figures in onlines version) demonstrates convergence paths of the normalized error $\|\mathbf{x}^k - \mathbf{x}^*\|/\|\mathbf{x}^0 - \mathbf{x}^*\|$ for IAG, DIAG, and GD when $n = 200$ and $\kappa = 10$. As we observe, IAG performs slightly better than GD, while the best performance belongs to DIAG. To be more precise, DIAG requires $7,069$ gradient evaluations (approximately 35 passes over the dataset) to achieve the relative error of $\|\mathbf{x}^k - \mathbf{x}^*\|/\|\mathbf{x}^0 - \mathbf{x}^*\| = 10^{-6}$, while IAG requires $12,330$ gradient evaluations (approximately 61 passes over the dataset) to achieve the same accuracy. The GD method has the worst performance and achieves the relative error $\|\mathbf{x}^k - \mathbf{x}^*\|/\|\mathbf{x}^0 - \mathbf{x}^*\| = 10^{-6}$ after 68 iterations, which is equivalent to $13,600$ gradient evaluations. We have also illustrated the theoretical bound for the DIAG method in Figure 2, which is computed by finding

the root of the polynomial in (30) for $n = 200$ and $\rho = (\kappa-1)/(\kappa+1) = 9/11$. In this case, the root of the polynomial is $\gamma_0 = 0.998067$ and the DIAG theoretical bound curve corresponds to the sequence $0.998067(\|\mathbf{x}^k - \mathbf{x}^*\|/\|\mathbf{x}^0 - \mathbf{x}^*\|)$. As we observe, the performance of the DIAG method is almost identical to its proven theoretical bound, which shows the tightness of the bound for DIAG.

Comparison of the convergence guarantees for GD, IAG, and DIAG shows that the IAG method is more sensitive to the problem condition number since the linear convergence factor of IAG is of the order $1 - \mathcal{O}(1/\kappa^2)$, while the linear convergence factor of GD and DIAG are of the order of $1-\mathcal{O}(1/\kappa)$. To study the effect of problem condition number in practice, we increase the constant η to have a poorly conditioned problem. In particular, we increase the problem condition number by setting $\eta = 2$ and using the realization with condition number $\kappa = 117$. The right-hand plot in Figure 2 illustrates the performance of these methods for the case in which $n = 200$ and $\kappa = 117$. We observe that the convergence path of IAG is almost identical to the one for GD. This observation verifies that the performance of IAG worsens more significantly by increasing the problem condition number. Interestingly, the relative performance of DIAG and GD does not change by increasing the problem condition number. To be more specific, for the case in which $n = 200$ and $\kappa = 117$, GD and IAG reach the relative error $\|\mathbf{x}^k - \mathbf{x}^*\|/\|\mathbf{x}^0 - \mathbf{x}^*\| = 10^{-6}$ after 1.54×10^5 gradient evaluations, while DIAG requires only 7.8×10^4 gradient computations to achieve the same accuracy. As in the previous case, we also compare the performance of DIAG with its proven theoretical bound. To do so, we find the root of the polynomial in (30) for $n = 200$ and $\rho = (\kappa - 1)/(\kappa + 1) = 116/118$, which is $\gamma_0 = 0.99983$. As we observe, the convergence path of DIAG is very close to the proven theoretical upper bound for this method.

Although n iterations of DIAG and IAG and one iteration of GD have the same complexity in terms of the total number of gradient evaluations, DIAG and IAG require more elementary operations than GD due to averaging of the gradients. In many large-scale machine learning applications the bottleneck is the computation of gradients; however, in the special case of quadratic programming problems, the additional elementary operations that IAG and DIAG require for computing the averages cannot be neglected. Therefore, to have a fair comparison between the incremental methods, i.e., IAG and DIAG, and the full-batch method, i.e., GD, we also compare these methods in terms of runtime, as shown in Figure 3 (see color figures in online version) for the quadratic programming problem given by (40). We observe in Figure 3 that the performance of GD becomes better relative to the incremental methods. In particular, for the case in which $n = 200$ and $\kappa = 10$, we observe that GD outperforms IAG and is marginally worse than DIAG. For the case in which $n = 200$ and $\kappa = 117$, where n is not significantly larger than κ, we observe that GD performs significantly better than IAG, while the convergence paths of GD and DIAG are close to each other. These observations lead to the conclusion that for quadratic programming problems, where the gradient evaluations are simply elementary operations, the cost of computing the averages in IAG and DIAG cannot be neglected.

We also compare the proposed DIAG method with SAG [29, 17] and Finito/MISO [10, 19], which are among the most successful stochastic incremental methods for solving the finite sum minimization problem in (1). For SAG we use the stepsize $1/16L$ as suggested in [29], and for the Finito algorithm we use the stepsize $1/2\mu$ as suggested in [10]. The left-hand plot in Figure 4 (see color figures in online version), which corresponds to the case in which $n = 200$ and $\kappa = 19$, shows that the performance of SAG and Finito are better than the one for DIAG, while they fluctuate more

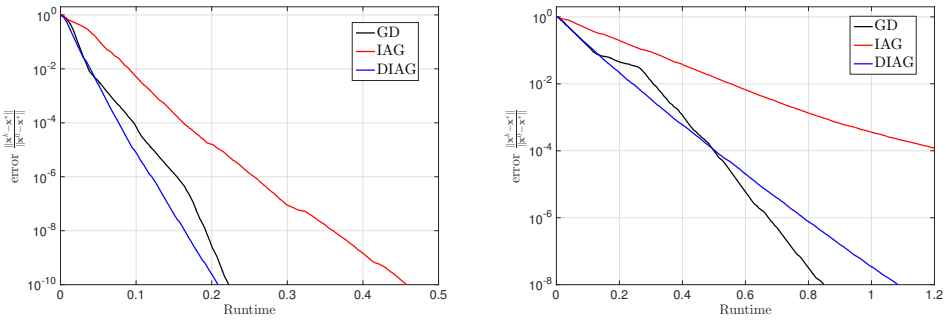

FIG. 3. *Relative error of GD, IAG, SAG, Finito, and DIAG versus number of gradient evaluations for the quadratic programming in* (40) *with* $n = 200$ *and* $\kappa = 10$ *(left) and* $n = 200$ *and* $\kappa = 117$ *(right).*

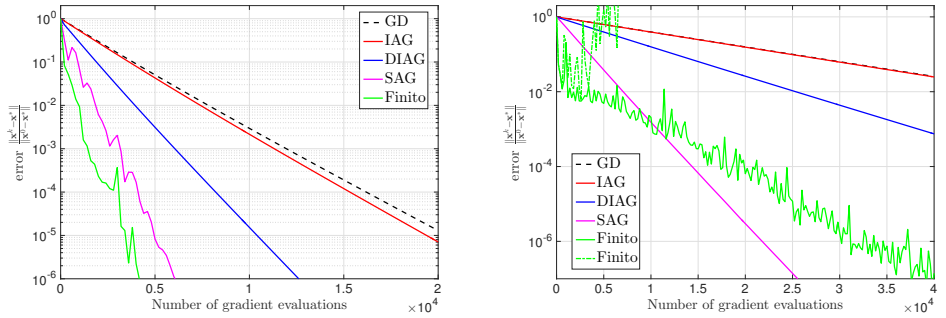

FIG. 4. *Relative error of GD, IAG, SAG, Finito, and DIAG versus number of gradient evaluations for the quadratic programming in* (40) *with* $n = 200$ *and* $\kappa = 19$ *(left) and* $n = 200$ *and* $\kappa = 120$ *(right).*

compared to IAG and DIAG. The gap between the performances of IAG and DIAG and their stochastic variants SAG and Finito comes from the fact that convergence guarantees of SAG and Finito hold for larger choices of stepsize compared to the ones for IAG and DIAG. But this improvement comes at the cost of moving from a deterministic convergence guarantee (for IAG and DIAG) to results that hold in expectation (for SAG and Finito) that might lead to volatile convergence paths, as shown in the left-hand plot in Figure 4.

The right-hand plot in Figure 4 illustrates the convergence paths of GD, IAG, SAG, Finito, and DIAG for a problem with large condition number $\kappa = 120$. We observe that SAG and Finito outperform DIAG; however, their convergence guarantees hold in expectation, which is a much weaker notion of convergence compared to deterministic convergence guarantees. As an example, one realization of Finito in the right-hand plot in Figure 4 performs pretty well, while the other one diverges. In contrast, the results for IAG and DIAG are deterministic, and for any realization of the iterates convergence to the optimal solution is guaranteed.

6.2. Logistic regression minimization. In this section, we compare the performance of GD, IAG, and DIAG in solving a binary classification problem. Consider the given training set $\mathcal{S} = \{\mathbf{u}_i, l_i\}_{i=1}^{i=n}$, which contains n realizations of the feature vectors $\mathbf{u}_i \in \mathbb{R}^p$ and respective labels l_i, where the labels are either -1 or 1. The goal is to find the optimal classifier $\mathbf{x}^* \in \mathbb{R}^p$ that minimizes the regularized logistic loss, which is given by

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-l_i\mathbf{x}^T\mathbf{u}_i)) + \frac{\lambda}{2}\|\mathbf{x}\|^2, \tag{41}$$

where the regularization term $(\lambda/2)\|\mathbf{x}\|^2$ is added to avoid overfitting. The problem in (41) is a particular case of the problem in (1) when the function f_i is defined as $f_i(\mathbf{x}) = \log(1+\exp(-l_i\mathbf{x}^T\mathbf{u}_i)) + (\lambda/2)\|\mathbf{x}\|^2$.

Note that the objective function f in (41) is strongly convex with the constant $\mu = \lambda$ and its gradients are Lipschitz continuous with the constant $L = \lambda + \zeta/4$, where $\zeta = \max_i \mathbf{u}_i^T\mathbf{u}_i$. It is easy to verify that the instantaneous functions f_i are also strongly convex with constant $\mu = \lambda$, and their gradients are Lipschitz continuous with constant $L = \lambda + \zeta/4$. This observation shows that the conditions in Assumption 1 hold for the logistic regression problem in (41). In this experiment, we normalize the samples to set the parameter $\zeta = 1$. Further, the regularization parameter is chosen as $\lambda = 1/\sqrt{n}$.

We apply GD, IAG, and DIAG to solve the logistic regression problem in (41) for the MNIST dataset [16]. We only use the samples that correspond to digits 0 and 8 and assign label $l_i = 1$ to the samples that correspond to digit 8 and label $l_i = -1$ to those associated with digit 0. We get a total of $n = 11,774$ training examples, each of dimension $p = 784$. The objective function error $f(\mathbf{x}^k) - f(\mathbf{x}^*)$ of the GD, IAG, and DIAG methods versus the number of passes over the dataset are shown in the left-hand plot in Figure 5 (see color figures in online version). We report the results for the stepsizes $\epsilon_{GD} = 2/(\mu + L)$, $\epsilon_{IAG} = 2/(nL)$, and $\epsilon_{DIAG} = 2/(\mu + L)$ as in the quadratic programming. We observe that the proposed DIAG method outperforms GD and IAG.

As we discussed in the quadratic programming example, n iterations of DIAG or IAG require more elementary operations than a single iteration of GD. Hence, we also compare these methods in terms of runtime, as shown in the right-hand plot in Figure 5. Note that in this case, in contrast to the quadratic programming example, gradient evaluations are more costly than the elementary operations required in the update and therefore we expect to gain more by running incremental methods. Indeed, we observe in the right plot in Figure 5 that the DIAG method outperforms GD significantly in terms of runtime. However, the performance of IAG and GD are almost similar to the one for GD. Comparing these results with the quadratic programming example shows that in terms of runtime incremental methods are more preferable in cases in which gradient evaluation is more costly than elementary operations.

7. Conclusion. In this paper we proposed a novel incremental method for solving the average of a set of n smooth and strongly convex functions. The proposed double incremental aggregated gradient method (DIAG) uses the aggregated average of both variables and gradients to update its iterate in contrast to classic cyclic incremental methods that only use the average of gradients. The convergence analysis of the DIAG method guarantees improvement with respect to the gradient descent (GD) method. This result makes DIAG the first cyclic incremental method that improves GD under all circumstances. Moreover, we showed that the sequence of

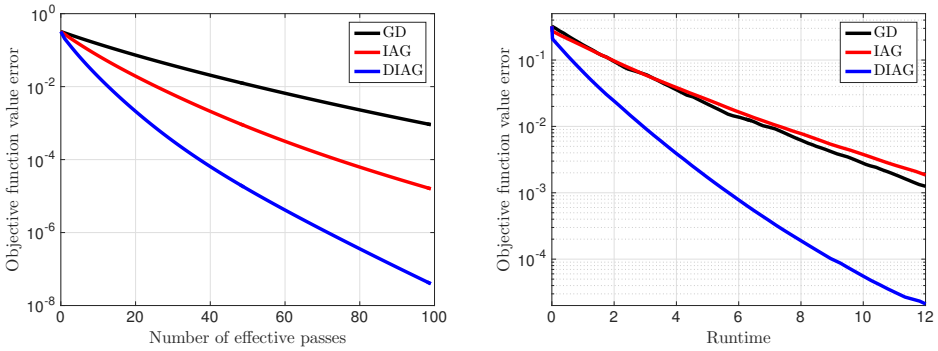

FIG. 5. *Convergence paths of GD, IAG, and DIAG in terms of number of effective passes over the dataset (left) and runtime (right) for the binary classification application.*

iterates generated by DIAG is linearly convergent. Numerical experiments matched the theoretical results and showcased the advantage of DIAG relative to GD and the classic incremental aggregated gradient (IAG) method.

As a future research direction, we aim to extend the double incremental idea to the accelerated gradient descent (AGD) method to obtain an incremental method that surpasses the optimal AGD method under any circumstances.

Appendix A. Proof of Lemma 1. Consider the update in (18). Subtract the optimal argument $\mathbf{x}^*$ from both sides of the equality to obtain

$$\mathbf{x}^{k+1} - \mathbf{x}^* = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{y}_i^k - \mathbf{x}^*) - \frac{\epsilon}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{y}_i^k). \tag{42}$$

Note that the global objective function gradient at the optimal point is null, i.e., $(1/n)\sum_{i=1}^{n}\nabla f_i(\mathbf{x}^*) = \mathbf{0}$. This observation in conjunction with the expression in (42) leads to

$$\begin{aligned}\mathbf{x}^{k+1} - \mathbf{x}^* &= \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{y}_i^k - \mathbf{x}^*\right) - \frac{\epsilon}{n}\sum_{i=1}^{n}\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\\ &= \frac{1}{n}\sum_{i=1}^{n}\Big[\mathbf{y}_i^k - \mathbf{x}^* - \epsilon\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\Big]. \end{aligned}\tag{43}$$

Compute the norm of both sides in (43) and use the Cauchy–Schwarz inequality to obtain

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \le \frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{y}_i^k - \mathbf{x}^* - \epsilon\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\right\|. \tag{44}$$

Now we proceed to derive an upper bound for each summand in (44). It can be shown that $\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*) = \nabla^2 f_i(\mathbf{u}_i^k)(\mathbf{y}_i^k - \mathbf{x}^*)$, where $\mathbf{u}_i^k$ is a convex combination of $\mathbf{y}_i^k$ and $\mathbf{x}^*$. Therefore,

$$\left\|\mathbf{y}_i^k - \mathbf{x}^* - \epsilon\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\right\| = \left\|\left(\mathbf{I} - \epsilon\nabla^2 f_i(\mathbf{u}_i^k)\right)(\mathbf{y}_i^k - \mathbf{x}^*)\right\|. \tag{45}$$

Since the functions f_i are μ-strongly convex and their gradients are L-Lipschitz continuous we can show that

$$\left\|\mathbf{y}_i^k - \mathbf{x}^* - \epsilon\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\right\| \le \max\{|1-\epsilon\mu|, |1-\epsilon L|\}\|\mathbf{y}_i^k - \mathbf{x}^*\|. \tag{46}$$

By setting the stepsize ϵ in (46) as $\epsilon = 2/(\mu + L)$, we can write

$$\left\|\mathbf{y}_i^k - \mathbf{x}^* - \epsilon\left(\nabla f_i(\mathbf{y}_i^k) - \nabla f_i(\mathbf{x}^*)\right)\right\| \leq \frac{\kappa - 1}{\kappa + 1}\ \|\mathbf{y}_i^k - \mathbf{x}^*\|, \tag{47}$$

where $\kappa = L/\mu$ is the function f_i condition number. By replacing the summands in the right hand side of (44) with their upper bounds $((\kappa - 1)/(\kappa + 1))\|\mathbf{y}_i^k - \mathbf{x}^*\|$, as shown in (47), we can show that the residual $\|\mathbf{x}^{k+1} - \mathbf{x}^*\|$ is bounded above as

$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \leq \left(\frac{\kappa - 1}{\kappa + 1}\right) \sum_{i=1}^{n} \frac{\|\mathbf{y}_i^k - \mathbf{x}^*\|}{n}. \tag{48}$$

Note that in the DIAG method we use a cyclic scheme to update the variables. Thus, the set of variables $\{\mathbf{y}_1^k, \ldots, \mathbf{y}_n^k\}$ is identical to the set of the last n iterates before the iterate $\mathbf{x}^{k+1}$, which is given by $\{\mathbf{x}^k, \ldots, \mathbf{x}^{k-n+1}\}$. Thus, we can replace the sum in $\sum_{i=1}^n \|\mathbf{y}_i^k - \mathbf{x}^*\|$ in (48) by the sum $\sum_{i=1}^n \|\mathbf{x}^{k-i+1} - \mathbf{x}^*\|$ and the claim in (21) follows.

Appendix B. Proof of Proposition 2. Consider the definition of the constant $\rho := (\kappa - 1)/(\kappa + 1)$, where $\kappa = L/\mu$ is the objective function condition number. Thus, if all the copies $\mathbf{y}_i$ are initialized at $\mathbf{x}^0$, the result in Lemma 1 implies that

$$\|\mathbf{x}^1 - \mathbf{x}^*\| \ \leq\ \rho\ \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{49}$$

We can use the same inequality for the second iterate to obtain

$$\|\mathbf{x}^2 - \mathbf{x}^*\| \leq\ \frac{\rho}{n}\ \|\mathbf{x}^1 - \mathbf{x}^*\| + \frac{\rho(n-1)}{n}\|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{50}$$

Replace $\|\mathbf{x}^1 - \mathbf{x}^*\|$ in (50) by its upper bound in (49) and regroup the terms to obtain

$$\begin{aligned} \|\mathbf{x}^2 - \mathbf{x}^*\| \ &\leq\ \frac{\rho^2}{n}\ \|\mathbf{x}^0 - \mathbf{x}^*\| + \frac{\rho(n-1)}{n}\|\mathbf{x}^0 - \mathbf{x}^*\| \\ &=\ \rho\left[1 - \frac{1-\rho}{n}\right]\|\mathbf{x}^0 - \mathbf{x}^*\|. \end{aligned} \tag{51}$$

Repeat the same process for the third residual $\|\mathbf{x}^3 - \mathbf{x}^*\|$ to obtain

$$\begin{aligned} \|\mathbf{x}^3 - \mathbf{x}^*\| \ &\leq\ \frac{\rho}{n}\ \|\mathbf{x}^2 - \mathbf{x}^*\| + \frac{\rho}{n}\ \|\mathbf{x}^1 - \mathbf{x}^*\| + \frac{\rho(n-2)}{n}\|\mathbf{x}^0 - \mathbf{x}^*\| \\ &\leq\ \rho\left[1 - \frac{2(1-\rho)}{n} - \frac{\rho(1-\rho)}{n^2}\right]\|\mathbf{x}^0 - \mathbf{x}^*\|, \end{aligned} \tag{52}$$

where in the second inequality we use the bounds in (49) and (50). Since the term $-\rho(1-\rho)/n^2$ is negative we can drop this term and show that the residual $\|\mathbf{x}^3 - \mathbf{x}^*\|$ is upper bounded by

$$\|\mathbf{x}^3 - \mathbf{x}^*\| \ \leq\ \rho\left[1 - \frac{2(1-\rho)}{n}\right]\|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{53}$$

By following the same logic we can show that for the first n residuals $\{\|\mathbf{x}^k - \mathbf{x}^*\|\}_{k=1}^n$ the following inequality holds:

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho\left[1 - \frac{(k-1)(1-\rho)}{n}\right]\|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k = 1, \ldots, n. \tag{54}$$

Thus, the result in (22) holds.

Now we proceed to show that the result in (23) holds for $k = n+1, \ldots, 2n$. According to the result in Lemma 1 we can write

$$\|\mathbf{x}^{n+1} - \mathbf{x}^*\| \leq \frac{\rho}{n} \|\mathbf{x}^n - \mathbf{x}^*\| + \cdots + \frac{\rho}{n}\|\mathbf{x}^1 - \mathbf{x}^*\|. \tag{55}$$

By replacing each summand in the right-hand side of (55) by its upper bound in (54) we obtain

$$\begin{aligned}\|\mathbf{x}^{n+1} - \mathbf{x}^*\| &\leq \frac{\rho}{n} \left[\sum_{i=1}^{n} \rho \left[1 - \frac{(i-1)(1-\rho)}{n}\right]\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \\ &= \rho^2 \left[1 - \frac{(1-\rho)(n-1)}{2n}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \end{aligned} \tag{56}$$

It follows from the result in (56) that the residual $\|\mathbf{x}^{n+1} - \mathbf{x}^*\|$ is upper bounded by

$$\|\mathbf{x}^{n+1} - \mathbf{x}^*\| \leq \rho^2 \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{57}$$

Therefore, the result in (23) holds for $k = n+1$. To prove the claim for $k = n+2$, first note that based on the upper bounds in (54) we can show that

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho \left[1 - \frac{(1-\rho)}{n}\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k = 2, \ldots, n. \tag{58}$$

Considering the inequality

$$\left[1 - \frac{1-\rho}{n}\right] \leq \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \tag{59}$$

and the upper bounds in (58) we obtain that

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k = 2, \ldots, n. \tag{60}$$

In addition, the result in (57) and the fact that $\rho < 1$ imply that the term $\|\mathbf{x}^{n+1} - \mathbf{x}^*\|$ is also upper bounded by

$$\|\mathbf{x}^{n+1} - \mathbf{x}^*\| \leq \rho \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{61}$$

Now, based on the result in Lemma 1, we can write

$$\|\mathbf{x}^{n+2} - \mathbf{x}^*\| \leq \frac{\rho}{n} \left[\|\mathbf{x}^{n+1} - \mathbf{x}^*\| + \cdots + \|\mathbf{x}^2 - \mathbf{x}^*\|\right]. \tag{62}$$

The inequalities in (60) and (61) show that all the summands in (62) are bounded by the same upper bound. Replace these term by the upper bound to obtain

$$\|\mathbf{x}^{n+2} - \mathbf{x}^*\| \leq \rho^2 \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{63}$$

and the claim in (23) for $k = n+2$ follows.

Since the constant ρ is strictly less than 1, we can replace the upper bound in (63) by

$$\|\mathbf{x}^{n+2} - \mathbf{x}^*\| \leq \rho \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{64}$$

Using the same argument, we can say that the upper bound in (64) holds for $k = n+2, \ldots, 3$ and apply the result in Lemma 1 to show that the distance $\|\mathbf{x}^{n+3} - \mathbf{x}^*\|$ is bounded above by

$$\|\mathbf{x}^{n+3} - \mathbf{x}^*\| \leq \rho^2 \left[1 - \frac{1-\rho}{n} \times \min\left\{1, \frac{(n-1)}{2n}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{65}$$

which yields the claim in (23) for $k = n+3$. By repeating the steps in (64) and (65) we can conclude that the result in (23) holds for $k = n+1, \ldots, 2n$.

The proof for steps $k > 2n$ is similar to the argument used for the steps $k = n+1, \ldots, 2n$, although we write it in a formal manner by using induction.

Assume that for $k = nj+1, \ldots, nj+n$ the following inequality holds:

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho^{\lfloor \frac{k-1}{n} \rfloor + 1} \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{66}$$

We intend to prove the same inequalities hold for $k = n(j+1)+1, \ldots, n(j+1)+n$. Note that the result in (66) is satisfied for $j = 1$, which corresponds to the iterates $k = n+1, \ldots, 2n$, and the base of induction holds.

As we assume that the result in (66) holds for $k = nj+1, \ldots, nj+n$, we obtain that

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq \rho^{j+1} \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\| \tag{67}$$

for $k = nj+1, \ldots, nj+n$. According to Lemma 1, the residual $\|\mathbf{x}^{n(j+1)+1} - \mathbf{x}^*\|$ is bounded above by

$$\|\mathbf{x}^{n(j+1)+1} - \mathbf{x}^*\| \leq \frac{\rho}{n} \|\mathbf{x}^{nj+1} - \mathbf{x}^*\| + \cdots + \frac{\rho}{n} \|\mathbf{x}^{nj+n} - \mathbf{x}^*\|. \tag{68}$$

Replacing the summands in the right-hand side of (68) by their upper bound in (67) implies that

$$\|\mathbf{x}^{n(j+1)+1} - \mathbf{x}^*\| \leq \rho^{j+2} \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{69}$$

which yields the claim in (66) for $k = n(j+1)+1$.

Now use the result in (69) and the inequality $\rho < 1$ to obtain

$$\|\mathbf{x}^{n(j+1)+1} - \mathbf{x}^*\| \leq \rho^{j+1} \left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right] \|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{70}$$

Proceed by writing the result in Lemma 1 for $k = n(j+1)+2$ to obtain

$$\|\mathbf{x}^{n(j+1)+2} - \mathbf{x}^*\| \leq \frac{\rho}{n} \|\mathbf{x}^{nj+2} - \mathbf{x}^*\| + \cdots + \frac{\rho}{n} \|\mathbf{x}^{nj+n+1} - \mathbf{x}^*\|. \tag{71}$$

Replace the summands in the right-hand side of (71) by their upper bounds in (67) and (70), which are the same upper bounds, to obtain

$$\|\mathbf{x}^{n(j+1)+2} - \mathbf{x}^*\| \leq \ \rho^{j+2}\left[1 - \frac{(1-\rho)}{n} \times \min\left\{1, \frac{n-1}{2}\right\}\right]\|\mathbf{x}^0 - \mathbf{x}^*\|, \tag{72}$$

and the claim in (66) for $k = n(j+1)+2$ follows. By repeating the steps from (70) to (72) we can show that the same result holds for $k = k = n(j+1)+3, \ldots, k = n(j+1)+n$. Thus, the inequality in (66) holds for $k = n(j+1)+1, \ldots, n(j+1)+n$. The induction is complete, which implies that the claim in (23) holds.

Appendix C. Proof of Theorem 4. Considering the result in (22) and the definition of the constant a in (26) we obtain that

$$\|\mathbf{x}^k - \mathbf{x}^*\| \leq a\gamma^k\|\mathbf{x}^0 - \mathbf{x}^*\| \quad \text{for } k = 1, \ldots, n. \tag{73}$$

Thus, the inequality in (25) holds for steps $k = 1, \ldots, n$.

Now we proceed to show that the claim in (25) also holds for $k > n$. To do so, we use an induction argument. Let's assume we aim to show that the inequality in (25) holds for $k = j$, while it holds for the last n iterates $k = j-1, \ldots, j-n$. According to the result in Lemma 1 we can write

$$\|\mathbf{x}^j - \mathbf{x}^*\| \leq \rho\left[\frac{\|\mathbf{x}^{j-1} - \mathbf{x}^*\| + \cdots + \|\mathbf{x}^{j-n} - \mathbf{x}^*\|}{n}\right], \tag{74}$$

where $\rho = (\kappa - 1)/(\kappa + 1)$. Based on the induction assumption, for steps $k = j - 1, \ldots, j-n$, the result in (25) holds. Thus, we can replace the terms in the right-hand side of (74) by the upper bounds from (25). This substitution implies

$$\begin{aligned}\|\mathbf{x}^j - \mathbf{x}^*\| &\leq \frac{\rho a}{n}\left[\gamma^{j-1} + \cdots + \gamma^{j-n}\right]\|\mathbf{x}^0 - \mathbf{x}^*\| \\ &= \frac{\rho a\gamma^{j-n}(1-\gamma^n)}{n(1-\gamma)}\|\mathbf{x}^0 - \mathbf{x}^*\|. \end{aligned} \tag{75}$$

Rearranging the terms in (27) allows us to show that $(\rho(1-\gamma^n))/(n(1-\gamma))$ is bounded above by γ^n. This is true since

$$\begin{aligned}\gamma^{n+1} - \left(1 + \frac{\rho}{n}\right)\gamma^n + \frac{\rho}{n} \leq 0 \ &\iff \ \frac{\rho}{n}(1-\gamma^n) - \gamma^n(1-\gamma) \leq 0 \\ &\iff \ \frac{\rho(1-\gamma^n)}{n(1-\gamma)} \leq \gamma^n. \end{aligned} \tag{76}$$

Therefore, we can replace the term $(\rho(1-\gamma^n))/(n(1-\gamma))$ in (75) by its upper bound γ^n to obtain

$$\|\mathbf{x}^j - \mathbf{x}^*\| \leq a\gamma^j\|\mathbf{x}^0 - \mathbf{x}^*\|. \tag{77}$$

The result in (77) completes the proof. Thus, by induction the claim in (25) holds for all $k \geq 1$ if the conditions in (26) and (27) are satisfied.

Appendix D. Proof of Proposition 5. To prove the claim in Proposition 5 we first derive the following lemma.

LEMMA 10. *For all $n \geq 1$ and $0 \leq \phi \leq 1$ we have*

$$\left(1 - \frac{\phi}{n}\right)^n \leq \left(1 - \frac{\phi}{n+1}\right)^{n+1} \tag{78}$$

Proof. Consider the function $h(x) = (1-(\phi/x))^x$ for $x > 1$. The natural logarithm of the function $h(x)$ is given by $\ln(h(x)) = x\ln(1-(\phi/x))$. Compute the derivative of both sides with respect to x to obtain

$$\frac{dh}{dx} \times \frac{1}{h(x)} = \ln\left(1-\frac{\phi}{x}\right) + x \times \frac{\frac{\phi}{x^2}}{1-\frac{\phi}{x}} \tag{79}$$

By multiplying both sides by $h(x)$, replacing $h(x)$ by the expression $(1-\frac{\phi}{x})^x$, and simplifying the terms we obtain that the derivative of the function $h(x)$ is given by

$$\frac{dh}{dx} = \left(1-\frac{\phi}{x}\right)^x \left[\ln\left(1-\frac{\phi}{x}\right) + \frac{\frac{\phi}{x}}{1-\frac{\phi}{x}}\right]. \tag{80}$$

Note that the sum $\ln(1-u) + u/(1-u)$ is always positive for $0 < u < 1$. By setting $u := \phi/x$, we can conclude that the term on the right-hand side of (80) is positive for $x > 1$. Therefore, the derivative dh/dx is always positive for $x > 1$. Thus, the function $h(x)$ is an increasing function for $x > 1$ and we can write

$$\left(1-\frac{\phi}{n}\right)^n \leq \left(1-\frac{\phi}{n+1}\right)^{n+1} \tag{81}$$

for $0 \leq \phi \leq 1$ and $n > 1$. It remains to show that the same claim is also valid for $n = 1$, which is equivalent to the inequality

$$1-\phi \leq \left(1-\frac{\phi}{2}\right)^2. \tag{82}$$

It is trivial to show (82) holds, and, therefore, the claim in (78) holds for all $n \geq 1$. ∎

Now proceed to prove the claim in Proposition 5 using the result in Lemma 10. To prove that the feasible set of the condition in (27) is nonempty we show that $\gamma = \rho^{1/n}$ satisfies the inequality in (27). In other words,

$$\rho^{\frac{n+1}{n}} - \left(1+\frac{\rho}{n}\right)\rho + \frac{\rho}{n} \leq 0. \tag{83}$$

Divide both sides of (83) by ρ and regroup the terms to obtain the inequality

$$\rho \leq \left(1-\frac{1-\rho}{n}\right)^n, \tag{84}$$

which is equivalent to (83). In other words, the inequality in (84) is a necessary and sufficient condition for the condition in (83).

Recall the result in Lemma 10. By setting $\phi = 1-\rho$ we obtain that

$$\rho = \left(1-\frac{1-\rho}{1}\right)^1 \leq \left(1-\frac{1-\rho}{2}\right)^2 \leq \cdots \leq \left(1-\frac{1-\rho}{n}\right)^n \tag{85}$$

for $n \geq 1$. Thus, the inequality in (84) holds, and, consequently, the inequality in (83) is valid. Therefore, $\gamma = \rho^{1/n}$ satisfies the inequality in (27).

Then, we can define a as the smallest constant that satisfies (26) for the choice $\gamma = \rho^{1/n}$, which is given by

$$a = \max_{k=1,\ldots,n} \left(1-\frac{(k-1)(1-\rho)}{n}\right)\rho^{1-\frac{k}{n}}. \tag{86}$$

Therefore, $\gamma = \rho^{1/n}$ and the constant a in (86) satisfy the conditions in (26) and (27), and the claim in Proposition 5 follows.

Appendix E. Proof of Theorem 9.

(i)This follows directly from the Perron–Frobenius theorem for irreducible nonnegative matrices [12, Theorem 8.4.4].

(ii) By [12, Theorem 8.5.1], we also have

$$\lim_{k\to\infty} \frac{\mathbf{M}_\rho^k}{\lambda^*(\rho)^k} = \mathbf{u}\mathbf{v}^T, \tag{87}$$

where $\mathbf{u}$, $\mathbf{v}$ are the right and left eigenvectors of $\mathbf{M}_\rho$ corresponding to the eigenvalue $\lambda^*(\rho)$ normalized to satisfy $\mathbf{v}^T\mathbf{u} = 1$. Note also that

$$d^k = \mathbf{e}_1^T\mathbf{d}^{k+1} = \mathbf{e}_1^T\mathbf{M}_\rho^{k-n+1}\mathbf{d}^n. \tag{88}$$

Therefore,

$$\lim_{k\to\infty} d^{k+1}/d^k = \lim_{k\to\infty} \frac{\mathbf{e}_1^T\mathbf{M}_\rho^{k-n+2}\mathbf{d}^n}{\mathbf{e}_1^T\mathbf{M}_\rho^{k-n+1}\mathbf{d}^n} = \lim_{k\to\infty} \frac{\mathbf{e}_1^T\mathbf{M}_\rho^{k-n+1}\mathbf{d}^n/\lambda^*(\rho)^{k-n+1}}{\mathbf{e}_1^T\mathbf{M}_\rho^{k-n}\mathbf{d}^n/\lambda^*(\rho)^{k-n+1}}, \tag{89}$$

where in the last inequality we divided both numerator and denominator with the same factor $\lambda^*(\rho)^{k-n+1}$. Simplifying and using (87), we obtain

$$\lim_{k\to\infty} d^{k+1}/d^k = \lim_{k\to\infty} \lambda^*(\rho)\frac{\mathbf{e}_1^T\mathbf{u}\mathbf{v}^T\mathbf{d}^n}{\mathbf{e}_1^T\mathbf{u}\mathbf{v}^T\mathbf{d}^n} = \lambda^*(\rho). \tag{90}$$

(iii) A direct consequence of [12, Theorem 8.1.22] is that $\lambda^*(\rho) \geq \rho$; this proves the lower bound on $\lambda^*(\rho)$. To get the upper bound, let $\mathbf{1} = [1\ 1\ 1\ \cdots\ 1]^T$ be the vector of ones. We will show that

$$\mathbf{M}_\rho^{2n}\mathbf{1} < \rho^2\mathbf{1}, \tag{91}$$

where the notation "$<$" denotes the componentwise inequality for vectors. Then, by [12, Corollary 8.1.29], this would imply

$$\lambda^*(\rho)^{2n} < \rho^2,$$

which is equivalent to the desired upper bound. It is a straightforward computation to show that if we set $\mathbf{d}^n = \mathbf{1}$, then after a simple induction argument we obtain $d^n = \rho$ and $d^{n+1} < \rho$, $d^{n+2} < \rho$, $\ldots$, $d^{2n-1} < \rho$, i.e.,

$$\mathbf{d}^{2n} = \begin{bmatrix} d^{2n-1} \\ d^{2n-2} \\ \cdots \\ d^n \end{bmatrix} = \mathbf{M}_\rho^n\mathbf{d}^n = \mathbf{M}_\rho^n\mathbf{1} = \rho\mathbf{v} \tag{92}$$

for a vector $\mathbf{v} = [v_1, v_2, \ldots, v_n]^T$, where $v_i < 1$ if $i < n$ and $v_n = 1$. Using similar arguments we can write

$$\mathbf{d}^{3n} = \begin{bmatrix} d^{3n-1} \\ d^{3n-2} \\ \cdots \\ d^{2n} \end{bmatrix} = \mathbf{M}_\rho^{2n}\mathbf{d}^n = \mathbf{M}_\rho\mathbf{M}_\rho^n\mathbf{1} = \rho\mathbf{M}_\rho\mathbf{v} \tag{93}$$

and a straightforward computation shows that $\mathbf{M}_\rho\mathbf{v} < \rho\mathbf{1}$. Combining this inequality with the previous equation proves (91) and concludes the proof.

(iv) It follows from (35) that the roots of the polynomial h are the same as the roots of the polynomial T except that h has an additional root at 1. By part (i) and (iii), $\lambda^*(\rho)$ is the largest real root of T and $0 < \lambda^*(\rho) < 1$. Therefore, it is also the largest real root of the function h over the interval $(0, 1)$, which is equal to γ_0.

REFERENCES

[1] D. P. Bertsekas, *A new class of incremental gradient methods for least squares problems*, SIAM J. Optim., 7 (1997), pp. 913–926.

[2] D. P. Bertsekas, *Incremental proximal methods for large scale convex optimization*, Math. Program., 129 (2011), pp. 163–195.

[3] D. Blatt, A. O. Hero, and H. Gauchman, *A convergent incremental gradient method with a constant step size*, SIAM J. Optim., 18 (2007), pp. 29–51.

[4] L. Bottou, *Large-scale machine learning with stochastic gradient descent*, in Proceedings of COMPSTAT'2010, Y. Lechevallier and G. Saporta, eds., Physica-Verlag, Heidelberg, 2010, pp. 177–186.

[5] L. Bottou and Y. Le Cun, *On-line learning for very large data sets: Research articles*, Appl. Stoch. Models Bus. Ind., 21 (2005), pp. 137–151.

[6] F. Bullo, J. Cortes, and S. Martinez, *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*, Princeton University Press, Princeton, NJ, 2009.

[7] Y. Cao, W. Yu, W. Ren, and G. Chen, *An overview of recent progress in the study of distributed multi-agent coordination*, IEEE Trans. Ind. Informat., 9 (2013), pp. 427–438.

[8] V. Cevher, S. Becker, and M. Schmidt, *Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics*, IEEE Signal Process. Mag., 31 (2014), pp. 32–43.

[9] A. Defazio, F. R. Bach, and S. Lacoste-Julien, *SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives*, in Adv. Neural Inf. Process. Syst. 27, MIT Press, Cambridge, MA, 2014, pp. 1646–1654.

[10] A. Defazio, J. Domke, and T. S. Caetano, *Finito: A faster, permutable incremental gradient method for big data problems*, in Proceedings of the 31st International Conference on Machine Learning, Proc. Mach. Learn. Res. 32, 2014, pp. 1125–1133.

[11] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, *On the convergence rate of incremental aggregated gradient algorithms*, SIAM J. Optim., 27 (2017), pp. 1035–1048.

[12] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, New York, 1991.

[13] B. Johansson, M. Rabi, and M. Johansson, *A randomized incremental subgradient method for distributed optimization in networked systems*, SIAM J. Optim., 20 (2009), pp. 1157–1170.

[14] R. Johnson and T. Zhang, *Accelerating stochastic gradient descent using predictive variance reduction*, in Adv. Neural Inf. Process. Syst. 26, MIT Press, Cambridge, MA, 2013, pp. 315–323.

[15] J. Konečnỳ and P. Richtárik, *Semi-Stochastic Gradient Descent Methods*, preprint, arXiv:1312.1666, 2013.

[16] Y. Le Cun, C. Cortes, and C. J. Burges, *The MNIST Database of Handwritten Digits*, 1998.

[17] N. Le Roux, M. Schmidt, and F. Bach, *A stochastic gradient method with an exponential convergence rate for finite training sets*, in Adv. Neural Inf. Process. Syst. 25, MIT Press, Cambridge, MA, 2012, pp. 2672–2680.

[18] C. G. Lopes and A. H. Sayed, *Diffusion least-mean squares over adaptive networks: Formulation and performance analysis*, IEEE Trans. Signal Process., 56 (2008), pp. 3122–3136.

[19] J. Mairal, *Incremental majorization-minimization optimization with application to large-scale machine learning*, SIAM J. Optim., 25 (2015), pp. 829–855.

[20] A. Mokhtari, M. Gürbüzbalaban, and A. Ribeiro, *A double incremental aggregated gradient method with linear convergence rate for large-scale optimization*, in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE Press, Piscataway, NJ, 2017, pp. 4696–4700.

[21] A. Mokhtari and A. Ribeiro, *DSA: Decentralized double stochastic averaging gradient algorithm*, J. Mach. Learn. Res., 17 (2016), pp. 1–35.

[22] A. Nedic and D. P. Bertsekas, *Incremental subgradient methods for nondifferentiable optimization*, SIAM J. Optim., 12 (2001), pp. 109–138.

[23] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Appl. Optim.

87, Springer Science & Business Media, New York, 2004.
[24] M. G. Rabbat and R. D. Nowak, *Quantized incremental algorithms for distributed optimization*, IEEE J. Sel. Areas Commun., 23 (2005), pp. 798–808.
[25] S. S. Ram, A. Nedić, and V. V. Veeravalli, *Incremental stochastic subgradient algorithms for convex optimization*, SIAM J. Optim., 20 (2009), pp. 691–717.
[26] A. Ribeiro, *Ergodic stochastic optimization algorithms for wireless communication and networking*, IEEE Trans. Signal Process., 58 (2010), pp. 6369–6386.
[27] A. Ribeiro, *Optimal resource allocation in wireless communication and networking*, EURASIP J. Wirel. Commun. Netw., 2012 (2012), pp. 1–19.
[28] H. Robbins and S. Monro, *A stochastic approximation method*, Ann. Math. Stat., 22 (1951), pp. 400–407.
[29] M. Schmidt, N. Le Roux, and F. Bach, *Minimizing finite sums with the stochastic average gradient*, Math. Program., 162 (2017), pp. 83–112.
[30] S. Shalev-Shwartz and N. Srebro, *SVM optimization: Inverse dependence on training set size*, in Proceedings of the 25th International Conference on Machine Learning (ICML 2008), ACM, New York, 2008, pp. 928–935.
[31] S. Shalev-Shwartz and T. Zhang, *Stochastic dual coordinate ascent methods for regularized loss*, J. Mach. Learn. Res., 14 (2013), pp. 567–599.
[32] S. Shalev-Shwartz and T. Zhang, *Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization*, Math. Program., 155 (2016), pp. 105–145.
[33] P. Tseng, *An incremental gradient(-projection) method with momentum term and adaptive stepsize rule*, SIAM J. Optim., 8 (1998), pp. 506–531.
[34] P. Tseng and S. Yun, *Incrementally updated gradient methods for constrained and regularized optimization*, J. Optim. Theory Appl., 160 (2014), pp. 832–853.
[35] N. D. Vanli, M. Gürbüzbalaban, and A. Ozdaglar, *Global Convergence Rate of Proximal Incremental Aggregated Gradient Methods*, preprint, arXiv:1608.01713, 2016.
[36] L. Xiao and T. Zhang, *A proximal stochastic gradient method with progressive variance reduction*, SIAM J. Optim., 24 (2014), pp. 2057–2075.
[37] L. Zhang, M. Mahdavi, and R. Jin, *Linear convergence with condition number independent access of full gradients*, in Adv. Neural Inf. Process. Syst. 26, MIT Press, Cambridge, MA, 2013, pp. 980–988.