Balancing Communication and Computation in Distributed Optimization

Albert S. Berahas, Raghu Bollapragada, Nitish Shirish Keskar, and Ermin Wei, *Member, IEEE*

***Abstract*—Methods for distributed optimization have received significant attention in recent years owing to their wide applicability in various domains including machine learning, robotics and sensor networks. A distributed optimization method typically consists of two key components: communication and computation. More specifically, at every iteration (or every several iterations) of a distributed algorithm, each node in the network requires some form of information exchange with its neighboring nodes (communication) and the computation step related to a (sub)-gradient (computation). The standard way of judging an algorithm via only the number of iterations overlooks the complexity associated with each iteration. Moreover, various applications deploying distributed methods may prefer a different composition of communication and computation.**

Motivated by this discrepancy, in this work we propose an adaptive cost framework which adjusts the cost measure depending on the features of various applications. We present a flexible algorithmic framework, where communication and computation steps are explicitly decomposed to enable algorithm customization for various applications. We apply this framework to the well-known distributed gradient descent (DGD) method, and show that the resulting customized algorithms, which we call DGD^t, NEAR-DGD^t and NEAR-DGD$^+$, compare favorably to their base algorithms, both theoretically and empirically. The proposed NEAR-DGD$^+$ algorithm is an exact first-order method where the communication and computation steps are nested, and when the number of communication steps is adaptively increased, the method converges to the optimal solution. We test the performance and illustrate the flexibility of the methods, as well as practical variants, on quadratic functions and classification problems that arise in machine learning, in terms of iterations, gradient evaluations, communications and the proposed cost framework.

***Index Terms*—Distributed Optimization, Communication, Optimization Algorithms, Network Optimization**

I. Introduction

THE problem of optimizing an objective function by employing a distributed procedure using multiple agents in a connected network has gained significant attention over the last years. This is motivated by its wide applicability to many important engineering and scientific domains such as, wireless sensor networks [1], [2], [3], [4], multi-vehicle and multi-robot networks [5], [6], [7], smart grids [8], [9] and machine learning [10], [11]. In such problems, each agent (or node) has access to a component of the overall objective function and can only communicate with its neighbors in the underlying network. The collective goal is to minimize the summation of individual components. Formally, the system-wide problem can be represented as

$$\min_{x \in \mathbb{R}^p} \quad h(x) = \sum_{i=1}^{n} f_i(x), \tag{I.1}$$

where n represents the number of agents in the network, convex function $h : \mathbb{R}^p \to \mathbb{R}$ is the *global objective function*, convex function $f_i : \mathbb{R}^p \to \mathbb{R}$ for each $i \in \{1, 2, ..., n\}$ is the *local objective function* available only to node i, and vector $x \in \mathbb{R}^p$ is the decision variable that the agents are optimizing cooperatively. This setup naturally calls for *distributed (optimization) algorithms*, where the agents iteratively perform local *computations* based on a local objective function and local *communications*, i.e., information exchange with their immediate neighbors in the underlying network, to solve the system-wide problem (I.1).

In order to decouple the computation of individual agents, problem (I.1) is often reformulated as the following consensus optimization problem by introducing a local copy $x_i \in \mathbb{R}^p$ for each agent $i \in \{1, 2, ..., n\}$ [12], [13],

$$\begin{aligned} \min_{x_i \in \mathbb{R}^p} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & x_i = x_j, \quad \forall i, j \in \mathcal{N}_i, \end{aligned} \tag{I.2}$$

where $\mathcal{N}_i$ denotes the set of neighbors of agent i. In this formulation, the local objective function of the i^{th} agent only depends on the local copy x_i. An equality constraint, often referred to as the *consensus* constraint, is imposed to enforce that the local copies of neighboring nodes are equal. Since the underlying network is connected and the consensus constraint ensures that all local copies are equal, problems (I.1) and (I.2) are equivalent.

While there is a proliferating literature on developing distributed optimization methods for the above problem, most follow the conventional approach of tracking the number of iterations to judge the efficiency of a distributed algorithm, i.e., the best algorithm achieves optimality in the minimal number of iterations, and overlook the complexity associated with each iteration. In this work, we propose an alternative metric, an adaptive cost framework (in Section II) to account for the different environments and hardware constraints of various applications, where distributed optimization methods are used.

A. S. Berahas was with the Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA. (email: albertberahas@u.northwestern.edu)

R. Bollapragada was with the Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA. (email: raghu.bollapragada@u.northwestern.edu)

N. S. Keskar was with Salesforce Research, Palo Alto, CA, USA. Work was performed when author was at Northwestern University. (email: keskar.nitish@u.northwestern.edu)

E. Wei was with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL, USA. (email: ermin.wei@northwestern.edu)

In this new cost framework, we consider communication and computation costs separately and weigh them using parameters specific to the environment. This cost framework also motivates our development of a class of flexible distributed methods, where we decompose communication and computation steps. This new class of algorithms is then customizable depending on the application.

A. Literature Review

Our work is related to the growing literature on distributed algorithms for solving problem (I.2). We outline the various lines of research next. One class of methods build upon the seminal works [12], [14], which proposed a parallel computation framework. In [13], the authors introduced a first-order primal iterative method, known as *distributed (sub)-gradient descent (DGD)*. In one step of DGD, each agent updates its estimate of the solution via a linear combination of a gradient descent step with respect to its local objective function and a weighted average with local neighbors (also known as a consensus step). A number of later contributions [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30] extended DGD to other settings, including stochastic networks, constrained problems, and noisy environments. *Coordinate descent* type methods, in either primal or dual space, have also been used in the distributed setting [31], [32], [33]. Another line of research is based on Nesterov's *dual averaging* algorithm [34], whose distributed version was proposed and analyzed in [10]. *Dual decomposition based* methods have also recently gained much attention. This class of methods includes augmented Lagrangian methods and Alternating Direction Method of Multiplier (ADMM) [35], [36], [37], [38], [39], [40], [41], [42], [3], [43], [44], [45], [46]. The last category includes *second-order methods*, where Newton-type methods are used to obtain faster rates of convergence [47], [48], [49], [50]. All these methods adopt the standard iteration count metric.

Closely related to our work are a few very recent contributions that incorporate communication considerations in the design of distributed algorithms [51], [52], [53], [54], [55], [56]. In [52], [53], [55], [56], the goal was to develop one particular method, where the number of communication steps is reasonable compared to the iteration complexity. While these methods are communication-efficient (with respect to some metric), they lack the flexibility to adapt to different environments. In [51], the authors consider how to design a network topology that is communication-efficient, assuming the network topology can be controlled. The closest work to ours is [54]. In this recent work, the authors control the frequency of communication steps and analyze the time performance of a dual averaging based algorithm. The goal of [54] was to show that speed-up is possible when the agents communicate less frequently. While this work pioneers the idea of adjusting the relative frequencies of communication and computation, it focuses solely on reducing the runtime of the algorithm and overlooks other important aspects, such as energy consumption.

In this paper, in order to demonstrate our new cost framework (II.1), we consider decoupling the communication and computation steps of the well-studied DGD method and vary the frequency of these steps. On this front, our work is related to [57], [58], [17]. In [58], the authors propose to increase the number of communication steps at rate k, where k is the number of iterations, to ensure convergence with a fixed stepsize (aka steplength) for proximal gradient-based algorithms on composite nonsmooth convex problems. In [17], the authors extended this idea to smooth problems and propose to increase communication at rate $\log(k)$.

In [57], the author propose two algorithms for quadratic problems: CTA (Combine-then-Adapt) and ATC (Adapt-then-Combine). Both of these methods stem from a new way of combining communication and computation steps and they differ in the order that the consensus and gradient operations are performed. While our method employs a similar way to decompose and combine communication and computation steps, our method is not restricted to using exactly one consensus step and one gradient step at each iteration. Hence, it is more general and offers flexibility in how to combine these steps. In particular, our convergence guarantee holds for many different ways of increasing the frequency of communication including k and $\log(k)$, as appeared in [58] and [17], respectively.

One major problem of the standard DGD method is that it only converges to a neighborhood of the optimal solution when a constant stepsize is employed. Recently, there have been many new distributed algorithms [29], [59], [60], [61], [62], [47] that can achieve exact convergence with a constant stepsize. In [29], [61], [60], [47], the authors also show their respective algorithms can achieve a linear rate of convergence. Our work is also related to this line of literature as many instances of our proposed class of algorithms achieve exact convergence. Moreover, one instance of our method converges linearly with respect to number of gradient computations, and sublinearly with respect to number of communications or our cost framework. While the proposed method has many similarities with the existing algorithms, it also boasts unique flexibility and adaptability.

B. Contributions

Our innovations in this paper are on two fronts: (i) the new adaptive cost framework, and (ii) the proposed class of flexible algorithms motivated by this framework. We next describe our main contributions:

- We introduce a metric of performance based on the weighted combination of costs of both communication and computation steps. This metric is adaptive to, and can accurately characterize, different features of the application environments independent of the distributed algorithms.
- We decompose the communication and computation steps of DGD to enable algorithm customization. Based on this, we propose three classes of related flexible algorithms, which we call DGD^t, NEAR-DGD^t and NEAR-DGD$^+$. We can tune the instances in these classes to balance the communication and computation costs according to the application.
- We develop a class of exact first-order methods with constant stepsize (NEAR-DGD$^+$), based on nesting the

communication and computation steps, and increasing the number of consensus steps performed as the algorithm evolves. When we increase the number of consensus steps at rate k, where k is the number of iterations, then we obtain an algorithm that achieves exact convergence at a linear rate. In particular, to get an ϵ-accurate solution, we need $\mathcal{O}(\log(1/\epsilon))$ numbers of gradient computations and $\mathcal{O}(\log(1/\epsilon)^2)$ number of communication rounds.

- We illustrate the empirical performance of some instances of the proposed class of methods on quadratic and logistic regression problems as measured by our new cost framework. We also demonstrate some practical instances of the class of methods that perform very well in practice in terms of iterations, number of communications, number of gradients and combined cost.

In summary, our main contribution is the proposed cost framework in conjunction with the decomposition of the communication and computation steps. This allows for flexibility in algorithmic design, for a class of theoretically sound and efficient algorithms, and the first step towards harmonization of the communication and computation costs.

The paper is organized as follows. In Section II we describe in detail our proposed cost framework. Section III reviews relevant distributed optimization preliminaries such as reformulations of (I.2), and the DGD method. The variant of DGD method with multiple consensus steps, which we call DGD^t, is introduced and analyzed in Section IV. In Section V, we describe the new NEAR-DGD^t and NEAR-DGD$^+$ methods, provide theoretical analysis of the variants and also present numerical results. We provide some final remarks and future directions of research in Section VI.

II. Adaptive Cost Framework

A typical iteration of a distributed optimization method consists of some local computation (typically gradient or Hessian evaluation) and neighborhood communication. While the amount of computation and communication per iteration differs from one algorithm to another, all iterations are counted blindly as equal in the traditional iteration counting metric. Moreover, as distributed algorithms are deployed in various contexts, the diverse range of scenarios calls for different ways to account for the cost (in terms of time, energy, or any other metric) of an algorithm. To illustrate this we discuss two motivating examples. Consider first the problem of controlling a swarm of battery powered robots, with low-energy computation modules onboard, connected via an energy-intense communication protocol. Since the robots have limited energy supply, communication steps can be very expensive, while longer task completion times may not be problematic. On the other hand, we consider solving a large-scale machine learning problem on a cluster of computers that are physically connected or with shared memory access. In this case, the cost of communication can be ignored (inexpensive in terms of time), while the computational cost (time) can be expensive depending on the size of the data set on each machine. Hence, a desirable metric should not only count the total number of communication and computation steps, it should also weigh the two appropriately according to different environments.

We propose an adaptive cost framework to evaluate the performance of distributed optimization methods which explicitly accounts for the cost of communication and computation, and can be customized depending on the specific application. In particular, we propose the following simple yet powerful metric

$$\text{Cost} = \#\text{Communications} \times c_c + \#\text{Computations} \times c_g, \tag{II.1}$$

where c_c and c_g are exogenous application-dependent parameters reflecting the costs of communication and computation, respectively. For instance, when energy is the most constraining resource of the environment, the parameters c_c and c_g would reflect the energy consumed per step of communication/computation. Similarly, when the runtime is of concern, the parameters would correspond to the time needed for a communication/computation step. This cost could also represent some combination of time and energy. In the battery powered robots example we would have $c_c > c_g$, and in the machine learning example we would have $c_c < c_g$. We note that if the cost of communication and computation of one iteration is 1, then to design an algorithm with minimal cost reduces to the standard problem of finding algorithm with the least iteration count.

III. Preliminaries

In this section, we introduce an equivalent compact reformulation of problem (I.2) and review the basics of the DGD method, both of which will be used to build our class of flexible algorithms.

A. Equivalent Reformulations

For compactness, we express problem (I.2) as

$$\begin{aligned} \min_{x_i \in \mathbb{R}^p} \quad & f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & (\mathbf{W} \otimes I_p)\mathbf{x} = \mathbf{x} \end{aligned} \tag{III.1}$$

where $\mathbf{x} \in \mathbb{R}^{np}$ is a concatenation of all local x_i's and $\mathbf{W}$ is a matrix of size $\mathbb{R}^{n \times n}$, i.e.,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{bmatrix}.$$

Matrix I_p is the identity matrix of dimension p, and the operator $\otimes$ denotes the Kronecker product operation, with $\mathbf{W} \otimes I_p \in \mathbb{R}^{np \times np}$. Moreover, matrix $\mathbf{W}$ has the following properties: it is symmetric, doubly-stochastic, with diagonal elements $w_{ii} > 0$ and off-diagonal elements $w_{ij} > 0$ $(i \neq j)$ if and only if i and j are neighbors in the underlying communication network. Matrix $\mathbf{W}$ is known as the *consensus matrix* and it has the property that $(\mathbf{W} \otimes I_p)\mathbf{x} = \mathbf{x}$ if and only if $x_i = x_j$ for all i and j in the connected network [13], i.e., problems (I.2) and (III.1) are equivalent. We also have that matrix $\mathbf{W}$ has exactly one eigenvalue equal to 1 and the rest of eigenvalues have absolute values strictly less than 1. We use

β, with $0 < \beta < 1$, to denote the second largest magnitude of the eigenvalues of $\mathbf{W}$. For the rest of the paper, we focus on developing methods to solve problem (III.1).

B. Distributed Gradient Descent

We now review the basic Distributed Gradient Descent (DGD) method [13], which is the building block for our later development of the DGD^t, NEAR-DGD^t and NEAR-DGD$^+$ methods. The DGD method is a first-order method for solving problem (III.1), where each agent updates its local estimate iteratively using a gradient based on local information and information exchanged with its neighbors in the network. The k^{th} iteration of the DGD method for any node i can be expressed as

$$x_{i,k+1} = \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} x_{j,k} - \alpha \nabla f_i(x_{i,k}), \qquad \forall i = 1, \dots, n,$$

where $x_{i,k}$ represents the local estimate of agent i at iteration k and the positive scalar α denotes the stepsize. Effectively, the i^{th} agent computes a weighted average of its and its neighbors local estimates, and takes a step in the negative gradient direction obtained using only local information. Equivalently, in the concatenated notation, the DGD method can be expressed as

$$\mathbf{x}_{k+1} = \mathbf{Z}\mathbf{x}_k - \alpha \nabla \mathbf{f}(\mathbf{x}_k) \tag{III.2}$$

where

$$\nabla \mathbf{f}(\mathbf{x}_k) = \begin{bmatrix} \nabla f(x_{1,k}) \\ \nabla f(x_{2,k}) \\ \vdots \\ \nabla f(x_{n,k}) \end{bmatrix} \in \mathbb{R}^{np}$$

and the matrix $\mathbf{Z} = \mathbf{W} \otimes I_p \in \mathbb{R}^{np \times np}$.

The DGD method can also be thought of as a gradient method with unit steplength on the following convex problem

$$\min_{\mathbf{x} \in \mathbb{R}^{np}} \frac{1}{2} \mathbf{x}^T (I - \mathbf{Z}) \mathbf{x} + \alpha \sum_{i=1}^{n} f_i(x_i). \tag{III.3}$$

The theoretical properties of the DGD method have been well established; see [12], [13], [63]. The convergence results are typically established under the following standard assumptions:

Assumption III.1. *Each local objective function f_i has L_i-Lipschitz continuous gradients.*

Assumption III.2. *The objective function h (I.1) is μ_h-strongly convex.*

Assumption III.3. *Each local objective function f_i is μ_i-strongly convex. Note, this Assumption implies Assumption III.2.*

These assumptions guarantee the existence of a unique optimal solution. Under Assumption III.1, the DGD method can be shown to converge at a sublinear rate with diminishing stepsize (decrease stepsize α as the algorithm evolves). The diminishing stepsize is effectively shrinking the penalty parameter α of problem (III.3) and thus DGD recovers a feasible and optimal solution of problem (III.1) in the limit. If we further assume the conditions in Assumption III.3, then with an appropriate constant stepsize, DGD converges at a linear rate to the optimal solution of (III.3), which is in a neighborhood of the optimal solution of (III.1) [63]. The limit point of DGD with constant stepsize is often infeasible for the equality constraint in problem (III.1).

Notation: For the rest of the paper, we follow the same notation as in this section. A boldface lower case letter indicates a concatenated vector, i.e., $\mathbf{v} \in \mathbb{R}^{np}$ represents the concatenation of local vectors $v_i \in \mathbb{R}^p$. Notation $\bar{v} \in \mathbb{R}^p$ denotes the average of all local vectors $v_i \in \mathbb{R}^p$, i.e., $\bar{v} = \frac{1}{n}\sum_{i=1}^{n} v_i$. The two subscripts $x_{i,k}$ indicate the agent index i and iteration count k.

IV. DGD^t: A Distributed Gradient Descent Variant

A close inspection of the DGD iterate update Eq. (III.2) reveals that the method performs a single round of communication and a single computation per iteration. However, a natural question is whether this is optimal or even necessary. Restating this question from a different angle: is there flexibility in creating a whole class of methods based on the components of the DGD method that perform different number of communication and computation steps depending on the application? Motivated by this question and our adaptive cost framework (Section II), we decompose and rearrange the communication and computation steps of DGD to construct more flexible algorithms. We present two improved classes of DGD-based algorithms in this and the following sections, which we call DGD^t and NEAR-DGD methods, respectively. We also provide answers to the following questions: (i) what the interpretation of these new methods are, and (ii) what theoretical guarantees can be established. For simplicity, we will focus on the constant stepsize implementations.

For the first class of algorithms, we consider scenarios in which communication is much cheaper than computation, as in the shared memory machine learning example. We note that a major drawback of the DGD algorithm is that to obtain a feasible solution we need to use diminishing stepsizes, which results in slow convergence speed. With constant stepsizes, the resulting solution of DGD is infeasible with respect to the equality constraint of problem (III.1). In order to improve the solution quality without sacrificing convergence speed, we propose to perform t consensus steps at each iteration, and consider the following constant stepsize iterate update equation,

$$\mathbf{x}_{k+1} = \mathbf{Z}^t \mathbf{x}_k - \alpha \nabla \mathbf{f}(\mathbf{x}_k), \qquad \mathbf{Z}^t = \mathbf{W}^t \otimes I_p, \tag{IV.1}$$

which we call the *DGD^t* method. The DGD^t method can be thought of as a gradient method with unit steplength on the following convex problem

$$\min_{\mathbf{x} \in \mathbb{R}^{np}} p_f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T (I - \mathbf{Z}^t) \mathbf{x} + \alpha \sum_{i=1}^{n} f_i(x_i). \tag{IV.2}$$

The intuition behind this method is that by increasing the number of consensus steps from 1 to t per iteration, the resulting solution should be closer to being feasible. Alternatively, we can view the DGD^t method as a DGD method with a different underlying graph (different weights in $\mathbf{W}$). As we will show next, the solution of DGD^t is indeed closer to being feasible compared to standard DGD. This is achieved at the cost of more communication steps per iteration. Also, unlike DGD, where the gradient computation and consensus can happen simultaneously, the t communication steps in DGD^t have to happen sequentially. This method can be desirable when communication is cheap, i.e., c_c is much smaller than c_g.

A. Convergence Analysis of DGD^t

We now provide a complete convergence analysis for the DGD^t method with constant stepsize. We should note again that DGD^t is a variant of DGD by replacing the weight matrix W by W^t. As such, our analysis follows a similar approach as in [63], and so for brevity we have omitted the proofs. The proofs can be found in [64].

For notational convenience, we introduce the following quantities that are used in the analysis

$$\bar{x}_k = \frac{1}{n}\sum_{i=1}^{n} x_{i,k}, \quad g_k = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_{i,k}),$$
$$\bar{g}_k = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\bar{x}_k). \tag{IV.3}$$

Vector $\bar{x}_k \in \mathbb{R}^p$ corresponds to the average of local estimates, vector $g_k \in \mathbb{R}^p$ represents the average of local gradients at the current local estimates and vector $\bar{g}_k \in \mathbb{R}^p$ indicates the average gradient at $\bar{x}_k$.

Lemma IV.1. ***(Bounded gradients)*** *Suppose Assumption III.1 holds, and let the steplength satisfy*

$$\alpha \leq \frac{1+\lambda_n(\boldsymbol{W}^t)}{L} \tag{IV.4}$$

where $\lambda_n(\boldsymbol{W}^t)$ is the smallest eigenvalue of $\boldsymbol{W}^t$ and $L = \max_i L_i$. Then, starting from $x_{i,0} = 0$ ($1 \leq i \leq n$), the sequence $x_{i,k}$ generated by the DGD^t method converges. In addition, we also have

$$\|\nabla \boldsymbol{f}(\boldsymbol{x}_k)\| \leq D = \sqrt{2L\left(\sum_{i=1}^{n}(f_i(0) - f_i^\star)\right)}$$

for all $k = 1, 2, \ldots$, where $f_i^\star = f_i(x_i^\star)$ and $x_i^\star = \arg\min_x f_i(x)$.

Proof. Note that the DGD^t iteration (IV.1) is equivalent to a gradient descent iteration, with unit steplength on the quadratic penalty function p_f (IV.2). We first show that the function p_f has $\left[\left(1-\lambda_n(\mathbf{W}^t)\right)+\alpha L\right]$-Lipschitz continuous gradients. By definition of p_f and the triangle inequality, we have

$$\begin{aligned}||\nabla p_f(u) - \nabla p_f(v)|| \leq &\left|\left|(I-\mathbf{Z}^t)(u-v)\right|\right| \\ &+ \alpha\left|\left|\sum_{i=1}^{n}\nabla f_i(u_i) - \sum_{i=1}^{n}\nabla f_i(v_i)\right|\right|.\end{aligned} \tag{IV.5}$$

By the Cauchy-Schwartz inequality, the first term satisfies

$$\begin{aligned}\left|\left|(I-\mathbf{Z}^t)(u-v)\right|\right| &\leq \left|\left|I-\mathbf{Z}^t\right|\right| ||u-v|| \\ &= \left(1-\lambda_n(\mathbf{W}^t)\right)||u-v||,\end{aligned}$$

where the last inequality follows from the fact that all eigenvalues of the matrix $\mathbf{W}$ lie in the interval $(-1, 1]$. The second term in Eq. (IV.5) satisfies

$$\alpha\left|\left|\sum_{i=1}^{n}\nabla f_i(u_i) - \sum_{i=1}^{n}\nabla f_i(v_i)\right|\right| \leq \alpha L\,||u-v||,$$

due to Assumption III.1. Thus, we have that the function p_f has Lipschitz continuous gradients with

$$L_{p_f} \leq \left(1-\lambda_n(\mathbf{W}^t)\right) + \alpha L.$$

From the classical analysis of gradient descent [12], we know that these iterates will converge with unit stepsize if $1 \leq \frac{2}{L_{p_f}}$, where L_{p_f} is the Lipschitz constant of the gradients of p_f. Since $\alpha \leq \frac{1+\lambda_n(\mathbf{W}^t)}{L}$ from Eq. (IV.4), we have

$$\begin{aligned}L_{p_f} &\leq \left(1-\lambda_n(\mathbf{W}^t)\right) + \alpha L \\ &\leq \left(1-\lambda_n(\mathbf{W}^t)\right) + \frac{1+\lambda_n(\mathbf{W}^t)}{L}L \leq 2.\end{aligned}$$

Hence when the condition in Eq. (IV.4) is satisfied, the iterates $\mathbf{x}_k$ will converge which implies that the individual iterates $x_{i,k}$ converge.

We now show the bound on the gradients. Since the function values obtained in the gradient descent method are non-increasing and $I - \mathbf{W}^t$ is a positive semi-definite matrix, we have,

$$\begin{aligned}\sum_{i=1}^{n} f_i(x_{i,k}) \leq \frac{1}{\alpha}p_f(\mathbf{x}_k) \leq \frac{1}{\alpha}p_f(\mathbf{x}_{k-1}) \leq \ldots \\ \cdots \leq \frac{1}{\alpha}p_f(\mathbf{x}_0) = \sum_{i=1}^{n} f_i(0).\end{aligned} \tag{IV.6}$$

By Theorem 2.1.5 in [66], any convex function ϕ with $L-$Lipschitz gradient satisfies

$$\phi(x) + \nabla\phi(x)^T(y-x) + \frac{1}{2L}||\nabla\phi(x) - \nabla\phi(y)||^2 \leq \phi(y)$$

for all x, y in its domain. We apply this relation to each of f_i at respective $x_i^\star$, and have

$$f_i(x_i^\star) + \frac{1}{2L_i}||\nabla f_i(x)||^2 \leq f_i(x) \tag{IV.7}$$

for all x in $\mathbb{R}^p$.

Finally, we can bound $\|\nabla \mathbf{f}(\mathbf{x}_k)\|^2$ by

$$\|\nabla \mathbf{f}(\mathbf{x}_k)\|^2 = \sum_{i=1}^{n} \|\nabla f_i(x_{i,k})\|^2 \le \sum_{i=1}^{n} 2L_i \left(f_i(x_{i,k}) - f_i^\star\right) \le 2L \left(\sum_{i=1}^{n} \left(f_i(0) - f_i^\star\right)\right).$$

where the first inequality follows from Eq. (IV.7) and the second inequality uses the definition of L and Eq. (IV.6). □

Lemma IV.1 shows that the iterates produced by the DGD^t method converge and have bounded gradients. A different bound can be shown if $x_{i,0} \neq 0$ for all i. For convenience, we assume that $x_{i,0} = 0$, for all i for the rest of this section.

Lemma IV.2. ***(Bounded deviation from mean)*** *If Assumptions III.1-III.2 hold. Then, starting from $x_{i,0} = 0$, the total deviation from the mean is bounded, namely,*

$$\|x_{i,k} - \bar{x}_k\| \le \frac{\alpha D}{1-\beta^t},$$

and

$$\|\nabla f_i(x_{i,k}) - \nabla f_i(\bar{x}_k)\| \le \frac{\alpha D L_i}{1-\beta^t}, \tag{IV.8}$$

$$\|g_k - \bar{g}_k\| \le \frac{\alpha D L}{1-\beta^t}, \tag{IV.9}$$

for all $k = 1, 2, \ldots$ and $1 \le i \le n$.

Proof. By iteratively applying the DGD^t iteration (IV.1) and the definition of $\mathbf{x}$, we obtain

$$\mathbf{x}_k = -\alpha \sum_{s=0}^{k-1} \left(\mathbf{W}^{t(k-1-s)} \otimes I\right) \nabla \mathbf{f}(x_s).$$

Let $\bar{\mathbf{x}}_k = [\bar{x}_k; \bar{x}_k; \ldots; \bar{x}_k] \in \mathbb{R}^{np}$, it follows that

$$\bar{\mathbf{x}}_k = \frac{1}{n}\left((1_n 1_n^T) \otimes I\right) \mathbf{x}_k.$$

As a result,

$$\begin{aligned}
\|x_{i,k} - \bar{x}_k\| &\le \|\mathbf{x}_k - \bar{\mathbf{x}}_k\| \\
&= \left\| \mathbf{x}_k - \frac{1}{n}\left((1_n 1_n^T) \otimes I\right) \mathbf{x}_k \right\| \\
&= \Bigg\| -\alpha \sum_{s=0}^{k-1} \left(\mathbf{W}^{t(k-1-s)} \otimes I\right) \nabla \mathbf{f}(x_s) \\
&\qquad + \alpha \sum_{s=0}^{k-1} \frac{1}{n} \left((1_n 1_n^T \mathbf{W}^{t(k-1-s)}) \otimes I\right) \nabla \mathbf{f}(x_s) \Bigg\| \\
&= \Bigg\| -\alpha \sum_{s=0}^{k-1} \left(\mathbf{W}^{t(k-1-s)} \otimes I\right) \nabla \mathbf{f}(x_s) \\
&\qquad + \alpha \sum_{s=0}^{k-1} \frac{1}{n} \left((1_n 1_n^T) \otimes I\right) \nabla \mathbf{f}(x_s) \Bigg\|,
\end{aligned}$$

where the third equality holds since $\mathbf{W}^t$ is doubly-stochastic, which is a direct consequence of $\mathbf{W}$ being doubly-stochastic. Thus we have,

$$\begin{aligned}
\|x_{i,k} - \bar{x}_k\| &\le \Bigg\| -\alpha \sum_{s=0}^{k-1} \left(\mathbf{W}^{t(k-1-s)} \otimes I\right) \nabla \mathbf{f}(x_s) \\
&\qquad + \alpha \sum_{s=0}^{k-1} \frac{1}{n} \left((1_n 1_n^T) \otimes I\right) \nabla \mathbf{f}(x_s) \Bigg\| \\
&= \alpha \left\| \sum_{s=0}^{k-1} \left((\mathbf{W}^{t(k-1-s)} - \frac{1}{n} 1_n 1_n^T) \otimes I \right) \nabla \mathbf{f}(x_s) \right\| \\
&\le \alpha \sum_{s=0}^{k-1} \left\| \mathbf{W}^{t(k-1-s)} - \frac{1}{n} 1_n 1_n^T \right\| \|\nabla \mathbf{f}(x_s)\| \\
&= \alpha \sum_{s=0}^{k-1} \beta^{t(k-1-s)} \|\nabla \mathbf{f}(x_s)\|,
\end{aligned}$$

where the inequality is due to Cauchy-Schwartz, and the last equality follows from the definition of β, since the matrix $\frac{1}{n} 1_n 1_n^T$ is the projection of $\mathbf{W}$ onto the eigenspace associated with the eigenvalue equal to 1. Using Lemma IV.1 and the fact that $\beta < 1$, it follows that

$$\begin{aligned}
\|x_{i,k} - \bar{x}_k\| &\le \alpha \sum_{s=0}^{k-1} \beta^{t(k-1-s)} \|\nabla \mathbf{f}(x_s)\| \\
&\le \alpha D \sum_{s=0}^{k-1} \beta^{t(k-1-s)} \le \frac{\alpha D}{1-\beta^t}.
\end{aligned} \tag{IV.10}$$

The result (IV.8) is a direct consequence of (IV.10) and the Lipschitz continuity of the individual gradients (Assumption III.1). For the second result (IV.9), we have

$$\begin{aligned}
\|g_k - \bar{g}_k\| &= \left\| \frac{1}{n} \sum_{i=1}^{n} \left(\nabla f_i(x_{i,k}) - \nabla f_i(\bar{x}_k)\right) \right\| \\
&\le \frac{1}{n} \sum_{i=1}^{n} L_i \|x_{i,k} - \bar{x}_k\| \le \frac{\alpha D L}{1-\beta^t}.
\end{aligned}$$

□

Lemma IV.2 shows that the distance between the local iterates and the average is bounded. As a consequence of this, the deviation in the gradients is also bounded.

We now look at the optimization error. We observe that due to the doubly-stochastic nature of $\mathbf{W}$,

$$\begin{aligned}
\bar{\mathbf{x}}_{k+1} &= \frac{1}{n}\left((1_n 1_n^T) \otimes I\right) \mathbf{x}_{k+1} \\
&= \frac{1}{n}\left((1_n 1_n^T) \otimes I\right) \left((\mathbf{W}^t \otimes I)\mathbf{x}_k - \alpha \nabla \mathbf{f}(\mathbf{x}_k)\right) \\
&= \frac{1}{n}\left((1_n 1_n^T \mathbf{W}^t) \otimes I\right) \mathbf{x}_k - \frac{\alpha}{n}\left((1_n 1_n^T) \otimes I\right) \nabla \mathbf{f}(\mathbf{x}_k) \\
&= [\bar{x}_k - \alpha g_k; \bar{x}_k - \alpha g_k; \ldots; \bar{x}_k - \alpha g_k].
\end{aligned}$$

Thus we have

$$\bar{x}_{k+1} = \bar{x}_k - \alpha g_k. \tag{IV.11}$$

Recall that g_k is the average of gradients at the current local estimates [cf. Eq. (IV-A)] and thus the above equation can be viewed as an inexact gradient descent step for the problem

$$\min_{x\in\mathbb{R}^p} \bar{f}(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x), \tag{IV.12}$$

where $\bar{g}_k$ is the exact gradient (at the average of the local estimates). Consequently, if h has L_h-Lipschitz continuous gradients, and is μ_h-strongly convex, then it can be shown that the function $\bar{f}$ has $L_{\bar{f}}$-Lipschitz continuous gradients and is $\mu_{\bar{f}}$ strongly convex with

$$L_{\bar{f}} = \frac{1}{n}\sum_{i=1}^{n} L_i = \frac{1}{n}L_h, \quad \mu_{\bar{f}} = \frac{1}{n}\sum_{i=1}^{n}\mu_i = \frac{1}{n}\mu_h.$$

Based on the above observations, we bound the distance to the optimal solution.

Theorem IV.3. ***(Bounded distance to optimal solution)*** *Suppose Assumptions III.1-III.2 hold, and let the steplength satisfy*

$$\alpha \le \min\left\{\frac{1+\lambda_n(\boldsymbol{W}^t)}{L}, c_4\right\}$$

where $\lambda_n(\boldsymbol{W}^t)$ is the smallest eigenvalue of $\boldsymbol{W}^t$, $L = \max_i L_i$ and $c_4 = \frac{2}{\mu_{\bar{f}}+L_{\bar{f}}}$. Then, starting from $x_{i,0} = 0$ ($1 \le i \le n$), for all $k = 0, 1, 2, \ldots$

$$\|\bar{x}_{k+1} - x^\star\|^2 \le c_1^2\|\bar{x}_k - x^\star\|^2 + \frac{c_3^2}{(1-\beta^t)^2},$$

where

$$c_1^2 = 1 - \alpha c_2 + \alpha\delta - \alpha^2\delta c_2, \quad c_2 = \frac{2\mu_{\bar{f}}L_{\bar{f}}}{\mu_{\bar{f}}+L_{\bar{f}}},$$

$$c_3^2 = \alpha^3(\alpha+\delta^{-1})L^2D^2, \quad D = \sqrt{2L\left(\sum_{i=1}^{n} f_i(0) - f^\star\right)},$$

$x^\star$ is the optimal solution of (III.1) *and $\delta > 0$. In particular, if we set $\delta = \frac{c_2}{2(1-\alpha c_2)}$ such that $c_1 = \sqrt{1-\frac{\alpha c_2}{2}} \in (0,1)$, then for $k = 0, 1, 2, \ldots$*

$$\|\bar{x}_k - x^\star\| \le c_1^k\|\bar{x}_0 - x^\star\| + \mathcal{O}\left(\frac{\alpha}{1-\beta^t}\right).$$

Proof. Using the definitions of the $\bar{x}_k$, g_k and (IV.11), we have

$$\begin{aligned}
\|\bar{x}_{k+1} - x^\star\|^2 &= \|\bar{x}_k - x^\star - \alpha g_k\|^2 \\
&= \|\bar{x}_k - x^\star - \alpha\bar{g}_k + \alpha(\bar{g}_k - g_k)\|^2 \\
&= \|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2 + \alpha^2\|\bar{g}_k - g_k\|^2 \\
&\quad + 2\alpha(\bar{g}_k - g_k)^T(\bar{x}_k - x^\star - \alpha\bar{g}_k) \\
&\le (1+\alpha\delta)\|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2 \\
&\quad + \alpha(\alpha+\delta^{-1})\|\bar{g}_k - g_k\|^2,
\end{aligned} \tag{IV.13}$$

where the last inequality follows from the fact that for any vectors a and b, $\pm 2a^Tb \le \delta^{-1}\|a\|^2 + \delta\|b\|^2$, for $\delta > 0$.

We now bound the quantity $\|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2$,

$$\begin{aligned}
&\|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2 \\
&\quad = \|\bar{x}_k - x^\star\|^2 + \alpha^2\|\bar{g}_k\|^2 - 2(\bar{x}_k - x^\star)^T(\alpha\bar{g}_k) \\
&\quad \le \|\bar{x}_k - x^\star\|^2 + \alpha^2\|\bar{g}_k\|^2 - \alpha c_4\|\bar{g}_k\|^2 - \alpha c_2\|\bar{x}_k - x^\star\|^2 \\
&\quad = (1-\alpha c_2)\|\bar{x}_k - x^\star\|^2 + \alpha(\alpha - c_4)\|\bar{g}_k\|^2 \\
&\quad \le (1-\alpha c_2)\|\bar{x}_k - x^\star\|^2,
\end{aligned} \tag{IV.14}$$

where the first inequality follows from [66, Theorem 2.1.12, Chapter 2], and in the last inequality we dropped the term $\alpha(\alpha-c_4)\|\bar{g}_k\|^2$, since $\alpha \le c_4$ and $\alpha(\alpha-c_4)\|\bar{g}_k\|^2 \le 0$. By combining (IV.13) and (IV.14), and using (IV.9), we obtain

$$\begin{aligned}
\|\bar{x}_{k+1} - x^\star\|^2 &\le (1+\alpha\delta)(1-\alpha c_2)\|\bar{x}_k - x^\star\|^2 \\
&\quad + \alpha^3(\alpha+\delta^{-1})\frac{L^2D^2}{(1-\beta^t)^2}.
\end{aligned} \tag{IV.15}$$

Combining $c_4 < \frac{1}{c_2}$, and $\alpha \le c_4$, we have $(1+\alpha\delta)(1-\alpha c_2) > 0$. Now, using the definitions of c_1 and c_3, and by recursive application of (IV.15), we have Now, using the definitions of c_1 and c_3, and by recursive application of (IV.15), we have

$$\|\bar{x}_k - x^\star\|^2 \le c_1^{2k}\|\bar{x}_0 - x^\star\|^2 + \frac{c_3^2}{(1-c_1^2)(1-\beta^t)^2},$$

and so

$$\|\bar{x}_k - x^\star\| \le c_1^k\|\bar{x}_0 - x^\star\| + \frac{c_3}{\sqrt{1-c_1^2}(1-\beta^t)}.$$

If $\delta = \frac{c_2}{2(1-\alpha c_2)} > 0$, then

$$c_1^2 = 1 - \frac{\alpha c_2}{2} < 1$$

and

$$\frac{c_3}{\sqrt{1-c_1^2}(1-\beta^t)} = \mathcal{O}\left(\frac{\alpha}{1-\beta^t}\right).$$

which completes the proof. □

Theorem IV.3 shows that the average of the iterates generated by the DGD^t method converges to a neighborhood of the solution that is defined by the steplength, the second largest eigenvalue of $\mathbf{W}$ and the number of consensus steps.

Corollary IV.4. ***(Local agent convergence)*** *Suppose Assumptions III.1-III.2 hold, and let the steplength satisfy*

$$\alpha \le \min\left\{\frac{1+\lambda_n(\boldsymbol{W}^t)}{L}, c_4\right\}.$$

where $L = \max_i L_i$ and $c_4 = \frac{2}{\mu_{\bar{f}}+L_{\bar{f}}}$. Then, starting from $x_{i,0} = 0$ ($1 \le i \le n$), for $k = 0, 1, 2, \ldots$

$$\|x_{i,k} - x^\star\| \le c_1^k\|x^\star\| + \frac{c_3}{\sqrt{1-c_1^2}(1-\beta^t)} + \frac{\alpha D}{1-\beta^t}.$$

Proof. Using the results from Lemma IV.2, Theorem IV.3 and the definition of c_4, we have

$$\begin{aligned}
\|x_{i,k} - x^\star\| &\le \|\bar{x}_k - x^\star\| + \|x_{i,k} - \bar{x}_k\| \\
&\le c_1^k\|x^\star\| + \frac{c_3}{\sqrt{1-c_1^2}(1-\beta^t)} + \frac{\alpha D}{1-\beta^t}.
\end{aligned}$$

□

The results of Theorem IV.3 and Corollary IV.4 are similar in nature to the results for the standard DGD method. More specifically, and not surprisingly, the DGD^t method converges to a neighborhood of the optimal solution of problem (III.1) when a constant steplength is employed. Compared to the DGD method, DGD^t converges to a better (smaller) neighborhood which is captured in the analysis

$$\mathcal{O}\left(\frac{1}{(1-\beta)^2}\right) \quad \text{v.s.} \quad \mathcal{O}\left(\frac{1}{(1-\beta^t)^2}\right).$$

Performing multiple communication steps is beneficial as it improves the neighborhood of convergence, however, multiple consensus alone cannot guarantee convergence of the DGD^t method to the solution. Namely, the error term that appears in Theorem IV.3 cannot be eliminated by simply performing multiple rounds of communication, since $\lim_{t\to\infty} \frac{1}{(1-\beta^t)^2} = 1 \neq 0$.

The results presented in Lemmas IV.1 and IV.2, Theorem IV.3 and Corollary IV.4 are not surprising, nevertheless, the results clearly illustrate the power of performing multiple rounds of communication steps.

B. Numerical Results for DGD^t

In this section, we provide some empirical evidence to support the theoretical results, and to ascertain that, in practice, the benefits of performing multiple consensus steps are realized. We chose a simple quadratic problem of the form

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x^T A_i x + b_i^T x,$$

where each node $i = \{1, ..., n\}$ has local information $A_i \in \mathbb{R}^{p\times p}$ and $b_i \in \mathbb{R}^p$. The problem was constructed as described in [65]; we chose a dimension size $p = 10$ and the condition number ($\kappa = \frac{L_f}{\mu_f}$) was set to 10^2. For this experiment, the number of agents in the network (n) was 10, and we considered a 4-cyclic graph topology (i.e., each node is connected to its 4 immediate neighbors).

We investigated the performance of four different variants of DGD^t, with $t = 1$ (standard DGD) and $t = 2, 5, 10$. Figure 1 illustrates the performance (relative error, $\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2$, with $x^\star \neq 0$) of the four methods in terms of: (i) iterations, (ii) cost (as described in Section II, with $c_c = c_g = 1$), (iii) number of gradient evaluations, and (iv) number of communications. Each of the four methods has a steplength parameter that needs to be tuned in order to achieve the best performance. We manually tuned the steplength for each method independently. We mention in passing that similar behavior was observed when we measured the performance of the methods in terms of consensus error $\left(\frac{1}{n}\sum_{i=1}^{n}\|x_{i,k} - x^\star\|^2/\|x^\star\|^2\right)$.

Figure 1 clearly illustrates what was predicted by the theory. Firstly, it shows that performing multiple rounds of communication improves the neighborhood of convergence. Secondly, it shows that there is a *diminishing returns* effect of the number of communication rounds on the performance. Namely, the neighborhood improves substantially when going from 1 consensus step to 2 consensus steps, however, going from 5 consensus steps to 10 consensus steps has a much smaller effect. It is interesting to investigate that the performance of the methods in terms of the cost. Given a fixed cost budget, (e.g., 10^4) it appears that only one method (blue: 2 consensus steps per gradient step) is competitive with and better than the baseline DGD method. Again, the reason for this is the marginal returns effect of performing many more consensus steps per iteration. We should, of course, mention that our observation about the performance of the methods in terms of the cost are highly dependent on the cost structure that we chose for these experiments ($c_c = c_g = 1$). In Section V-B, we show results for different cost structures.

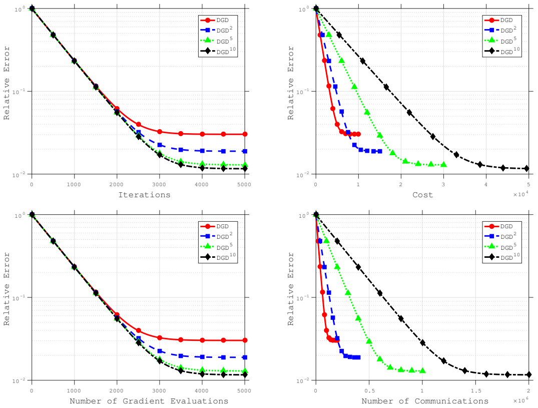


Fig. 1. Performance of DGD, DGD^2, DGD^5 and DGD^{10} measured in terms of relative error $\left(\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2\right)$ with respect to: (i) number of iterations, (ii) cost, (iii) number of gradient evaluations, and (iv) number of communications, on a quadratic problem ($n = 10$, $p = 10$, $\kappa = 10^2$).

Before proceeding forward we make one final remark. We tested the performance of the four methods on quadratic problems with different characteristic and different graph sizes. While the absolute performance of the methods changed, the relative performance of the methods did not. More specifically, with the cost structure $c_c = c_g = 1$, similar behavior as that displayed in Figure 1 was observed.

V. NEAR-DGD

Motivated by the improved results of the DGD^t method and the power of performing multiple consensus steps, we ask the question whether a first-order distributed method can achieve exact convergence to the optimal solution of problem (III.1) by simply performing multiple rounds of communication. The results from the previous section suggest that a simple modification as in DGD^t is not sufficient. To construct new algorithms, we observe that each iteration of DGD [cf. Eq. (III.2)] consists of two operators,

- Consensus Operator: $\mathcal{W}[\mathbf{x}] = \mathbf{Z}\mathbf{x}$,
- Gradient Operator: $\mathcal{T}[\mathbf{x}] = \mathbf{x} - \alpha\nabla\mathbf{f}(\mathbf{x})$.

Using these operators the DGD method can be expressed as

$$\mathbf{x}_{k+1} = (\mathcal{T} - I + \mathcal{W})[\mathbf{x}_k] = \mathbf{Z}\mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k). \qquad \text{(V.1)}$$

We can view the consensus and gradient steps as two separable operations. This enables a decomposition of the computation and communication operations and allows for flexible customization in view of our new cost framework. An alternative way to combine these two operators is by nesting them. Simply alternating between these two operations leads to our first new method, which we call the NEAR-DGD method – **N**ested **E**xact **A**lternating **R**ecursion method. The τ^{th} iterate of the method can be expressed as,

$$\mathbf{x}_\tau = [\mathcal{W}[\mathcal{T}[\cdots \overbrace{[\mathcal{W}\,\underbrace{[\mathcal{T}[\mathcal{W}[}_{y_k}}^{x_k}\mathcal{T}[\cdots[\mathcal{W}[\mathcal{T}[\mathbf{x}_0]]]\cdots]]]]]\cdots]]]. \tag{V.2}$$

Each iteration of NEAR-DGD involves the same amount of communication and computation as the standard DGD method. The main difference is that the gradient is now computed at the variable after the consensus step, i.e., the counterpart of Eq. (V.1) is given by

$$\mathbf{x}_{k+1} = \mathbf{Z}\mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{Z}\mathbf{x}_k).$$

Compared to the original DGD method (III.2), where the gradient step and communication step can be done in parallel, newer information is used to compute the gradient step in NEAR-DGD, and thus it has two inherently sequential steps. Alternatively, we can view the NEAR-DGD method as a method that produces an intermediate iterate $\mathbf{y}_k$ after the gradient step ($\mathcal{T}$), and the iterate $\mathbf{x}_k$ after the consensus step ($\mathcal{W}$). The iterates $\mathbf{x}_k$ and $\mathbf{y}_k$ can be expressed as

$$\mathbf{x}_k = \mathcal{W}[\mathbf{y}_k] = \mathbf{Z}\mathbf{y}_k$$
$$\mathbf{y}_{k+1} = \mathcal{T}[\mathbf{x}_k] = \mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k).$$

We assume here that the local iterates $x_{i,0}$ are initialized to be equal[1]. As a result, we can start with either the $\mathcal{T}$ or $\mathcal{W}$ operation and have the same expression as Eq. (V.2), since an initial consensus step would result in the same iterate ($\mathbf{x}_0 = \mathbf{Z}\mathbf{y}_0 = \mathbf{y}_0$).

As with the DGD method, we propose a variant of the NEAR-DGD method that performs multiple consensus steps per gradient step. This method–which we call the NEAR-DGD^t– can be expressed as

$$\mathbf{x}_\tau = [\mathcal{W}^t[\mathcal{T}[\cdots \overbrace{[\mathcal{W}^t\,\underbrace{[\mathcal{T}[\mathcal{W}^t[}_{y_k}}^{x_k}\mathcal{T}[\cdots[\mathcal{W}^t[\mathcal{T}[\mathbf{x}_0]]]\cdots]]]]]\cdots]]],$$

where $\mathcal{W}^t[x]$ denotes t nested consensus operations (steps),

$$\mathcal{W}^t[x] = \underbrace{\mathcal{W}[\cdots[\mathcal{W}[\mathcal{W}}_{\text{t operations}}[x]]]\cdots].$$

In terms of the iterates $\mathbf{x}_k$ and $\mathbf{y}_k$ the NEAR-DGD^t method can be expressed as

$$\mathbf{x}_k = \mathcal{W}^t[\mathbf{y}_k] = \mathbf{Z}^t\mathbf{y}_k \tag{V.3}$$
$$\mathbf{y}_{k+1} = \mathcal{T}[\mathbf{x}_k] = \mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k). \tag{V.4}$$

The NEAR-DGD method is a special case of the NEAR-DGD^t method, with $t = 1$. Given the flexibility in designing algorithms, we note that the number of consensus steps does not have to stay constant throughout the algorithm, hence we also propose and analyze the NEAR-DGD$^+$ method with time-varying consensus steps,

$$\mathbf{x}_\tau = [\mathcal{W}^{t(\tau)}[\mathcal{T}[\cdots \overbrace{[\mathcal{W}^{t(k)}\,\underbrace{[\mathcal{T}[\mathcal{W}^{t(k-1)}[}_{y_k}}^{x_k}\mathcal{T}[\cdots$$
$$\cdots[\mathcal{W}^{t(2)}[\mathcal{T}[\mathcal{W}^{t(1)}[\mathcal{T}[\mathbf{x}_0]]]]]\cdots]]]]]\cdots]]],$$

where $\mathcal{W}^{t(k)}[x]$ denotes $t(k)$ nested consensus operations (steps). In terms of the iterates $\mathbf{x}_k$ and $\mathbf{y}_k$ the NEAR-DGD$^+$ method can be expressed as

$$\mathbf{x}_k = \mathcal{W}^{t(k)}[\mathbf{y}_k] = \mathbf{Z}^{t(k)}\mathbf{y}_k \tag{V.5}$$
$$\mathbf{y}_{k+1} = \mathcal{T}[\mathbf{x}_k] = \mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k). \tag{V.6}$$

where at every iteration we change/increase the number of consensus steps ($t(k)$).

A. Convergence Analysis of NEAR-DGD^t and NEAR-DGD$^+$

We first analyze the NEAR-DGD^t method, and then illustrate the convergence properties of the NEAR-DGD$^+$ method. We adopt the same assumptions (III.1 & III.3) as in Section III-B, and similarly to Section III-B, we define the average of $y_{i,k}$ as

$$\bar{y}_k = \frac{1}{n}\sum_{i=1}^{n} y_{i,k}.$$

We note that the gradient step in the NEAR-DGD method, and by extension the NEAR-DGD^t and NEAR-DGD$^+$ methods, can be viewed as a single step gradient iteration at the point x_k on the following unconstrained problem

$$\min_{x_i\in\mathbb{R}^p} \quad \sum_{i=1}^{n} f_i(x_i). \tag{V.7}$$

We use this observation to bound the iterates $\mathbf{x}_k$ and $\mathbf{y}_k$.

Lemma V.1. ***(Bounded iterates)*** *Suppose Assumptions III.1 & III.3 hold, and let the steplength satisfy*

$$\alpha < \frac{1}{L}$$

where $L = \max_i L_i$. Then, starting from $x_{i,0} = s_0$ or $y_{i,0} = s_0$ ($1 \le i \le n$), the iterates generated by the NEAR-DGD^t method (V.3)-(V.4) *are bounded, namely,*

$$\|\boldsymbol{x}_k\| \le D, \quad \|\boldsymbol{y}_k\| \le D$$

for all $k = 1, 2, \ldots$, where $D = \|\boldsymbol{y}_0 - \boldsymbol{u}^\star\| + \frac{\nu+4}{\nu}\|\boldsymbol{u}^\star\|$, $\boldsymbol{u}^\star = [u_1^\star; u_2^\star; ...; u_n^\star] \in \mathbb{R}^{np}$, $u_i^\star = \arg\min_{u_i} f_i(u_i)$, $\boldsymbol{u}^\star$ is the optimal solution of (V.7)*, $\nu = 2\alpha\gamma$, $\gamma = \min_i \gamma_i$ and $\gamma_i = \frac{\mu_i L_i}{\mu_i + L_i}$.*

Proof. Using standard results for the gradient descent method [66, Theorem 2.1.15, Chapter 2], and noting that $\alpha < \frac{1}{L} < \frac{2}{\mu_i + L_i}$, which is the necessary condition on the steplength, we have for any $i \in \{1, 2, ..., n\}$

$$\|x_{i,k} - \alpha\nabla f_i(x_{i,k}) - u_i{}^\star\| \le \sqrt{1 - 2\alpha\gamma_i}\|x_{i,k} - u_i{}^\star\|.$$

[1] With slightly more complex notation and algebra, we can show that similar results hold for either $[\mathcal{T}[\mathcal{W}[x]]$ or $[\mathcal{W}[\mathcal{T}[x]]$, in the case where the agents initialize at different points.

From this, we have,

$$\begin{aligned}\|\mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k) - \mathbf{u}^\star\| &= \sqrt{\sum_{i=1}^{n} \|x_{i,k} - \alpha\nabla f_i(x_{i,k}) - u_i{}^\star\|^2} \\ &\le \sqrt{\sum_{i=1}^{n}(1-2\alpha\gamma_i)\|x_{i,k} - u_i{}^\star\|^2} \\ &\le \sqrt{(1-\nu)}\|\mathbf{x}_k - \mathbf{u}^\star\|. \end{aligned} \tag{V.8}$$

where the last inequality follows from the definition of ν.

Using the definitions of ν, $\mathbf{y}_{k+1}$ and Eq. (V.8), we have

$$\begin{aligned}\|\mathbf{y}_{k+1} - \mathbf{u}^\star\| &= \|\mathbf{x}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k) - \mathbf{u}^\star\| \\ &\le \sqrt{(1-\nu)}\|\mathbf{x}_k - \mathbf{u}^\star\| \\ &= \sqrt{(1-\nu)}\|\mathbf{Z}^t\mathbf{y}_k - \mathbf{u}^\star\| \\ &\le \sqrt{(1-\nu)}\left[\|\mathbf{Z}^t\|\|\mathbf{y}_k - \mathbf{u}^\star\| + \|I - \mathbf{Z}^t\|\|\mathbf{u}^\star\|\right].\end{aligned}$$

The eigenvalues of matrix $\mathbf{Z}^t$ are the same as those of matrix $\mathbf{W}^t$. The spectrum property of $\mathbf{W}$ guarantees that the magnitude of each eigenvalue is upper bounded by 1. Hence $\|\mathbf{Z}\| \le 1$ and $\|I - \mathbf{Z}^t\| \le 2$ for all t. Hence the above relation implies that

$$\|\mathbf{y}_{k+1} - \mathbf{u}^\star\| \le \sqrt{(1-\nu)}\|\mathbf{y}_k - \mathbf{u}^\star\| + 2\sqrt{(1-\nu)}\|\mathbf{u}^\star\|.$$

Recursive application of the above relation gives,

$$\begin{aligned}&\|\mathbf{y}_{k+1} - \mathbf{u}^\star\| \\ &\le (1-\nu)^{(k+1)/2}\|\mathbf{y}_0 - \mathbf{u}^\star\| + 2\sum_{j=0}^{k}(1-\nu)^{(j+1)/2}\|\mathbf{u}^\star\| \\ &\le \|\mathbf{y}_0 - \mathbf{u}^\star\| + \frac{2\sqrt{1-\nu}}{1-\sqrt{1-\nu}}\|\mathbf{u}^\star\| \\ &\le \|\mathbf{y}_0 - \mathbf{u}^\star\| + \frac{4}{\nu}\|\mathbf{u}^\star\|.\end{aligned}$$

Thus, we bound the iterate as

$$\begin{aligned}\|\mathbf{y}_{k+1}\| &\le \|\mathbf{y}_{k+1} - \mathbf{u}^\star\| + \|\mathbf{u}^\star\| \\ &\le \|\mathbf{y}_0 - \mathbf{u}^\star\| + \frac{\nu+4}{\nu}\|\mathbf{u}^\star\|.\end{aligned}$$

We now show that the same result is true for the iterates $\mathbf{x}_k$. Using the definition of $\mathbf{x}_k$ Eq. (V.3)

$$\begin{aligned}\|\mathbf{x}_{k+1}\| &= \|\mathbf{Z}^t\mathbf{y}_{k+1}\| \\ &\le \|\mathbf{Z}^t\|\|\mathbf{y}_{k+1}\| \\ &\le \|\mathbf{y}_{k+1}\| \\ &\le D,\end{aligned}$$

which completes the proof. □

Lemma V.1 shows that the iterates generated by the NEAR-DGD^t method are bounded. Since eigenvalues of Z^t and $I - Z^t$ are bounded above by 1 and 2, for any t, respectively, the same analysis can be used to show that the iterates generated by the NEAR-DGD$^+$ method are also bounded.

Lemma V.2. ***(Bounded deviation from mean)*** *If Assumptions III.1 & III.3 hold. Then, starting from $x_{i,0} = s_0$ or $y_{i,0} = s_0$ $(1 \le i \le n)$, the total deviation of each agent's estimate $(x_{i,k})$ from the mean is bounded, namely,*

$$\|x_{i,k} - \bar{x}_k\| \le \beta^t D \tag{V.9}$$

and

$$\|\nabla f_i(x_{i,k}) - \nabla f_i(\bar{x}_k)\| \le \beta^t D L_i \tag{V.10}$$

$$\|g_k - \bar{g}_k\| \le \beta^t D L \tag{V.11}$$

for all $k = 1, 2, \dots$ and $1 \le i \le n$. Moreover, the total deviation of the local iterates $y_{i,k}$ is also bounded,

$$\|y_{i,k} - \bar{y}_k\| \le \beta^t D + 2D. \tag{V.12}$$

Proof. Consider,

$$\begin{aligned}\|x_{i,k} - \bar{x}_k\| &= \|x_{i,k} - \bar{y}_k\| \\ &\le \left\|\mathbf{x}_k - \frac{1}{n}\left((1_n 1_n^T) \otimes I\right)\mathbf{y}_k\right\| \\ &= \left\|(\mathbf{W}^t \otimes I)\mathbf{y}_k - \frac{1}{n}\left((1_n 1_n^T) \otimes I\right)\mathbf{y}_k\right\| \\ &\le \left\|\left(\mathbf{W}^t - \frac{1}{n}\left((1_n 1_n^T)\right) \otimes I\right)\right\| \|\mathbf{y}_k\| \\ &\le \beta^t\|\mathbf{y}_k\| \le \beta^t D,\end{aligned}$$

where the first equality is due to the fact that $\bar{x}_k = \mathbf{Z}^t\bar{y}_k = \bar{y}_k$ and the last inequality is due to Lemma V.1.

The result (V.10) is a direct consequence of the (V.9) and the Lipschitz continuity of individual gradients (Assumption III.1). To establish the next result (V.11), we have

$$\begin{aligned}\|g_k - \bar{g}_k\| &= \left\|\frac{1}{n}\sum_{i=1}^{n}\left(\nabla f_i(x_{i,k}) - \nabla f_i(\bar{x}_k)\right)\right\| \\ &\le \frac{1}{n}\sum_{i=1}^{n} L_i\|x_{i,k} - \bar{x}_k\| \le \beta^t D L.\end{aligned}$$

Finally, for the local $y_{i,k}$ iterates in (V.12), consider

$$\begin{aligned}\|y_{i,k} - \bar{y}_k\| &\le \|x_{i,k} - \bar{y}_k\| + \|y_{i,k} - x_{i,k}\| \\ &\le \beta^t D + \|\mathbf{y}_k - \mathbf{x}_k\| \\ &= \beta^t D + \left\|\mathbf{y}_k - \left(\mathbf{W}^t \otimes I\right)\mathbf{y}_k\right\| \\ &\le \beta^t D + \left\|\left(I - \mathbf{W}^t \otimes I\right)\right\|\|\mathbf{y}_k\| \\ &\le \beta^t D + 2D,\end{aligned}$$

where the second inequality is due to (V.9) and the last inequality is due to Lemma V.1. □

Similar to Lemma IV.2, Lemma V.2 shows that the distance between the local iterates $x_{i,k}$ and $y_{i,k}$ are bounded from their means.

We now use an argument similar to that in the previous section to investigate the optimization error of the NEAR-DGD^t method. For that we make the following observation, due to the doubly-stochastic nature of $\mathbf{W}$,

$$\begin{aligned}\bar{\mathbf{y}}_{k+1} &= \frac{1}{n}\left((1_n 1_n^T) \otimes I\right)\mathbf{y}_{k+1} \\ &= \frac{1}{n}\left((1_n 1_n^T) \otimes I\right)\left((\mathbf{W}^t \otimes I)\mathbf{y}_k - \alpha\nabla\mathbf{f}(\mathbf{x}_k)\right) \\ &= \frac{1}{n}\left((1_n 1_n^T\mathbf{W}^t) \otimes I\right)\mathbf{y}_k - \frac{\alpha}{n}\left((1_n 1_n^T) \otimes I\right)\nabla\mathbf{f}(\mathbf{x}_k) \\ &= [\bar{y}_k - \alpha g_k; \bar{y}_k - \alpha g_k; \dots; \bar{y}_k - \alpha g_k],\end{aligned}$$

where g_k is the average of local gradients as defined in (IV-A). Thus we have

$$\bar{y}_{k+1} = \bar{y}_k - \alpha g_k. \tag{V.13}$$

The above equation can be viewed as an inexact gradient descent step for the problem (IV.12), where $\bar{g}_k$ is the exact gradient. Before we proceed we should note again that $\bar{x}_k = \mathbf{Z}^t \bar{y}_k = \bar{y}_k$ due to the nature of the matrix $\mathbf{Z}$. We next follow Theorem IV.3 to bound the distance of iterates to the optimal solution.

Theorem V.3. ***(Bounded distance to minimum)*** *Suppose Assumptions III.1 & III.3 hold, and let the steplength satisfy*

$$\alpha \le \min\left\{\frac{1}{L}, c_4\right\}$$

where $L = \max_i L_i$ *and* $c_4 = \frac{2}{\mu_{\bar{f}} + L_{\bar{f}}}$. *Then, starting from* $x_{i,0} = s_0$ *or* $y_{i,0} = s_0$ $(1 \le i \le n)$, *for all* $k = 0, 1, 2, \dots$

$$\|\bar{x}_{k+1} - x^\star\|^2 \le c_1^2 \|\bar{x}_k - x^\star\|^2 + c_3^2 \beta^{2t},$$

where

$$c_1^2 = 1 - \alpha c_2 + \alpha\delta - \alpha^2 \delta c_2, \;\; c_2 = \frac{2\mu_{\bar{f}} L_{\bar{f}}}{\mu_{\bar{f}} + L_{\bar{f}}},$$
$$c_3^2 = \alpha(\alpha + \delta^{-1}) D^2 L^2, \;\; D = \|\boldsymbol{y}_0 - \boldsymbol{u}^\star\| + \frac{\nu + 4}{\nu}\|\boldsymbol{u}^\star\|,$$

$x^\star$ *is the optimal solution of* (III.1), $\boldsymbol{u}^\star$ *is the optimal solution of* (V.7), $\nu = 2\alpha\gamma$, $\gamma = \min_i \gamma_i$, $\gamma_i = \frac{\mu_i L_i}{\mu_i + L_i}$ *and* $\delta > 0$. *In particular, if we set* $\delta = \frac{c_2}{2(1-\alpha c_2)}$ *such that* $c_1 = \sqrt{1 - \frac{\alpha c_2}{2}} \in (0, 1)$, *then for* $k = 0, 1, 2, \dots$

$$\|\bar{x}_k - x^\star\| \le c_1^k \|\bar{x}_0 - x^\star\| + \frac{LD\beta^t}{c_2}\sqrt{2(2 - \alpha c_2)}.$$

Proof. Following the analysis of Theorem IV.3, and using the definitions of the $\bar{x}_k$ and g_k, and (V.13), we have

$$\begin{aligned} \|\bar{x}_{k+1} - x^\star\|^2 &= \|\bar{x}_k - x^\star - \alpha g_k\|^2 \\ &\le (1 + \alpha\delta)\|\bar{x}_k - x^\star - \alpha \bar{g}_k\|^2 \\ &\quad + \alpha(\alpha + \delta^{-1})\|\bar{g}_k - g_k\|^2, \end{aligned} \tag{V.14}$$

where the last inequality follows from the fact that for any vectors a and b, $\pm 2a^T b \le \delta^{-1}\|a\|^2 + \delta\|b\|^2$, for $\delta > 0$.

We now bound the quantity $\|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2$ as

$$\begin{aligned} &\|\bar{x}_k - x^\star - \alpha\bar{g}_k\|^2 \\ &= \|\bar{x}_k - x^\star\|^2 + \alpha^2\|\bar{g}_k\|^2 - 2(\bar{x}_k - x^\star)^T(\alpha\bar{g}_k) \\ &\le \|\bar{x}_k - x^\star\|^2 + \alpha^2\|\bar{g}_k\|^2 - \alpha c_4\|\bar{g}_k\|^2 - \alpha c_2\|\bar{x}_k - x^\star\|^2 \\ &= (1 - \alpha c_2)\|\bar{x}_k - x^\star\|^2 + \alpha(\alpha - c_4)\|\bar{g}_k\|^2 \\ &\le (1 - \alpha c_2)\|\bar{x}_k - x^\star\|^2, \end{aligned} \tag{V.15}$$

where the first inequality follows from [66, Theorem 2.1.12, Chapter 2], and in the last inequality we dropped the term $\alpha(\alpha - c_4)\|\bar{g}_k\|^2$, since $\alpha \le c_4$ and $\alpha(\alpha - c_4)\|\bar{g}_k\|^2 \le 0$. Combining (V.14), (V.15) and using (V.11),

$$\begin{aligned} \|\bar{x}_{k+1} - x^\star\|^2 &\le (1 + \alpha\delta)(1 - \alpha c_2)\|\bar{x}_k - x^\star\|^2 \\ &\quad + \alpha(\alpha + \delta^{-1})L^2 D^2 \beta^{2t}. \end{aligned} \tag{V.16}$$

Using the definitions of c_1 and c_3, by recursive application of (V.16), we have

$$\|\bar{x}_k - x^\star\|^2 \le c_1^{2k}\|\bar{x}_0 - x^\star\|^2 + \frac{c_3^2}{(1 - c_1^2)}\beta^{2t},$$

and so

$$\|\bar{x}_k - x^\star\| \le c_1^k\|\bar{x}_0 - x^\star\| + \frac{c_3}{\sqrt{1 - c_1^2}}\beta^t.$$

If $\delta = \frac{c_2}{2(1-\alpha c_2)}$, then

$$c_1^2 = 1 - \frac{\alpha c_2}{2} < 1,$$

and

$$\frac{c_3}{\sqrt{1 - c_1^2}}\beta^t = \frac{LD\beta^t}{c_2}\sqrt{2(2 - \alpha c_2)},$$

which completes the proof. □

Theorem V.3 show that the average of the iterates generated by the NEAR-DGD^t method converge to a neighborhood of the optimal solution whose size is defined by the steplength, the second largest eigenvalue of $\mathbf{W}$ and the number of consensus steps. We now provide a convergence result for the local agent estimates of the NEAR-DGD^t method.

Corollary V.4. ***(Local agent convergence)*** *Suppose Assumptions III.1 & III.3 hold, and let the steplength satisfy*

$$\alpha \le \min\left\{\frac{1}{L}, c_4\right\}.$$

where $L = \max_i L_i$ *and* $c_4 = \frac{2}{\mu_{\bar{f}} + L_{\bar{f}}}$. *Then, starting from* $x_{i,0} = s_0$ *or* $y_{i,0} = s_0$ $(1 \le i \le n)$ *for* $k = 0, 1, \dots$

$$\|x_{i,k} - x^\star\| \le c_1^k\|x_0 - x^\star\| + \frac{c_3}{\sqrt{1 - c_1^2}}\beta^t + \beta^t D.$$

Moreover, the local iterates $y_{i,k}$ *are bounded by*

$$\|y_{i,k} - x^\star\| \le c_1^k\|x_0 - x^\star\| + \frac{c_3}{\sqrt{1 - c_1^2}}\beta^t + \beta^t D + 2D.$$

Proof. Using the results from Lemma V.2 and Theorem V.3,

$$\begin{aligned} \|x_{i,k} - x^\star\| &\le \|\bar{x}_k - x^\star\| + \|x_{i,k} - \bar{x}_k\| \\ &\le c_1^k\|x_0 - x^\star\| + \frac{c_3}{\sqrt{1 - c_1^2}}\beta^t + \beta^t D. \end{aligned}$$

Following the same approach for the local iterates $y_{i,k}$, we have

$$\begin{aligned} \|y_{i,k} - x^\star\| &\le \|\bar{y}_k - x^\star\| + \|y_{i,k} - \bar{y}_k\| \\ &= \|\bar{x}_k - x^\star\| + \|y_{i,k} - \bar{y}_k\| \\ &\le c_1^k\|x_0 - x^\star\| + \frac{c_3}{\sqrt{1 - c_1^2}}\beta^t + \beta^t D + 2D, \end{aligned}$$

where the equality is due to $\bar{x}_k = \mathbf{Z}^t\bar{y}_k = \bar{y}_k$. □

The main takeaway of Theorem V.3 is that the iterates generated by the NEAR-DGD^t method converge at a linear rate to a neighborhood of the optimal solution, where the neighborhood is defined as,

$$c_3^2\beta^{2t}.$$

A natural question to ask is whether there is a way to increase the number of consensus steps at every iteration in order to eliminate the error term. In the next theorem, we show that this can actually be achieved, and that the NEAR-DGD$^+$ method with an appropriate increase in the number of consensus steps converges at an R-Linear rate to the solution. Before we proceed, we should mention that the results of Lemmas V.1 and V.2 extend naturally to the case with increasing number of consensus steps.

Theorem V.5. ***(Bounded distance to minimum)*** *Suppose Assumptions III.1 & III.3 hold, and let the steplength satisfy*

$$\alpha \leq \min\left\{\frac{1}{L}, c_4\right\}$$

where $L = \max_i L_i$ *and* $c_4 = \frac{2}{\mu_{\bar{f}} + L_{\bar{f}}}$. *Then, starting from* $x_{i,0} = s_0$ *or* $y_{i,0} = s_0$ $(1 \leq i \leq n)$, *for all* $k = 0, 1, 2, \ldots$

$$\|\bar{x}_{k+1} - x^\star\|^2 \leq c_1^2 \|\bar{x}_k - x^\star\|^2 + c_3^2 \beta^{2t(k)},$$

where

$$c_1^2 = 1 - \alpha c_2 + \alpha\delta - \alpha^2 \delta c_2, \quad c_2 = \frac{2\mu_{\bar{f}} L_{\bar{f}}}{\mu_{\bar{f}} + L_{\bar{f}}},$$
$$c_3^2 = \alpha(\alpha + \delta^{-1}) D^2 L^2, \quad D = \|\boldsymbol{y}_0 - \boldsymbol{u}^\star\| + \frac{\nu + 4}{\nu}\|\boldsymbol{u}^\star\|,$$

$x^\star$ *is the optimal solution of* (III.1), $\boldsymbol{u}^\star$ *is the optimal solution of* (V.7), $\nu = 2\alpha\gamma$, $\gamma = \min_i \gamma_i$, $\gamma_i = \frac{\mu_i L_i}{\mu_i + L_i}$ *and* $\delta > 0$. *Moreover, for any strictly increasing sequence* $\{t(k)\}_k$, *with* $\lim_{k\to\infty} t(k) \to \infty$, *the iterates produced by the NEAR-DGD*$^+$ *algorithm converge to* $x^\star$.

Proof. The proof of Theorem V.5 is exactly the same as that of Theorem V.3, with the difference that the constant number of consensus steps t is replaced by a varying number of consensus steps $t(k)$. The convergence result follows from the fact that

$$\lim_{k\to\infty} \beta^{2t(k)} = 0,$$

for any increasing sequence $\{t(k)\}$ with $\lim_{k\to\infty} t(k) \to \infty$, and thus the size of the error neighborhood $\mathcal{O}(\beta^{2t(k)})$ shrinks to 0. □

Theorem V.6. ***(R-Linear convergence of the NEAR-DGD$^+$ method)*** *Suppose Assumptions III.1 & III.3 hold, let the steplength satisfy*

$$\alpha \leq \min\left\{\frac{1}{L}, c_4\right\}$$

where $L = \max_i L_i$ *and* $c_4 = \frac{2}{\mu_{\bar{f}} + L_{\bar{f}}}$, *and let* $t(k) = k$. *Then, starting from* $x_{i,0} = s_0$ *or* $y_{i,0} = s_0$ $(1 \leq i \leq n)$, *the iterates generated by the NEAR-DGD*$^+$ *method* (V.5)-(V.6) *converge at an R-Linear rate to the solution. Namely,*

$$\|\bar{x}_k - x^\star\| \leq C\rho^k \tag{V.17}$$

for all $k = 0, 1, 2, \ldots$, *where*

$$C = \max\left\{\|\bar{x}_0 - x^\star\|, \frac{2c_3}{\sqrt{\alpha c_2}}\right\}, \quad \rho = \max\left\{\beta, \sqrt{1 - \frac{\alpha c_2}{4}}\right\},$$

and c_1, c_2, c_3 *and* c_4 *are given in Theorem V.5.*

Proof. We prove the result by induction. By the definitions of C and ρ the base case $k = 0$ holds. Assume that the result is true for the k^{th} iteration, and consider the $(k+1)^{th}$ iteration. Using the result of Theorem V.5, we have

$$\begin{aligned}
\|\bar{x}_{k+1} - x^\star\|^2 &\leq c_1^2 \|\bar{x}_k - x^\star\|^2 + c_3^2 \beta^{2k} \\
&\leq c_1^2 \left(C\rho^k\right)^2 + c_3^2 \beta^{2k} \\
&= \left(C\rho^k\right)^2 \left[c_1^2 + \frac{c_3^2 \beta^{2k}}{(C\rho^k)^2}\right] \\
&\leq \left(C\rho^k\right)^2 \left[c_1^2 + \frac{c_3^2}{C^2}\right] \\
&\leq \left(C\rho^k\right)^2 \left[1 - \frac{\alpha c_2}{2} + \frac{\alpha c_2}{4}\right] \\
&\leq \left(C\rho^{k+1}\right)^2
\end{aligned}$$

where the third inequality is due to the fact that $\rho \geq \beta$, the fourth inequality is due to the definitions of c_1, relations $C \geq \frac{2c_3}{\sqrt{\alpha c_2}}$ and $\alpha\delta - \alpha^2\delta c_2 \leq 0$, and the last inequality is due to the definition of ρ, where $\rho \geq \sqrt{1 - \frac{\alpha c_2}{4}}$. □

Theorem V.6 illustrates that when the number of consensus steps is increased at an appropriate rate ($t(k) = k$) then the NEAR-DGD$^+$ method converges to the solution at an R-Linear rate. Going back to Corollary V.4, the result implies that the local iterates $x_{i,k}$ generated by NEAR-DGD$^+$ method converge to the optimal solution, whereas the local iterates $y_{i,k}$ do not.

We now investigate the work complexity of the NEAR-DGD$^+$ method. By work complexity we mean the total amount of work (gradient evaluations and communication steps) required to get an ϵ-accurate solution ($\|\bar{x}_k - x^\star\| \leq \epsilon$).

Corollary V.7. ***(Work Complexity)*** *If the conditions in Theorem V.6 are satisfied, then the work complexity (total number of gradient evaluations* τ_g *and rounds of communications* τ_c*) to get an* ϵ*-accurate solution, that is* $\|\bar{x}_k - x^\star\| \leq \epsilon$, *for the NEAR-DGD*$^+$ *algorithm are given as follows,*

$$\tau_g = \mathcal{O}\left(\log\left({}^1\!/\!_\epsilon\right)\right),$$
$$\tau_c = \mathcal{O}\left(\left(\log\left({}^1\!/\!_\epsilon\right)\right)^2\right).$$

Proof. Due to the result of Theorem V.6, we require $\mathcal{O}\left(\log({}^1\!/\!_\epsilon)\right)$ iterations to get an ϵ-accurate solution, and so the number of gradient evalutions (τ_g) is $\mathcal{O}\left(\log({}^1\!/\!_\epsilon)\right)$. Since we require k communications at the k^{th} iterate, the total number of communications (τ_c) required

$$\tau_c = \sum_{i=1}^{k} i = \frac{(k)(k+1)}{2} = \mathcal{O}\left(k^2\right) = \mathcal{O}\left(\left(\log({}^1\!/\!_\epsilon)\right)^2\right).$$

□

Similar analysis can be done to show the work complexity required to get an ϵ-accurate solution for the local iterates. Note, that this can only be done for the local iterates $x_{i,k}$, but not the local iterates $y_{i,k}$ as these iterates do not converge. These results can then be used in our cost framework (II.1) to obtain the final cost of the algorithm.

B. Numerical Results for NEAR-DGD and NEAR-DGD$^+$

In this section, we present numerical results demonstrating the performance of the NEAR-DGD^t and NEAR-DGD$^+$ methods in practice on quadratic problems and classification problems that arise in machine learning. The aim of this section is to demonstrate that the theoretical results can be realized in practice. More specifically, that the NEAR-DGD^t method converges to a neighborhood of the solution and that the NEAR-DGD$^+$ method converges to the solution, for objective functions that are strongly convex.

We investigated the performance of 6 different variants of the NEAR-DGD method and compared against the DGD method. We define the variants of the NEAR-DGD method as NEAR-DGD (a, b, c), where a is the number of initial gradient steps, b is the number of initial consensus steps and c describes if/how we increase the number of communication steps. For example, (i) NEAR-DGD $(10, 1, -)$ is the NEAR-DGD method with 10 gradient steps for every 1 consensus step with fixed number of consensus and gradient steps; (ii) NEAR-DGD$^+$ $(1, 1, k)$ is the NEAR-DGD$^+$ method with 1 gradient step and 1 consensus step initially, and where the number of consensus steps is increased at every iteration (at the k^{th} iteration k consensus steps are performed for every gradient step); and (iii) NEAR-DGD$^+$ $(1, 1, 500)$ is the NEAR-DGD$^+$ method with 1 gradient step and 1 consensus step initially, and the number of consensus steps is doubled every 500 iterations. The last class of methods are practical implementations of NEAR-DGD$^+$, which we found to perform well in the numerical studies. While the theoretical analysis in this paper does not apply to NEAR-DGD (10,1,-), a method with multiple gradient steps at each iteration, we include it in our numerical studies for comparison purposes.

1) Quadratic Problems: We first investigate the performance of the methods on quadratic functions, similar to those described in Section IV-B,

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x^T A_i x + b_i^T x$$

where each node $i = \{1, ..., n\}$ has local information $A_i \in \mathbb{R}^{p \times p}$ and $b_i \in \mathbb{R}^p$. For these experiments we chose a dimension size $p = 10$, the number of nodes was $n = 10$ and the condition number was $\kappa = 10^4$. We considered a 4-cyclic graph topology (each node is connected to its 4 immediate neighbors). The markers in the figures in this section are placed every 500 iterations. In this regard, one can clearly see the effect of the cost per iteration for the different methods. For example, in the NEAR-DGD$^+$ $(1, 1, k)$ method (dark green lines) of Figure 2, in terms of iterations we have markers, whereas in terms of cost there are no markers. Of course, this is due to the high communication cost associated with each iteration.

Figure 2 illustrates the performance of the methods; we again plot relative error, $(\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2)$ in terms of: (i) iterations, (ii) cost (as described in Section II, with $c_c = c_g = 1$), (iii) number of gradient evaluations, and (iv) number of communications.

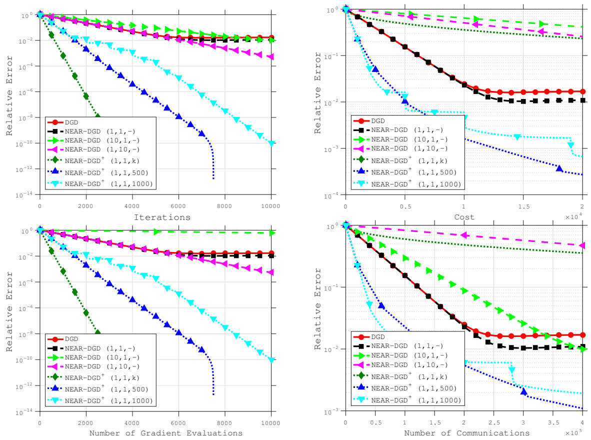


Fig. 2. Performance of DGD, NEAR-DGD $(1, 1, -)$, NEAR-DGD $(10, 1, -)$, NEAR-DGD $(1, 10, -)$, NEAR-DGD$^+$ $(1, 1, k)$, NEAR-DGD$^+$ $(1, 1, 500)$, NEAR-DGD$^+$ $(1, 1, 1000)$ measured in terms of relative error $(\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2)$ with respect to: (i) number of iterations, (ii) cost, (iii) number of gradient evaluations, and (iv) number of communications, on a quadratic problem ($n = 10$, $p = 10$, $\kappa = 10^4$).

Figure 2 shows the convergence rates of the methods. As predicted by the theory, it is evident that the methods that do not increase the number of consensus steps converge only to a neighborhood of the solution, whereas methods that increase the number of consensus steps converge to the solution. We note in passing that the NEAR-DGD method (black line) converges to a better neighborhood that the DGD method (red line); this is not predicted by the theory, and is probably an artifact of the specific problem.

The NEAR-DGD$^+$ method converges to the solution as predicted by the theory. In terms of number of iterations, the fastest method is the NEAR-DGD$^+$ $(1, 1, k)$ method. This is not surprising as the amount of work per iteration in this method is higher than that of the of NEAR-DGD$^+$ $(1, 1, 500)$ and NEAR-DGD$^+$ $(1, 1, 1000)$ (the practical variants). When comparing the methods in terms of the cost or the number of communications, the practical variants of the NEAR-DGD$^+$ method perform significantly better. The cost metric in this case is a better indication of the performance of the method. In Figure 2 we assumed that the cost of a gradient evaluation and the cost of communication was the same, namely $c_c = c_g = 1$.

In Figure 3 we show the performance of the methods for different cost structures: (i) $c_c = 1, c_g = 10$; (ii) $c_c = 1, c_g = 1$; (iii) (i) $c_c = 10, c_g = 1$. The cost structures where $c_c \neq c_g$ arise in applications (problems) such as those described in Section II. For example, the cost structure $c_c = 1, c_g = 10$ can be found in applications such as large scale machine learning problems on a cluster of physically connected computers, and the cost structure $c_c = 10, c_g = 1$ can be in found in applications such as controlling a swarm of battery powered robots.

Figure 3 shows that the performance of the methods is highly dependent on the specific cost structure of the application. Although in all cases the practical variants of the NEAR-DGD$^+$ method perform the best in terms of the cost, the benefit of doing multiple consensus steps varies. On the left-most figure, the benefits are very apparent, whereas on the

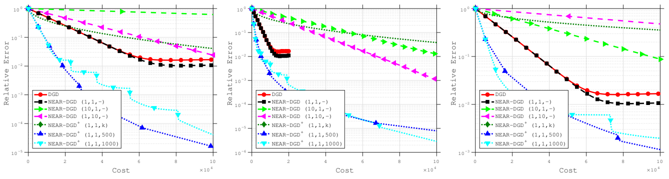


Fig. 3. Performance of DGD, NEAR-DGD (1,1,-), NEAR-DGD (10,1,-), NEAR-DGD (1,10,-), NEAR-DGD$^+$ (1,1,k), NEAR-DGD$^+$ (1,1,500), NEAR-DGD$^+$ (1,1,1000) measured in terms of relative error ($\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2$) with respect to different cost structures, on a quadratic problem ($n = 10$, $p = 10$, $\kappa = 10^4$). **Left**: $c_c = 1$, $c_g = 10$; **Center**: $c_c = 1$, $c_g = 1$; **Right**: $c_c = 10$, $c_g = 1$.

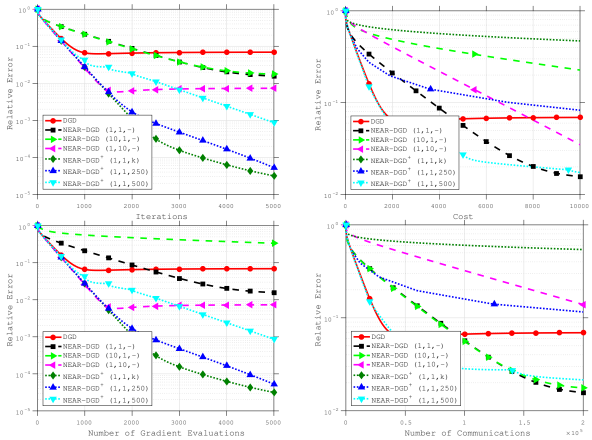


Fig. 4. Performance of DGD, NEAR-DGD $(1, 1, -)$, NEAR-DGD $(10, 1, -)$, NEAR-DGD $(1, 10, -)$, NEAR-DGD$^+$ $(1, 1, k)$, NEAR-DGD$^+$ $(1, 1, 500)$, NEAR-DGD$^+$ $(1, 1, 1000)$ measured in terms of relative error ($\|\bar{x}_k - x^\star\|^2/\|x^\star\|^2$) with respect to: (i) number of iterations, (ii) cost, (iii) number of gradient evaluations, and (iv) number of communications, on a binary classification logistic regression problem (mushroom. $n = 10$, $p = 114$, $n_i = 812$).

right-most figure, the benefits are not as apparent. That being said, of course, it is still the case that the methods that do not increase the number of consensus steps cannot converge to the solution.

2) Binary Classification Logistic Regression Problems: We now show numerical results illustrating the performance of the NEAR-DGD^t and NEAR-DGD$^+$ methods on binary classification logistic regression problems that arise in machine learning. The objective function can be expressed as

$$f(x) = \frac{1}{n \cdot n_i} \sum_{i=1}^{n} \log(1 + e^{-(b_i)^T(A_i x)}) + \frac{1}{n \cdot n_i} \|x\|_2^2$$

where $A \in \mathbb{R}^{n \cdot n_i \times p}$ and $b \in \{-1, 1\}^{n \cdot n_i}$, ($n$ denotes the number of nodes, n_i denotes the size of the local data and p is the dimension of the problem) and each node $i = 1, ..., n$ has a portion of A and b, $A_i \in \mathbb{R}^{n_i \times p}$ and $b_i \in \mathbb{R}^{n_i}$. We report results for the mushrooms dataset ($n = 10$, $p = 114$, $n_i = 812$) [67], where the underlying graph is 4-cyclic. Similar results were obtained for other standard machine learning datasets. For this experiment we set $c_c = 1, c_g = 1$.

Figure 4 illustrates the convergence rates of the 7 methods. The results are similar to those for the quadratic problem. Namely, the variants that increase the number of consensus steps converge to the optimal solution whereas the other methods only converge to a neighborhood of the solution. Interestingly, the NEAR-DGD (1,1,-) method is able to converge to a significantly better neighborhood that the base DGD method. Moreover, for a fixed budget (cost), it appears that the NEAR-DGD (1,1,-) method is competitive with the NEAR-DGD$^+$ variants. We should note that the figure plotting the error with respect to the cost does not show the outcome of the full experiment but rather only till the point that the DGD method terminated. However, looking at the per-iterations plots, the performance of the NEAR-DGD (1,1,-) method stagnates whereas the performance of the NEAR-DGD$^+$ methods does not.

VI. Final Remarks and Future Work

In this paper, we propose an adaptive cost framework to evaluate the performance of distributed optimization methods. Given a specific application, our framework incorporates the costs associated with both communication and computation in order to evaluate the efficiency of distributed optimization methods. This work is a first step towards applying the proposed general cost framework. In particular, we study the well-known distributed gradient descent (DGD) method and decompose its communication and computation steps to construct three classes of more flexible methods: DGD^t, NEAR-DGD^t and NEAR-DGD$^+$.

The flexibility for each of these methods is illustrated by the fact that multiple consensus steps can be performed per gradient evaluation in environments where communication is relatively inexpensive. We show both theoretically and empirically that multiple consensus steps lead to better solution quality. We also design NEAR-DGD$^+$, an exact first order method, which increases the number of consensus steps as the algorithm progresses. As such, NEAR-DGD$^+$ with a constant steplength converges to the optimal solution, as opposed to the standard DGD method that only converges to a neighborhood of the optimal solution. Additionally, we show that for strongly convex functions, the NEAR-DGD$^+$ method converges at a linear rate. Finally, through numerical experiments of different instances of these methods on quadratic and (binary classification) logistic regression problems, we illustrate the empirical performance of the methods and demonstrate the flexibility offered by our framework.

We should note that the same communication-computation decomposition approach can be applied seamlessly to many other distributed optimization methods, and this is a direction of future research that we wish to explore. Moreover, we plan to include other cost aspects into this framework, such as memory access, quantization and dynamic environments. Lastly, the question of how to automatically adjust the number of communication and computation steps, in an algorithmic way, depending on the environment, is a direction that we are currently investigating.

References

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

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

[3] I. D. Schizas, R. Ribeiro, and G. B. Giannakis, "Consensus in Ad Hoc WSNs with Noisy Links - Part I: Distributed Estimation of Deterministic Signals," *IEEE Transactions on Singal Processing*, vol. 56, pp. 350–364, 2008.

[4] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal processing magazine*, vol. 19, no. 2, pp. 61–72, 2002.

[5] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[6] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems*, vol. 27, no. 2, pp. 71–82, 2007.

[7] K. Zhou and S. I. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 678–695, 2011.

[8] G. B. Giannakis, V. Kekatos, N. Gatsis, S. Kim, H. Zhu, and B. F. Wollenberg, "Monitoring and optimization for power grids: A signal processing perspective," *IEEE Signal Processing Magazine*, vol. 30, no. 5, pp. 107–128, 2013.

[9] V. Kekatos and G. B. Giannakis, "Distributed robust power system state estimation," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1617–1626, 2013.

[10] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[11] K. Tsianos, S. Lawlor, and M. G. Rabbat, "Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE, 2012, pp. 1543–1550.

[12] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.

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

[14] J. N. Tsitsiklis, "Problems in Decentralized Decision Making and Computation," Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1984.

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, no. SI, pp. 2508–2530, 2006.

[16] A. Jadbabaie, J. Lin, and S. Morse, "Coordination of Groups of Mobile Autonomous Agents using Nearest Neighbor Rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[17] D. Jakovetic, J. Xavier, and J. M. F. Moura, "Fast Distributed Gradient Methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.

[18] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient Methods and Consensus Algorithms for Solving Convex Optimization Problems," *Proceedings of IEEE Conference on Decision and Control(CDC)*, pp. 4185–4190, 2008.

[19] I. Lobel and A. Ozdaglar, "Convergence Analysis of Distributed Subgradient Methods over Random Networks," *Proceedings of Annual Allerton Conference on Communication, Control, and Computing*, 2008.

[20] I. Lobel, A. Ozdaglar, and D. Feijer, "Distributed Multi-agent Optimization with State-Dependent Communication," *Mathematical Programming, special issue in honor of Paul Tseng*, vol. 129, no. 2, pp. 255–284, 2011.

[21] I. Matei and J. S. Baras, "Performance Evaluation of the Consensus-Based Distributed Subgradient Method Under Random Communication Topologies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 754–771, 2011.

[22] A. Nedić, "Asynchronous broadcast-based convex optimization over a network," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1337–1351, 2011.

[23] A. Nedić, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "Distributed Subgradient Algorithms and Quantization Effects," *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2008.

[24] A. Nedić and A. Ozdaglar, *Convex Optimization in Signal Processing and Communications*. Eds., Eldar, Y. and Palomar, D., Cambridge University Press, 2008, ch. Cooperative distributed multi-agent optimization.

[25] ——, "Approximate primal solutions and rate analysis for dual subgradient methods," *SIAM Journal on Optimization*, vol. 19(4), pp. 1757–1780, 2009.

[26] A. Nedić, A. Ozdaglar, and P. A. Parrilo, "Constrained Consensus and Optimization in Multi-agent Networks," *IEEE Transactions on Automatic Control*, vol. 55(4), pp. 922–938, 2010.

[27] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed Stochastic Subgradient Projection Algorithms for Convex Optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, 2010.

[28] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Asynchronous gossip algorithm for stochastic optimization: Constant stepsize analysis," in *Recent Advances in Optimization and its Applications in Engineering*. Springer, 2010, pp. 51–60.

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

[30] K. Srivastava and A. Nedić, "Distributed asynchronous constrained stochastic optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 772–790, 2011.

[31] M. Hong, M. Razaviyayn, Z. Luo, and J. Pang, "A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing," *IEEE Signal Processing Magazine*, vol. 33, no. 1, pp. 57–77, 2016.

[32] Z. Peng, Y. Xu, M. Yan, and W. Yin, "Arock: an algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, 2016.

[33] P. Richtárik and M. Takáč, "Parallel coordinate descent methods for big data optimization," *Mathematical Programming*, vol. 156, no. 1-2, pp. 433–484, 2016.

[34] Y. Nesterov, "Primal-dual subgradient methods for convex problems," *Mathematical Programming*, vol. 120, no. 1, pp. 221–259, 2009.

[35] D. P. Bertsekas, "Incremental Gradient, Subgradient, and Proximal Methods for Convex Optimization: A Survey," *LIDS Report 2848*, 2010.

[36] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[37] T. Chang, M. Hong, W. Liao, and X. Wang, "Asynchronous Distributed ADMM for Large-Scale OptimizationPart I: Algorithm and Convergence Analysis," *IEEE Transactions on Signal Processing*, vol. 64, no. 12, pp. 3118–3130, 2015.

[38] J. Eckstein, "Augmented Lagrangian and Alternating Direction Methods for Convex Optimization: A Tutorial and Some Illustrative Computational Results," *Rutcor Research Report*, 2012.

[39] P. Giselsson and S. Boyd, "Diagonal scaling in douglas-rachford splitting and admm," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 5033–5039.

[40] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous Distributed Optimization using a Randomized Alternating Direction Method of Multipliers," *Submitted to IEEE Conference on Decision and Control (CDC)*, 2013.

[41] J. Mota, J. Xavier, P. Aguiar, and M. Püschel, "ADMM For Consensus On Colored Networks," *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2012.

[42] ——, "D-ADMM : A Communication-Efficient Distributed Algorithm For Separable Optimization," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, 2013.

[43] E. Wei and A. Ozdaglar, "Distributed Alternating Direction Method of Multipliers," *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2012.

[44] ——, "On the $O(1/k)$ convergence of asynchronous distributed alternating Direction Method of Multipliers," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 551–554.

[45] L. Xiao and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 2057–2075, 2014.

[46] H. Zhu, A. Cano, and G. B. Giannakis, "In-Network Channel Decoding Using Consensus on Log-Likelihood Ratio Averages," *Proceedings of Conference on Information Sciences and Systems (CISS)*, 2008.

[47] M. Eisen, A. Mokhtari, and A. Ribeiro, "Decentralized quasi-newton methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2613–2628, 2017.

[48] F. Mansoori and E. Wei, "Superlinearly Convergent Asynchronous Distributed Network Newton Method," *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2017.

[49] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network newton-part i: Algorithm and convergence," *arXiv preprint arXiv:1504.06017*, 2015.

[50] ——, "Network newton-part ii: Convergence rate and implementation," *arXiv preprint arXiv:1504.06020*, 2015.

[51] Y. Chow, W. Shi, T. Wu, and W. Yin, "Expander graph and communication-efficient decentralized optimization," in *Signals, Systems and Computers, 2016 50th Asilomar Conference on*. IEEE, 2016, pp. 1715–1720.

[52] G. Lan, S. Lee, and Y. Zhou, "Communication-efficient algorithms for decentralized and stochastic optimization," *arXiv preprint arXiv:1701.03961*, 2017.

[53] O. Shamir, N. Srebro, and T. Zhang, "Communication-efficient distributed optimization using an approximate newton-type method," in *International conference on machine learning*, 2014, pp. 1000–1008.

[54] K. Tsianos, S. Lawlor, and M. G. Rabbat, "Communication/computation tradeoffs in consensus-based distributed optimization," in *Advances in neural information processing systems*, 2012, pp. 1943–1951.

[55] Y. Zhang and X. Lin, "Disco: Distributed optimization for self-concordant empirical loss," in *International conference on machine learning*, 2015, pp. 362–370.

[56] Y. Zhang, M. J. Wainwright, and J. C. Duchi, "Communication-efficient algorithms for statistical optimization," in *Advances in Neural Information Processing Systems*, 2012, pp. 1502–1510.

[57] A. H. Sayed, *Diffusion adaptation over networks*. Academic Press Library in Signal Processing, 2013, vol. 3.

[58] A. I. Chen and A. Ozdaglar, "A fast distributed proximal-gradient method," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE, 2012, pp. 601–608.

[59] P. Di Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[60] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *arXiv preprint arXiv:1607.03218*, 2016.

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

[62] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *arXiv preprint arXiv:1704.07807*, 2017.

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

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

[65] A. Mokhtari, Q. Ling, and A. Ribeiro, "Network newton distributed optimization methods," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 146–161, 2017.

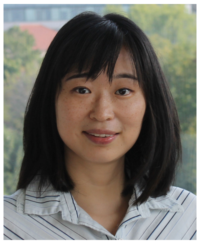
[66] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[67] C. C. Chang and C. J. Lin, "Libsvm: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

Albert S. Berahas is currently a Postdoctoral Research Fellow in the Industrial Engineering and Management Sciences (IEMS) Dept. at Northwestern University working under the supervision of Professor Jorge Nocedal. He completed his PhD studies in the Engineering Sciences and Applied Mathematics (ESAM) Dept. at Northwestern University in 2018, advised by Professor Jorge Nocedal. He received his undergraduate degree in Operations Research and Industrial Engineering (ORIE) from Cornell University in 2009, and in 2012 obtained an M.S. degree in Applied Mathematics from Northwestern University. Berahas has received the ESAM Outstanding Teaching Assistant Award, the Walter P. Murphy Fellowship and the John N. Nicholson Fellowship. Berahas' research interests include optimization algorithms for machine learning, convex optimization and analysis, derivative-free optimization and distributed optimization.

Raghu Bollapragada is currently a PhD student in the Industrial Engineering and Management Sciences (IEMS) Dept. at Northwestern University, working under the supervision of Professor Jorge Nocedal. He received his undergraduate degree in Mechanical Engineering from Indian Institute of Technology (IIT) Madras, India in 2014. Bollapragada has received the IEMS Arthur P. Hurter Award for outstanding academic excellence among first year graduate students and the Walter P. Murphy Fellowship at Northwestern University. Bollapragada's research interests include algorithms for machine learning, large-scale nonlinear optimization methods, convex optimization and analysis, stochastic optimization and distributed optimization.

Nitish Shirish Keskar is currently a Senior Research Scientist at Salesforce Research. He completed his PhD studies in Industrial Engineering and Management Sciences at Northwestern University in 2017. He was co-advised by Jorge Nocedal and Andreas Waechter. Prior to his PhD, Keskar received his undergraduate degree in Mechanical Engineering from VJTI in Mumbai, India. Keskar is the recipient of the 2017 best paper award for the Optimization Methods and Software journal for his paper on second-order method for L1 regularized convex optimization. Keskar's research interests include nonlinear optimization, deep learning and large-scale computing.



Ermin Wei is currently an Assistant Professor at the EECS Dept. of Northwestern University. She completed her PhD studies in Electrical Engineering and Computer Science at MIT in 2014, advised by Professor Asu Ozdaglar, where she also obtained her M.S.. She received her undergraduate triple degree in Computer Engineering, Finance and Mathematics with a minor in German, from University of Maryland, College Park. Wei has received many awards, including the Graduate Women of Excellence Award, second place prize in Ernst A. Guillemen Thesis Award and Alpha Lambda Delta National Academic Honor Society Betty Jo Budson Fellowship. Wei's research interests include distributed optimization methods, convex optimization and analysis, smart grid, communication systems and energy networks and market economic analysis.