2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)
Allerton Park and Retreat Center
Monticello, IL, USA, October 2-5, 2018

A Code-Based Distributed Gradient Descent Method

Elie Atallah
Department of Electrical and Computer Engineering
University of Central Florida
Orlando, FL
elieatallah@knights.ucf.edu

Nazanin Rahnavard
Department of Electrical and Computer Engineering
University of Central Florida
Orlando, FL
nazanin@eecs.ucf.edu

***Abstract*—Distributed gradient descent is an optimization algorithm that is used to solve a minimization problem distributed over a network through minimizing local functions that sum up to form the overall objective function. These local functions $f_i(.)$ contribute to local gradients adding up incrementally to form the overall gradient. Recently, the gradient coding paradigm was introduced for networks with a centralized fusion center to resolve the problem of straggler nodes. Through introducing some kind of redundancy on each node, such coding schemes are utilized to form new coded local functions g_i from the original local functions f_i. In this work, we consider a distributed network with a defined network topology and no fusion center. At each node, linear combinations of the local coded gradients $\nabla \bar{g}_i$ can be constructed to form the overall gradient. Our iterative method, referred to as *Code-Based Distributed Gradient Descent* (CDGD), updates each node's local estimate by applying an adequate weighing scheme. This scheme adapts the coded local gradient descent step along with local estimates from neighboring nodes. We provide the convergence analysis for CDGD and we analytically show that we enhance the convergence rate by a scaling factor over conventional incremental methods without any predefined tuning. Furthermore, we demonstrate through numerical results significant performance and enhancements for convergence rates.**

***Index Terms*—decentralized optimization, gradient coding, consensus, distributed networks**

I. Introduction

OPTIMIZATION in decentralized distributed systems has played a significant role for solving various problems such as distributed spectrum sensing in cognitive radio networks, distributed parameter estimation in wireless sensor networks, source localization in cellular networks as well as processing big-data in machine learning [5, 9]. The optimization problem is posed as a minimization program of the sum of local objective functions. These functions add up incrementally to form the total overall objective function to be minimized, that is $f = \sum_{i=1}^{n} f_i$ [11, 13]. For such minimization, different algorithms have been proposed ranging from distributed gradient based methods to methods using Lagrangians and dual variables throughout their variant forms [1]. This optimization process can be carried out on either centralized or decentralized networks. Moreover, these methods have achieved sublinear convergence rates for convex functions. When the convex functions are nonsmooth, the sublinear convergence rate matches the centralized gradient method. More recent studies have used predefined tuning to conventional incremental gradient methods to achieve linear convergence rate [6]. Our method seeks such an improvement by a scaling factor on the conventional methods convergence rate through utilizing new local functions. That can also be better improved if adequately enhanced by tuning. Accordingly, we present a decentralized algorithm for solving optimization problems in static networks dependent upon the specified connection topology and utilizing a gradient descent approach. The nodes should have a number of connecting neighbors that satisfy a corresponding gradient coding scheme [7, 14, 17]. Consequently, this scheme is used to construct a corresponding weighting structure along with coded local functions, carrying some redundancy, that add up to form the overall objective function. This structure is used in updating the solution estimates per each node. Thus, each node updates its estimates by adding all the weighted local estimates of the neighboring nodes with its local gradient descent-direction step. After exchanging their local estimates, the structure of such updating ensures that all agents reach both consensus and optimality.

The remainder of the paper is organized as follows. In Section II we present the problem setup with the considered model assumptions. In Section III, we formulate our proposed algorithm CDGD with the needed background material. In Section IV, we state our main fundamental theorem for the validity of the algorithm convergence. In Section V we prove the convergence of our algorithm. Afterwards, we find the convergence rate in Section VI. We complement our work in Section VII with simulation results. Finally, Section VIII concludes our paper. In what follows, we reserve capital letters e.g. A, B for matrices.Scalar variables and vectors are denoted by lower case letters e.g x, y as it is understood from the context. A row of matrix A is denoted by A_i while the entry of index (i, j) is denoted by $[A]_{ij}$. While matrix $[A]_{[i:i'][j:j']}$ is the matrix block formed by varying row index from i to $i^{'}$ and column index from j to $j^{'}$, respectively. ∇f denotes the gradient of f and $A^{'}$ denotes the transpose of A, while $1_{m \times n}$ is reserved for the all $1's$ vector or matrix of dimension $m \times n$. $\|M\|_{\infty} = \max_i \sum_{j=1}^{n} |[M]_{ij}|$, the infinity norm of a matrix M and $\|M\|_{2,\infty} = max_i(\sum_{j=1}^{n}[M]_{ij}^2)^{\frac{1}{2}}$ the norm of the row with the maximum Euclidean norm in a matrix M.

II. Problem Setup

We consider a distributed convex optimization problem over an undirected graph $\mathbf{G} = (V, \mathcal{E})$ that represents the network topology, where V is the set of vertices of cardinality $|V| = n$ and $\mathcal{E} \subset V \times V$ is the set of edges. The neighborhood of node i is $\mathcal{N}_i = \{j \mid (i,j) \in \mathcal{E}\} \cup \{i\}$. We aim on solving an optimization problem in a distributed manner. That is,

$$\min_{x \in \mathbf{R}^N} f(x) = \sum_{i=1}^{m} f_i(x), \tag{1}$$

where $f(x)$ is the global overall function to be minimized, $f_i(x)$ are local functions related to the used partition, and m is the number of partition subsets.

Instead of using uncoded local gradients of local functions (that sum up incrementally to the overall objective function) [2, 11, 13], we use coded local gradients dependent on the network topology by borrowing the coded paradigm presented in [17]. That is, instead of solving

$$\min_{x \in \mathbf{R}^N} f(x) = \sum_{i=1}^{n} f_i(x) \tag{2}$$

where the number of partitions m is equal to the number of nodes n, we divide the global function into the sum of l local functions $g_i(x)$ available at each node. In the first approach (2) the coefficient of each function is set to unity so that each node can incrementally add its local function share to form the overall function. However, in our proposed coded gradient approach (3), each node has a local function $g_i(x)$ which is a combination of the original uncoded functions in such a manner that a combination of these new coded local functions forms the overall function. That is,

$$\min_{x \in \mathbf{R}^N} f(x) = \sum_{i=1}^{n} f_i(x) = \sum_{j=1}^{l} A_{ij} g_j(x), \tag{3}$$

where $g_i(x)$ is a combination of $f_i(x)$ according to the coding scheme structure at each node (cf., III-B) and A is the weighing combination matrix.

According to the nodes connection topology at each iteration (here we assume a fixed topology), we apply an adequate weighting to construct the estimate update at each node. Thus, when examining scenarios of different network topologies, each network is identified with a specific coding scheme. Each of which is a coding scheme in [17], which is usually used when some nodes are vulnerable to failures or delays known as stragglers.

Assumption 1. *We list the following assumptions essential for the applicability of our algorithm.*
(a) *The network* $\mathbf{G}$ *is strongly connected [3, 18].*
(b) *The function* f *is convex and each coded function* $\bar{g}_i : \mathbb{R}^N \to \mathbb{R}$ *is convex for* $1 \leq i \leq 2n$.
(c) *The solution set of (3) and the optimal value exist.*
$x^* \in X^* = \{x | f(x) = \min_{x'} f(x')\}$,
$f^* = f(x^*) = \min f(x)$
(d) *The gradients* $\nabla f_i(x)$*, where* $i \in V$ *are bounded over the* $\mathbb{R}^N$*, i.e. there exists a constant* F *such that* $||\nabla f_i(x)|| \leq F$ *for all* $x \in \mathbb{R}^N$ *and all* $i \in V$ *(i.e.,* $\|g_i(x)\| \leq G = \sqrt{n}\|B\|_{2,\infty}F$

Remark 1. *Where* $\bar{g}_i$ *used in Assumption 1 (b) means the coded local function* g_i *for* $1 \leq i \leq n$ *available at node* i *and* $-g_{i-n}$ *for* $n+1 \leq i \leq 2n$ *available at node* $i-n$*. That is,* g_i *convex if the corresponding coefficients of node* i *used in* A_{fit} *are all positive and* g_i *concave if the corresponding coefficients of node* i *used in* A_{fit} *are all negative and* g_i *linear if some of the corresponding coefficients of node* i *are positive and the others are negative. (N.B. in our detailed to be published work we use the more relaxed assumption of requiring only the global function* f *to be convex.)*

III. Main Algorithm

A. Gradient Coding Scheme

We begin with an overview of gradient coding as introduced in [17] since it is fundamental in forming the weighing matrix used in our distributed algorithm.

Problems in networks arise when worker nodes become stragglers (Li et al. [10], Ho et al. [8], Dean et al. [4]) i.e. fail or get delayed significantly when computing or communicating their local information. Tandon et al. [17] discuss one way to resolve this problem by replicating some data across machines in a defined coding scheme. This scheme is described on a centralized network where local workers exchange their local gradients with the master node to solve the global optimization problem. To that end, they propose a deterministic construction based on a defined coding scheme accompanied with an efficient decoder, which is used to recover the full gradient update at iteration k from a fixed number of local gradients from $\Delta(k) \subset \{1, .., n\}$ returning machines with $|\Delta(k)| \geq n - s$, where s is the maximum number of allowed stragglers. This coding scheme can be used without any feedback and coordination for identifying the straggler nodes. Particularly, $\Delta(k)$ contains all partition subsets J_i for $i \in \{1, .., m\}$ (i.e. $\cup_{i=1}^{m} J_i \subset$ partitions of $\Delta(k)$). Thus, the overall cost function, gradient respectively, of the system can be determined from the set $\Delta(k)$. The coding theoretic question is to design a code such that any $n-s$ linear combination of these individual linear combinations contains the overall gradient of the function $f = \sum_{i=1}^{m} f_i$. A coding scheme robust to any s stragglers corresponding to n nodes and m data partitions can be identified by a system of linear equation:

$$AB = 1_{\binom{n}{s} \times m}, \tag{4}$$

where $\binom{n}{s}$ denotes the number of combinations of connected-nodes/stragglers scenarios, $A \in \mathbb{R}^{\binom{n}{s} \times n}$ and $B \in \mathbb{R}^{n \times m}$. Without loss of generality, we assume the number of nodes n equals to the number of partition subsets m in our algorithm. The i^{th} row of B, B_i is associated with node i. The support of B_i, $supp(B_i)$, corresponds to the local functions f_l, (i.e the corresponding local functions f_l of partitions J_l), which node i has access to, and the entries B_{ij} encode a linear combination over the gradients that node i transmits to the task master at the aggregation step. Let $\overline{\overline{\nabla f}} \triangleq [\nabla f_1, ..., \nabla f_m]' \in \mathbb{R}^{m\times N}$ be a matrix with each row being the partial gradient corresponding to the function f_i of partition J_i. Then, node i transmits $\nabla \bar{g}_i \triangleq B_i \overline{\overline{\nabla f}}$ while each row of A corresponds to a specific failure/straggler scenario, (i.e. only for the maximum allowed number of stragglers case). The support of A_i, $supp(A_i)$, corresponds to the scenario where the nodes in $supp(A_i)$ are connected. A and B can be computed using Algorithm 1 and Algorithm 2 in [17], respectively.

The aforementioned outline is based on the particular coding scheme introduced in [17], however there exist other schemes with different coding structures [7, 14].

B. Forming the matrix A_{fit}

The following steps in **Algorithm 1** are used for identifying the matrix A_{fit} corresponding to the gradient coding scheme.

C. Code-based Distributed Gradient Descent (CDGD) over a Network

As previously mentioned, for achieving convergence of our algorithm to the optimal desired solution we need to consider a matrix that carries as close as possible the features of *stochastic matrices*. More specifically, we form a block matrix O with each block of size $2n \times 2n$ such that

$$O = \begin{pmatrix} A_{fit} & \epsilon I_{2n\times 2n} \\ I_{2n\times 2n} - A_{fit} & \begin{pmatrix} D - \epsilon I_{n\times n} & 0 \\ 0 & D - \epsilon I_{n\times n} \end{pmatrix} \end{pmatrix}. \quad (5)$$

CDGD performs the following updating iterations at each node i for $i \in \{1, 2, \ldots, n\}$. Please note that Γ_i is the fixed support of the row of A_{fit} identified with node i:

$$\begin{aligned}
x_i^+(k+1) &= \sum_{j\in\Gamma_i} [A_{fit}]_{ij} x_j(k) + \epsilon y_i^+(k) - \alpha_k \nabla \bar{g}_i(x_i^+(k)) \\
x_i^-(k+1) &= \sum_{j\in\Gamma_i} [A_{fit}]_{ij} x_j(k) + \epsilon y_i^-(k) + \alpha_k \nabla \bar{g}_i(x_i^-(k)) \\
y_i^+(k+1) &= x_i^+(k) - \sum_{j\in\Gamma_i} [A_{fit}]_{ij} x_j(k) + \sum_{j\in\mathcal{N}_i} [D]_{ij} y_j^+(k) - \epsilon y_i^+(k) \\
y_i^-(k+1) &= x_i^-(k) - \sum_{j\in\Gamma_i} [A_{fit}]_{ij} x_j(k) + \sum_{j\in\mathcal{N}_i} [D]_{ij} y_j^-(k) - \epsilon y_i^-(k)
\end{aligned} \quad (6)$$

Each node $i \in V$ maintains four vectors: two estimates $x_i^+(k)$, $x_i^-(k)$, and two surpluses $y_i^+(k)$ and $y_i^-(k)$, all in $\mathbb{R}^N$, where k is the discrete time iteration. We use $x_j(k)$ to mean either $x_j^+(k)$ and $x_{j-n}^-(k)$ for $1 \le j \le n$ and $n+1 \le j \le 2n$, respectively. At the k^{th} iteration, node j sends its estimates and surpluses to each of its neighbors, $i \in \mathcal{N}_j$ such that each estimate is weighted by $[A_{fit}]_{ij}$ and each surplus is weighted by $[D]_{ij}$. We can initialize our estimates $x_i^+(0)$ and $x_i^-(0)$ randomly or set them initially to zero.

The CDGD algorithm can be summarized by the following recursive equation:

$$z_i(k+1) = \sum_{j=1}^{4n} [O]_{ij} z_j(k) - \alpha_k \nabla \bar{g}_i(z_i(k)) \; for \; 1 \le i \le 4n, \quad (7)$$

Algorithm 1 Forming A_{fit} and the Updating Matrix O

```
Input: The Static Network Graph $\mathbf{G} = (V, \mathcal{E})$
1: Find $n = |V|$.
2: Finding Neighborhoods: Find $\mathcal{N}_i$ for all nodes $i$.
3: Find $s_i = n - |\mathcal{N}_i|$.
4: Choosing Coding Scheme: Choose a coding scheme [17] with $\lfloor \frac{n}{2} \rfloor \ge s \ge \max_i s_i$ where $s$ is the allowed number of stragglers.
5: Forming Rows of $A'$: Match the sets of nonzeros in the $A$ matrix of [17] corresponding to node $i$. Identify each column of $A$ with a unique fixed node. That is:
6: for $i = 1$ to $n$ do
7:   for $l = 1$ to $\binom{n}{s}$ do
8:     if $supp(A_l) \subset \mathcal{N}_i$ then
9:       $A_i' = A_l$
10:    end if
11:  end for
12: end for
13: Forming $A''$ : Move the negative coefficient at column $1 \le j \le n$ of $A'$ to be the opposite positive coefficient at $n+j$. That is, define $A'' = [A' 0_{n\times n}]$. If $[A']_{ij} < 0 \rightarrow [A'']_{ij} = 0$ and $[A'']_{i(j+n)} = -[A']_{ij}$.
14: Normalizing $A''$: Normalize the rows of $A''$ by their $l_1$ norm.
15: Forming $A_{fit}$: $A_{fit} = \begin{bmatrix} A'' \\ A'' \end{bmatrix}$. For $1 \le i \le n$, row or column $i$ of $A_{fit}$ correspond to node $i$. For $n+1 \le i \le 2n$, row or column $i$ of $A_{fit}$ corresponds to node $i-n$. The $2n \times 2n$ matrix $A_{fit}$ is formed.
16: Identifying the local coded gradient $\nabla g_i$ at each node $i$: Local gradient $\nabla g_i$ corresponds to row $i$ of matrix $B$ of [17].
17: Forming adjacency matrix $D$: Form $D$, the Adjacency Matrix of the network $G$, as a stochastic matrix, where $[D]_{ij} = \begin{cases} \frac{1}{deg(j)+1}, & i \in \mathcal{N}_j \\ 0, & otherwise \end{cases}$ $\sum_{i=1}^n [D]_{ij} = 1, \forall j$
18: Choosing $\epsilon$: Choose perturbation $\epsilon$ satisfying Theorem 1
19: Forming Matrix $O$ according to Definition 2.
```

where

$$z_i(k) = \begin{cases} x_i^+(k), & 1 \le i \le n \\ x_{i-n}^-(k), & n+1 \le i \le 2n \\ y_{i-2n}^+(k), & 2n+1 \le i \le 3n \\ y_{i-3n}^-(k), & 3n+1 \le i \le 4n \end{cases}$$

We take $\nabla \bar{g}_i(z_i(k))$ to correspond to the coded gradients relative to the first $2n$ variables identified with the estimates x_i. And $\nabla \bar{g}_i(k) = 0$ for $2n+1 \le i \le 4n$ corresponding to the surplus variables y_i. That is,

$$\nabla \bar{g}_i(z_i(k)) = \begin{cases} \nabla g_i(z_i(k)) = \nabla g_i(x_i^+(k)), & 1 \le i \le n \\ -\nabla g_{i-n}(z_i(k)) = -\nabla g_{i-n}(x_i^-(k)), & n+1 \le i \le 2n \\ 0, & 2n+1 \le i \le 3n \\ 0, & 3n+1 \le i \le 4n \end{cases}$$

The step-size $\alpha_k \geq 0$ and satisfies $\sum_{k=0}^{\infty} \alpha_k = \infty, \sum_{k=0}^{\infty} \alpha_k^2 < \infty$. The scalar ϵ is a small positive number which is essential for the convergence of the algorithm and satisfying the conditions of Theorem 1. The steps of CDGD are summarized in Algorithm 2. We will prove in Section V that all nodes reach consensus to the optimal solution.

Algorithm 2 The CDGD Algorithm

```
Input: Initialization: Initialize estimates $x_i^+(0)$, $x_i^-(0)$ and
    surpluses $y_i^+(0)$, $y_i^-(0)$ at each node $i$. Define Tolerance
    value $tol$ for halting the algorithm and set $e_i(0) = tol$.
    $k = 0$.
1: while $e_i(k) \geq tol$ do {Halting is done at each node
    independently with no coordination}
2:     $k = k + 1$
3:     At each node $i$ Update Estimates using (7).
4:     Find error $e_i(k) = \|w_i(k) - w_i(k-1)\|$ where $w_i(k) =$
    $[x_i^+(k)', x_i^-(k)', y_i^+(k)', y_i^-(k)']'$ for all $1 \leq i \leq n$
5: end while
Output: $x_i^+(k)$ for the corresponding $k$ for each node
```

D. Conditions on the Network Topology

Definition 1. *$\delta(\boldsymbol{G})$ is the minimum degree of the network graph. The degree of a node $deg(i)$ is equal to the number of edges connected to node i. Hence, $\delta(\boldsymbol{G}) = \min_i deg(i)$. Let $s_i = n - |\mathcal{N}_i|$ be the number of nodes that are not connected to i.*

Condition 1. *For the coding schemes presented in [17] we can use a coding scheme with the allowed number of stragglers s satisfying $\lfloor \frac{n}{2} \rfloor \geq s \geq \max_i s_i$. This means that $\delta(\boldsymbol{G}) \geq n - s - 1$. It is easy to check that $\delta(\boldsymbol{G}) \geq \lfloor \frac{n}{2} \rfloor$.*

IV. Main Fundamental Theorem

A. Bounds on ϵ, characteristics of O and convergence of the algorithm:

Let us define matrices Q and F as

$$Q = \begin{pmatrix} A_{fit} & 0_{2n\times 2n} \\ I_{2n\times 2n} - A_{fit} & \begin{pmatrix} D & 0_{n\times n} \\ 0_{n\times n} & D \end{pmatrix} \end{pmatrix}$$

and

$$F = \begin{pmatrix} 0_{2n\times 2n} & I_{2n\times 2n} \\ 0_{2n\times 2n} & -I_{2n\times 2n} \end{pmatrix}.$$

Then $O = Q + \epsilon F$.
And for a future reference, let us define

$$\bar{\epsilon} = \frac{1}{(20+8n)^n}(1 - |\lambda_4|)^n, \tag{10}$$

where λ_4 is the fourth largest eigenvalue of matrix Q.

Lemma 1. *The matrix Q has spectral radius equal to 1 and eigenvalue 1 is a semi-simple eigenvalue of multiplicity 3. That is, eigenvalues $1 = |\lambda_1| = |\lambda_2| = |\lambda_3| > |\lambda_4| > ... > |\lambda_{4n}|$.*

Proof: We follow a similar analysis as [3] modified to fit our case, that we refrain from mentioning it here due to the limited space.

Theorem 1. *Suppose that the graph $\mathbf{G}$ of the network is strongly connected and O is the matrix defined in (5) with the parameter ϵ satisfying $\epsilon \in (0, \bar{\epsilon})$ where $\bar{\epsilon}$ is defined in (10), with λ_4 the fourth largest eigenvalue of matrix O by setting $\epsilon = 0$. Then*
(a) $\lim_{k\to\infty} O^k \to P$. Specifically, $\lim_{k\to\infty} O_1^k = 1_{2n\times 1}\pi'$ and $\lim_{k\to\infty} O_2^k = 0$, where $O_1 = [O]_{[1:2n][1:2n]} = A_{fit}$ and $O_2 = [O]_{[1:2n][2n+1:4n]} = \epsilon I_{2n\times 2n}$.
(b) For all $i, j \in V$, $[O^k]_{ij}$ converge to P as $k \to \infty$ at a geometric rate. That is, $\|[O^k] - P\| \leq \Gamma\gamma^k$ where $0 < \gamma < 1$ and $\Gamma > 0$.
(c) $\bar{\epsilon}$ is a necessary and sufficient bound such that for every $\epsilon < \bar{\epsilon}$ we have (a) and (b) above.

Proof : We begin by subsequently proving the following lemmas to get the above result.

Lemma 2. *If the parameter $\epsilon \in (0, \bar{\epsilon})$ with $\bar{\epsilon}$ defined in (10) where λ_4 is the fourth largest eigenvalue of matrix Q, then $1 > |\lambda_4(\epsilon)|, ..., |\lambda_{4n}(\epsilon)| > 0$, the eigenvalues corresponding to matrix O.*

Proof: We follow a similar analysis as [3] modified to fit our case, that we refrain from mentioning it here due to the limited space.

Lemma 3. *(a) The changes of the semi-simple eigenvalue $\lambda_1 = \lambda_2 = \lambda_3 = 1$ of Q under a small perturbation ϵF are $\frac{d\lambda_1(\epsilon)}{d\epsilon} = 0$, $\frac{d\lambda_2(\epsilon)}{d\epsilon} = -1$ and $0 > \frac{d\lambda_3(\epsilon)}{d\epsilon} \geq -2n^2$, respectively.*
(b) Particularly, for $\epsilon \leq \bar{\epsilon}$ where $\bar{\epsilon} = \frac{1}{(20+8n)^n}(1 - |\lambda_4|)^n$ then $1 \approx |\lambda_1(\epsilon)| \geq |\lambda_2(\epsilon)| \geq |\lambda_3(\epsilon)| > 0$.

We omit the proof here as it uses the method provided in Chapter 2 [15].

We refer the reader to Theorem 2.8 in [16] for its use in the next lemma.

Lemma 4. *If $\epsilon \leq \bar{\epsilon}$ where $\bar{\epsilon}$ defined in (10), the eigenvalues corresponding to the dominant factor are $\lambda_1(\epsilon) \approx 1$, $\lambda_2(\epsilon) \approx 1$, $\lambda_3(\epsilon) \approx 1$ and the invariant subspace corresponding to the first three dominant eigenvalues are almost the same.*

Lemma 5. *Assume that the network graph $\mathbf{G}$ is strongly connected and O is the updating matrix of the algorithm defined in (5). Then*
(a) $lim_{k\to\infty} O^k \to P$.
(b) For all $i, j \in$, $[O^k]_{ij}$ converge to P as $k \to \infty$ at a geometric rate. That is, $\|[O^k] - P\| \leq \Gamma\gamma^k$, where $0 < \gamma < 1$ and $\Gamma > 0$.

Proof : The algebraic and geometric multiplicities of eigenvalue 1 of matrix O are equal to 3. Since Lemma states that all eigenvalues have moduli less or equal to 1. Assume the left and right eigenvectors of eigenvalue 1 are v_1, v_2, v_3 and u_1, u_2, u_3 respectively. And $v_4, v_5, ..., v_{4n}$ and $u_4, u_5, ..., u_{4n}$

are the left and right eigenvectors of eigenvalues $\lambda_4, \lambda_5, ..., \lambda_{4n}$ counted without multiplicity.

Then, O represented in Jordan Form decomposition is such that $\|O^k - P\| = \|\sum_{i=4n-3}^{4n} P_i J_i^k O_i\| \leq \sum_{i=4n-3}^{4n} \|P_i\| \|J_i^k\| \|O_i\| \leq \Gamma\gamma^k$ where $\Gamma < \infty$ and $\gamma \in (0,1)$. And $lim_{k\to\infty} O^k = lim_{k\to\infty} Q^k = P$. □

Lemma 6. *For $0 < \epsilon \leq \bar{\epsilon}$ where $\bar{\epsilon}$ is defined in (10), then $\lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P$. That is, $\hat{\epsilon}$ is a necessary bound for $\lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P$.*

Proof : This follows from the contrapositive of Lemma 2. □

Lemma 7. *For $0 < \epsilon < \hat{\epsilon} \leq \bar{\epsilon}$ where $\bar{\epsilon}$ is defined in (10), $\hat{\epsilon}$ is a sufficient bound for $\lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P$.*

Proof : This follows from Lemmas 1-5. □

Lemma 8. *For $0 < \epsilon \leq \bar{\epsilon}$ where $\bar{\epsilon}$ is defined in (10), then $\lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P$. And $\bar{\epsilon}$ is a necessary and sufficient bound for $\lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P$.*

Proof : This follows from Lemmas 6 and 7. □
That is, $P = \lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k$. But

$$P = \lim_{k\to\infty} O^k = \lim_{k\to\infty} Q^k = P + \lim_{k\to\infty} \sum_{i=4n-3}^{4n} P_i J_i^k Q_i \tag{11}$$

$$= \lim_{k\to\infty} \begin{pmatrix} A_{fit} & 0 \\ I_{2n\times 2n} - A_{fit} & \begin{pmatrix} D & 0 \\ 0 & D \end{pmatrix} \end{pmatrix}^k$$

That is

$$P = \begin{pmatrix} \lim_{k\to\infty} A_{fit}^k & 0 \\ P_3 & \begin{pmatrix} P_4 \end{pmatrix} \end{pmatrix}$$

But A_{fit} is a row stochastic matrix, then $\lim_{k\to\infty} A_{fit}^k = 1_{2n\times 1}\pi^{'}$ which is of dimension $2n \times 2n$ of rank 1 (repeated row π) where π is the stationary state of matrix A_{fit}. This result is needed in Lemma 10 and 12, that is $[P]_{jl} = [P]_{ql} = \pi_l$ for $1 \leq j, q \leq 2n$. Therefore, we have proved Theorem 1 which is the main edifice for the validity of our algorithm. □

Therefore, by using Lemma 2 to Lemma 8 we prove Theorem 1.

V. Convergence Analysis

A. Auxiliary Variables Definitions

Using (7), we define $\hat{z}_l(k) = \sum_{i=1}^{4n} [O]_{li} z_i(k)$. Then

$$\hat{z}_l(k+1) = \sum_{i=1}^{4n} [O]_{li} \sum_{j=1}^{4n} [O]_{ij} z_j(k) - \alpha_k \sum_{i=1}^{4n} [O]_{li} \nabla \bar{g}_i(z_i(k))$$
$$= \sum_{i=1}^{4n} [O]_{li} \hat{z}_i(k) - \alpha_k \nabla f_{\jmath}(z_l(k)),$$

where $\nabla f_{\jmath}(z_l(k)) = \sum_{j\in\Gamma_l} d_j(k)$ i.e. $\nabla f_{\jmath}(z_l(k)) = \sum_{j\in\Gamma_l} A_{fit(l)_n,j} \nabla \bar{g}_j(z_j(k))$ for $1 \leq l \leq 2n$. Thus,

$$\hat{z}_l(k+1) = \sum_{i=1}^{4n} [O]_{li} \hat{z}_i(k) - \alpha_k \sum_{j\in\Gamma_l} A_{fit(l)_n,j} \nabla \bar{g}_j(z_j(k)) \tag{14}$$

Next, we define $\hat{\hat{z}}_l(k) = \sum_{i=1}^{4n} [P]_{li} \hat{z}_i(k)$. Then

$$\hat{\hat{z}}_l(k+1) = \sum_{i=1}^{4n} [P]_{li} \hat{z}_i(k) - \alpha_k \sum_{i=1}^{4n} [P]_{li} \sum_{q\in\Gamma_i} A_{fit(i)_n,q} \nabla \bar{g}_q(z_q(k)) \tag{15}$$

for $1 \leq l \leq 2n$ by having $P = \lim_{k\to\infty} O^k$ and $[P]_{i,j} = 0$ for $1 \leq i \leq 2n$ and $2n+1 \leq j \leq 4n$

Remark 2. *We define $A_{fit(i)_n,j}$ to mean the entry of A_{fit} in the row corresponding to node i and column corresponding to node j. Then we have $\sum_{j\in\Gamma_i} A_{fit(i)_n,j}$ which makes it an easier notation for the analysis of the proof. Although we could have used $\sum_j [A_{fit}]_{ij^{'}}$ where $j^{'} = j$ for $1 \leq j^{'} \leq n$ and $j^{'} = j + n$ for $n+1 \leq j^{'} \leq 2n$ to mean the same quantity. Note that in A_{fit} a node i corresponds to two rows: row i and row $i+n$.*

B. Convergence Theorems

Theorem 2. *Suppose that the graph **G** of the network is strongly connected and O is the matrix defined in (5) with the parameter ϵ satisfying $\epsilon \in (0, \bar{\epsilon})$ where $\bar{\epsilon}$ is defined in (10). Then the algorithm defined by $z_i(k+1) = \sum_{j=1}^{4n} [O]_{ij} z_j(k) - \alpha_k \nabla \bar{g}_i(k)$ converges to the optimal result and consensus over all nodes. That is, for $1 \leq i, j \leq 2n$, $\lim_{k\to\infty} f(z_i(k)) = \lim_{k\to\infty} f(z_j(k)) = f^*$.*

Proof : We begin by subsequently proving the following lemmas to get the above result.

Lemma 9. *Let **Assumption 1** holds, then the sequence $\hat{\hat{z}}_j(k)$ for $1 \leq j \leq 2n$, defined earlier follows*

$$\begin{aligned} \|\hat{\hat{z}}_j(k+1) - x\|^2 &\leq \|\hat{\hat{z}}_j(k) - x\|^2 \\ &+ 2\alpha_k \sum_{i=1}^{4n} [P]_{ji} \sum_{q\in\Gamma_i} \|A_{fit(i)_n,q} \nabla \bar{g}_q(z_q(k))\| \|\hat{\hat{z}}_j(k) - z_j(k)\| \\ &+ 4\alpha_k \sum_{i=1}^{4n} [P]_{ji} \sum_{q\in\Gamma_i} G A_{fit(i)_n,q} \|z_j(k) - z_q(k)\| \\ &- 2\alpha_k \sum_{i=1}^{4n} [P]_{ji} \sum_{q\in\Gamma_i} (A_{fit(i)_n,q} \bar{g}_q(z_j(k)) - A_{fit(i)_n,q} \bar{g}_q(x)) \\ &+ \alpha_k^2 \|\sum_{i=1}^{4n} [P]_{ji} \sum_{q\in\Gamma_i} A_{fit(i)_n,q} \nabla \bar{g}_q(z_q(k))\|^2 \end{aligned} \tag{16}$$

We omit the proof here due to the space limits.

Lemma 10. *Let **Assumption 1** holds. Then $z_j(k)$ and $\hat{\hat{z}}_j(k)$ satisfy the following bounds:*
For $1 \leq j \leq 2n$ and $k \geq 1$

$$\begin{aligned} \|\hat{\hat{z}}_j(k) - z_j(k)\| &\leq \sum_{l=1}^{4n} \|z_l(0)\| \Gamma \gamma^k \\ &+ 4n \sum_{r=1}^{k-1} \Gamma \gamma^{k-r} \alpha_{r-1} \|A_{fit(i)n}\|_{2,\infty} G \\ &+ 2 \sum_{r=1}^{k-1} \alpha_{r-1} \|P\|_{2,\infty} \|A_{fit(i)n}\|_{2,\infty} G \\ &+ \alpha_{k-1} \|P\|_{2,\infty} \|A_{fit(i)n}\|_{2,\infty} G + \alpha_{k-1} G \end{aligned} \tag{17}$$

Proof : We have

$$z_i(k) = \sum_{j=1}^{4n} [O^k]_{ij} z_j(0) - \sum_{r=1}^{k} \sum_{j=1}^{4n} [O^{k-r}]_{ij} \alpha_{r-1} \nabla \bar{g}_j(z_j(r-1)) \tag{18}$$

Then by using (7) and (15) successively we get

$$\begin{aligned} &\|\hat{\bar{z}}_j(k) - z_j(k)\| \leq \|\sum_{l=1}^{4n} z_l(0)([P]_{jl} - [O^k]_{jl})\| \\ &+ \sum_{r=1}^{k-1}\sum_{l=1}^{4n} \|[P]_{jl} - [O^{k-r}]_{jl}\| \|\alpha_{r-1} \sum_{q\in\Gamma_l} A_{fit(l)_n,q} \nabla \bar{g}_l(z_l(r-1))\| \\ &+ \sum_{r=1}^{k-1}\sum_{l=1}^{4n} \|[P]_{jl}\| \|\alpha_{r-1} \sum_{q\in\Gamma_l} A_{fit(l)_n,q} (\nabla \bar{g}_q(z_q(r-1)) - \nabla \bar{g}_l(z_l(r-1)))\| \\ &+ \alpha_{k-1}\| \sum_{l=1}^{4n} [P]_{jl} \sum_{q\in\Gamma_l} A_{fit(l)_n,q} \nabla \bar{g}_q(z_q(k-1))\| + \alpha_{k-1}\|\nabla \bar{g}_j(z_j(k-1))\| \end{aligned} \tag{19}$$

since $[P]_{jl} - [O^k]_{jl} \leq \Gamma\gamma^k$ (see **Theorem 1 (b)**) and using the bounds on $\|A_{fit}\|_{2,\infty}, \|B\|_\infty, \|\nabla \bar{g}_j\|, \|P\|_{2,\infty}$ (in fact P is doubly stochastic so $\|P\|_\infty = 1$) and $z_l(0)$ the result follows. □

Lemma 11. *Let* ***Assumption 1*** *holds. Then for* $1 \leq j \leq 2n$ *we have* $\sum_{k=0}^{\infty} \alpha_k \|\hat{\bar{z}}_j(k) - z_j(k)\| < \infty$.

Proof : Using (19) from Lemma 10 and multiplying each term by α_k and summing the terms over k from 0 to ∞, then using the bounds below we have

$$\sum_{k=0}^{K} \alpha_k \gamma^k \leq \frac{1}{2}\sum_{k=0}^{K}(\alpha_k^2 + \gamma^{2k}) \leq \sum_{k=0}^{K} \frac{1}{2}\alpha_k^2 + \frac{1}{2}\frac{1}{1-\gamma^2} < \infty$$

since $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$ and $0 < \gamma < 1$. Similarly,

$$\begin{aligned} \sum_{k=0}^{K}\sum_{r=1}^{k-1} \alpha_k \alpha_{r-1} \gamma^{(k-r)} &< \frac{1}{2}\sum_{k=0}^{K}\alpha_k^2 \sum_{r=1}^{k-1}\gamma^{(k-r)} \\ &+ \frac{1}{2}\sum_{r=1}^{K-1}\alpha_{r-1}^2 \sum_{k=r+1}^{K}\gamma^k \leq \frac{1}{1-\gamma}\sum_{k=0}^{K}\alpha_k^2 \end{aligned}$$

Thus,

$$\sum_{k=0}^{\infty}\sum_{r=1}^{k-1} \alpha_k \alpha_{r-1}\gamma^{(k-r)} \leq \frac{1}{1-\gamma}\sum_{k=0}^{K}\alpha_k^2 < \infty$$

Same for $\sum_{k=0}^{K}\sum_{r=1}^{k-1}\alpha_k\gamma^{2(k-r)} < \infty$ and $\sum_{k=0}^{K}\sum_{r=1}^{k-1}\alpha_k\alpha_{k-1} < \infty$ and $\|z_l(0)\|$ bounded. (By initialization in a bounded space.)

Also P, A and B are fixed thus $\|P\|_{2,\infty}, \|B\|_\infty and \|A\|_{2,\infty}$ are bounded. More precisely $\|P\|_\infty = 1$ (doubly stochastic matrix) and $\|A_{fit(i)n}\|_\infty = 1$ (row normalized matrix). Thus the proof follows. □

Lemma 12. *Let* ***Assumption 1*** *holds. Then*
(a) For $1 \leq j, q \leq 2n$ *we have*

$$\sum_{k=0}^{\infty} \alpha_k \|z_j(k) - z_q(k)\| < \infty, and \tag{20}$$

(b)

$$\lim_{k\to\infty} \|z_j(k) - z_q(k)\| = 0 \tag{21}$$

That is $\lim_{k\to\infty} z_j(k) = \lim_{k\to\infty} z_q(k)$ *for all* j *and* q *(i.e.* $1 \leq j, q \leq 2n$*)*

Proof : For (a),

$$\begin{aligned} \sum_{k=0}^{\infty}\alpha_k\|z_j(k) - z_q(k)\| &\leq \sum_{k=0}^{\infty}\alpha_k\sum_{l=1}^{4n}\|[O^k]_{jl} - [P]_{jl}\|\|z_l(0)\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\sum_{l=1}^{4n}\|[P]_{jl} - [P]_{ql}\|\|z_l(0)\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\sum_{l=1}^{4n}\|[O^k]_{ql} - [P]_{ql}\|\|z_l(0)\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\sum_{r=1}^{k-1}\sum_{l=1}^{4n}\|[O^{k-r}]_{jl} - [P]_{jl}\|\alpha_{r-1}\|\nabla\bar{g}_l(z_l(r-1))\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\sum_{r=1}^{k-1}\sum_{l=1}^{4n}\|[P]_{jl} - [P]_{ql}\|\alpha_{r-1}\|\nabla\bar{g}_l(z_l(r-1))\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\sum_{r=1}^{k-1}\sum_{l=1}^{4n}\|[O^{k-r}]_{ql} - [P]_{ql}\|\alpha_{r-1}\|\nabla\bar{g}_l(z_l(r-1))\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\alpha_{k-1}\|\nabla\bar{g}_j(z_j(k-1)) - \nabla\bar{g}_q(z_q(k-1))\| \end{aligned} \tag{22}$$

But $[P]_{jl} = [P]_{ql}$ for $1 \leq j, q \leq 2n$ and $1 \leq l \leq 2n$. (see **Theorem 1 (a)**)

Then, the above becomes

$$\begin{aligned} \sum_{k=0}^{\infty}\alpha_k\|z_j(k) - z_q(k)\| &\leq 2\sum_{k=0}^{\infty}\alpha_k\sum_{l=1}^{4n}\Gamma\gamma^k\|z_l(0)\| \\ &+ 2\sum_{k=0}^{\infty}\alpha_k\sum_{r=1}^{k-1}\sum_{l=1}^{4n}\Gamma\gamma^{k-r}\alpha_{r-1}\|\nabla\bar{g}_l(z_l(r-1))\| \\ &+ \sum_{k=0}^{\infty}\alpha_k\alpha_{k-1}\|\nabla\bar{g}_j(z_j(k-1)) - \nabla\bar{g}_q(z_q(k-1))\|. \end{aligned} \tag{23}$$

Similarly using bounds as in the proof of Lemma 11 we get

$$\sum_{k=0}^{\infty}\alpha_k\|z_j(k) - z_q(k)\| < \infty. \tag{24}$$

Thus (a) follows.
For (b), using a modified expression of (22) without the summation over α_k and having,
$\alpha_k \to 0$ and $\gamma^k \to 0$ as $k \to \infty$ and $\|\nabla\bar{g}_j(k)\| \leq G$ then,

$$\lim_{k\to\infty}\|z_j(k) - z_q(k)\| \to 0. \tag{25}$$

Thus, (b) follows. □

Remark 3. *The use of* **Theorem 1 (a)** *in Lemma 12 parts (a) and (b) restricts j, q to be* $1 \leq j, q \leq 2n$ *for the results to follow.*

Lemma 13. *Let* ***Assumption 1*** *holds. Then*

$$2\sum_{k=0}^{\infty}\alpha_k \sum_{i=1}^{4n}[P]_{ji}\sum_{q\in\Gamma_i}(A_{fit(i)_n,q}\bar{g}_q(z_j(k)) - A_{fit(i)_n,q}\bar{g}_q(x)) \leq$$
$$\|\hat{\bar{z}}_j(0)-x\|^2$$
$$+\sum_{k=0}^{\infty}2\alpha_k\sum_{i=1}^{4n}[P]_{ji}\sum_{q\in\Gamma_i}\|A_{fit(i)_n,q}\nabla\bar{g}_q(z_q(k))\|\|\hat{\bar{z}}_j(k)-z_j(k)\|$$
$$+4\sum_{k=0}^{\infty}\alpha_k\sum_{i=1}^{4n}[P]_{ji}\sum_{q\in\Gamma_i}GA_{fit(i)_n,q}\|z_j(k)-z_q(k)\|$$
$$\sum_{k=0}^{\infty}\alpha_k^2\|\sum_{i=1}^{4n}[P]_{ji}\sum_{q\in\Gamma_i}A_{fit(i)_n,q}\nabla\bar{g}_q(z_q(k))\|^2 \tag{26}$$

Proof : Using Lemma 9 and summing from $k=0$ to ∞ the result follows. □

Lemma 14. *From Lemmas 11, 12(a), 12(b) and 13 and Assumption 1(d), we have for* $1 \leq i,j \leq 2n$, $\lim_{k\to\infty} f(z_i(k)) = \lim_{k\to\infty} f(z_j(k)) = f^*$.

Proof : Take $x = x^*$ the optimal value, in Lemma 13. Inspecting the RHS of the inequality of (26), we have: $\|\hat{\bar{z}}_j(0) - x^*\|^2 < \infty$ as the initial estimate and the solution are fixed (Bounded space). And since from Lemma 11 we have $\sum_{k=0}^{\infty}\alpha_k\|z_j(k) - \hat{\bar{z}}_j(k)\| < \infty$ for the involved estimates $1 \leq i \leq 2n$. And $\sum_{k=0}^{\infty}\alpha_k\|z_j(k) - z_q(k)\| < \infty$ from Lemma 12(a). And the fourth term of (26) bounded by Assumption 1(d) and the fixed P and A_{fit}. Then we get that the LHS is finite, that is $2\sum_{k=0}^{\infty}\alpha_k\sum_{i=1}^{4n}[P]_{ji}\sum_{q\in\Gamma_i}(A_{fit(i)_n,q}\bar{g}_q(z_j(k)) - A_{fit(i)_n,q}\bar{g}_q(x^*)) < \infty$.

But from Lemma 12(b) we have $z_j(k) = z_q(k)$ for $1 \leq j,q \leq 2n$ and we have $f(z_j(k)) = \sum_{q\in\Gamma_i}(A_{fit(i)_n,q}\bar{g}_q(z_j(k))$. Similarly, $f(x^*) = \sum_{q\in\Gamma_i}A_{fit(i)_n,q}\bar{g}_q(x^*)$. But for $k \geq 0$, we have $f(z_j(k)) - f(x^*) \geq 0$ and $\sum_{k=0}^{\infty}\alpha_k = \infty$ and from what preceeded $2\sum_{k=0}^{\infty}\alpha_k\sum_{i=1}^{4n}[P]_{ji}(f(z_j(k)) - f(x^*)) < \infty$ then we get $\lim_{k\to\infty}\inf(f(z_j(k)) - f(x^*)) = 0$. Thus, $\lim_{k\to\infty} f(z_j(k)) = f^*$. □

VI. RATE OF CONVERGENCE

After elaborating more on Lemma 9 and knowing that $\sum_{k=0}^{K}\alpha_k(f_{min} - f(x^*)) \leq \sum_{k=0}^{K}\alpha_k(f(z_i(k)) - f(x^*))$, where $f_{min} = min_{0\leq k\leq K}f(z_i(k))$, we get the following :

$$f_{min} - f(x^*) \leq \frac{A_*}{\sum_{k=0}^{K}\alpha_k} + \frac{B_*\sum_{k=0}^{K}\alpha_k^2}{\sum_{k=0}^{K}\alpha_k} \tag{27}$$

where

$$A_* = \frac{1}{2\|P\|_{2,\infty}}(dist^2(\hat{\bar{z}}(0),\mathcal{X}^*) - \frac{1}{2\|P\|_{2,\infty}}dist^2(\hat{\bar{z}}(k),\mathcal{X}^*)$$
$$+\frac{5}{2}\frac{\|A_{fit}\|_{2,\infty}\Gamma\|B\|_{2,\infty}\sqrt{n}F}{1-\gamma^2}\sum_{l=1}^{4n}\|z_l(0)\| \tag{28}$$

and

$$B_* = \frac{7}{2}\|A_{fit}\|_{2,\infty}\|B\|^2_{2,\infty}nF^2(\|P\|_{2,\infty}\|\|A_{fit}\|_{2,\infty} + \frac{10}{7})$$
$$4n\|A_{fit}\|_{2,\infty}\|B\|^2_{2,\infty}nF^2\frac{\Gamma}{1-\gamma}(\|A_{fit}\|_{2,\infty}+4) \tag{29}$$
$$+3\|A_{fit}\|_{2,\infty}\|B\|_{2,\infty}\sqrt{n}F\Gamma\sum_{l=1}^{4n}\|z_l(0)\|$$

where we used $G = \sqrt{n}\|B\|_{2,\infty}F$. For $\alpha_k = \frac{1}{\sqrt{k}}$ then the convergence rate is a scaled coefficient adequate to the coding scheme/network topology of rate $\mathbf{O}(\frac{\ln \mathbf{k}}{\sqrt{\mathbf{k}}})$.

Therefore, the convergence rate of this algorithm is a scaled version of the convergence rate of the distributed gradient descent (DGD) presented in [12, 18] (for directed and undirected graphs) with a scaling factor depending on the considered adapted gradient coding scheme. Thus, we can perfectly adjust this scaling factor according to a desired coding scheme so that the algorithm is tuned to perform better than DGD by that factor yet still under $\mathbf{O}(\frac{\ln \mathbf{k}}{\sqrt{\mathbf{k}}})$ for $\alpha_k = \frac{1}{\sqrt{k}}$.

However, from the norm inequality

$$\|M\|_F \leq \sqrt{n}\|M\|_{2,\infty} \tag{30}$$

we have the minimum of $\|M\|_{2,\infty}$ attained when $M = \frac{1}{n}\mathbf{1}^{'}\mathbf{1}$ where $\|M\|_F = 1$ and $\|M\|_{2,\infty} = \frac{\|M\|_F}{\sqrt{n}} = \frac{1}{\sqrt{n}}$.

But A_{fit} and P are stochastic matrices (particularly row normalized). The first relative to n and the latter relative to $2n$ although the sizes are $2n\times 2n$ and $4n\times 4n$ respectively.(i.e., P involved is for $1 \leq i \leq 2n$ rows and the corresponding nonzero columns are $1 \leq j \leq 2n$). Therefore, $\frac{1}{\sqrt{n}} \leq \|A_{fit}\|_{2,\infty} \leq 1$, and $\frac{1}{\sqrt{2n}} \leq \|P\|_{2,\infty} \leq 1$.

Thus the scaling in (28) and (29) can be adjusted to be less than one. Hence, a better convergence rate than DGD.

Moreover, if we implement our algorithm but with no coding(no redundancy (i.e., local functions f_i) where we have $\|B\|_{2,\infty} = 1$ and $\|A_{fit}\|_{2,\infty} = \frac{1}{\sqrt{n}}$, we are still able to achieve a better convergence rate than DGD although our updating matrix can be of a larger size ($4n$ rather than n). We can reach that by suitably choosing the updating matrix O so that its limit P has a value of $\|P\|_{2,\infty}$ as close as possible as $\frac{1}{\sqrt{2n}}$.

VII. SIMULATION RESULTS

The purpose of this simulation is to examine the convergence rate of our proposed algorithm for different network topologies. Two types of performance metrics are used given by

$Estimation\ Error := \|x^* - x_i(k)\|$

$Consecutive\ Error := \|x_i(k+1) - x_i(k)\|$

In Fig. 1 we compare CDGD with the conventional incremental gradient descent on a distributed network with $n = 21$ nodes, where the step size is $\alpha_k = \frac{1}{(k+10000)^{\frac{1}{2}}}$. The estimation error of our approach decreases significantly in comparison with the other algorithm. Specifically, after a slight bump increase in the estimation error due to large nonadaptive step sizes α_k for the initial hundred iterates ($1 < k < 100$), the algorithm error decreases significantly. This shows that our method achieves a better convergence rate. Moreover, the consecutive error of our method decreases gradually.

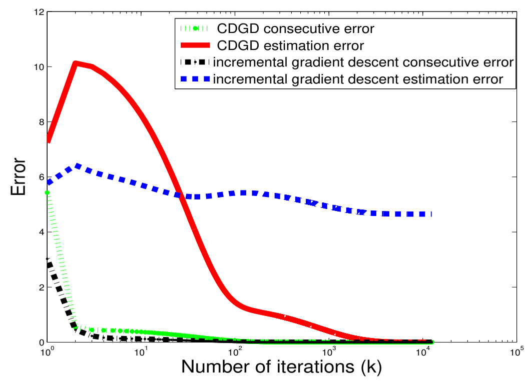

Fig. 1. Estimation and consecutive errors vs iteration of CDGD for a 21-node decentralized network.

In Fig. 2 we compare the convergence rate of CDGD to the incremental gradient descent on a $n = 7$ node network with the step size $\alpha_k = \frac{1}{(k+100000)^{\frac{1}{2}}}$. From the plot we can verify the closeness of our algorithm to the expected rate at this step size taking into consideration the scaling factor. As shown in the figure the consecutive error of the incremental gradient descent is the most decreasing, while its estimation error is the least. However, the closeness of the consecutive error to the estimation error for our method verifies the use of this first error as an indicator measure to reaching the exact solution. Moreover, our algorithm has a better convergence rate, which is apparent in the figure.

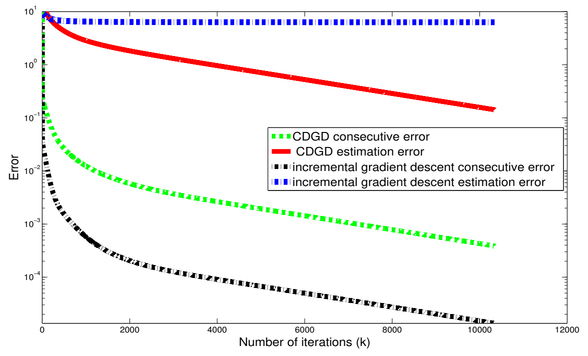

Fig. 2. Estimation and consecutive errors vs iteration of CDGD for a 7-node decentralized network.

VIII. Conclusion

We presented in this paper a distributed decentralized algorithm for minimizing convex functions using coded local gradients referred to as Code-Based Distributed Gradient Descent (CDGD). Each type of network topology is identified with a corresponding coding scheme that adapts the coded gradients in an attempt to enhance the convergence rate. A proof for the convergence of this algorithm was provided that relies implicitly on the structure of the updating matrix. In implementing such approach, we matched the convergence rates of $\mathbf{O}(\frac{\mathbf{lnk}}{\sqrt{\mathbf{k}}})$ in compliance with the uncoded incremental gradient methods [6, 11]. However, by adapting a convenient coding scheme we enhanced our convergence rate over conventional methods. This was achieved by affecting the constant scaling factor without using any predefined tuning. Detailed analysis of this work will be presented in an upcoming publication, where the more relaxed assumption of requiring only the global function f to be convex is used as mentioned earlier.

References

[1] Bedi, A. S. and K. Rajawat (2018). Asynchronous incremental stochastic dual descent algorithm for network resource allocation. *IEEE Transactions on Signal Processing 66*(9), 2229–2244.

[2] Bertsekas, D. P. (2011). Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. *Optimization for Machine Learning 2010*(1-38), 3.

[3] Cai, K. and H. Ishii (2012). Average consensus on general strongly connected digraphs. *Automatica 48*(11), 2750–2761.

[4] Dean, J., G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le, et al. (2012). Large scale distributed deep networks. In *Advances in neural information processing systems*, pp. 1223–1231.

[5] Fu, D., L. Han, L. Liu, Q. Gao, and Z. Feng (2015). An efficient centralized algorithm for connected dominating set on wireless networks. *Procedia Computer Science 56*, 162–167.

[6] GÜRBÜZBALABAN, M., A. OZDAGLAR, and P. PARRILO (2015). Convergence rate of incremental gradient and incremental newton methods. *arXiv preprint arXiv:1510.08562*.

[7] Halbawi, W., N. Azizan-Ruhi, F. Salehi, and B. Hassibi (2017). Improving distributed gradient descent using reed-solomon codes. *arXiv preprint arXiv:1706.05436*.

[8] Ho, Q., J. Cipar, H. Cui, S. Lee, J. K. Kim, P. B. Gibbons, G. A. Gibson, G. Ganger, and E. P. Xing (2013). More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in neural information processing systems*, pp. 1223–1231.

[9] Johansson, B. (2008). *On distributed optimization in networked systems*. Ph. D. thesis, KTH.

[10] Li, M., D. G. Andersen, A. J. Smola, and K. Yu (2014). Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pp. 19–27.

[11] Nedić, A. and D. Bertsekas (2001). Convergence rate of incremental subgradient algorithms. In *Stochastic optimization: algorithms and applications*, pp. 223–264. Springer.

[12] Nedic, A. and A. Ozdaglar (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control 54*(1), 48–61.

[13] Nedic, A., A. Ozdaglar, and P. A. Parrilo (2010). Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control 55*(4), 922–938.

[14] Raviv, N., I. Tamo, R. Tandon, and A. G. Dimakis (2017). Gradient coding from cyclic mds codes and expander graphs. *arXiv preprint arXiv:1707.03858*.

[15] Seyranian, A. P. and A. A. Mailybaev (2003). *Multiparameter stability theory with mechanical applications*, Volume 13. World Scientific.

[16] Stewart, G. W. (1990). Matrix perturbation theory.

[17] Tandon, R., Q. Lei, A. G. Dimakis, and N. Karampatziakis (2016). Gradient coding. *arXiv preprint arXiv:1612.03301*.

[18] Xi, C., Q. Wu, and U. A. Khan (2017). On the distributed optimization over directed networks. *Neurocomputing 267*, 508–515.