Push-Sum Distributed Dual Averaging for Convex Optimization

Konstantinos I. Tsianos, Sean Lawlor and Michael G. Rabbat

***Abstract*— Recently there has been a significant amount of research on developing consensus based algorithms for distributed optimization motivated by applications that vary from large scale machine learning to wireless sensor networks. This work describes and proves convergence of a new algorithm called Push-Sum Distributed Dual Averaging which combines a recent optimization algorithm [1] with a push-sum consensus protocol [2]. As we discuss, the use of push-sum has significant advantages. Restricting to doubly stochastic consensus protocols is not required and convergence to the true average consensus is guaranteed without knowing the stationary distribution of the update matrix in advance. Furthermore, the communication semantics of just summing the incoming information make this algorithm truly asynchronous and allow a clean analysis when varying intercommunication intervals and communication delays are modelled. We include experiments in simulation and on a small cluster to complement the theoretical analysis.**

I. Introduction

In this work we describe and prove convergence of a new algorithm called Push-Sum Distributed Dual averaging (PS-DDA) for solving convex optimization problems of separable functions whose components are distributed over the nodes of a network. Our algorithm builds on the recently published Distributed Dual Averaging (DDA) algorithm [1] which we modify to use the Push-Sum consensus protocol [2]. PS-DDA has significant advantages. It guarantees convergence to the unbiased optimum without knowing the stationary distribution of the averaging matrix or the size of the network, and the convergence rate is the same as standard DDA. Furthermore, the communication semantics of Push-Sum make PS-DDA truly asynchronous and allow for a clean analysis when modelling varying intercommunication intervals and communication delays.

Recently there has been a significant amount of research on developing consensus based algorithms for distributed optimization motivated by applications ranging from large scale machine learning to wireless sensor networks [1], [3], [4]. In large scale machine learning, datasets may be larger than a single processor's memory. Peer-to-peer architectures are scalable and robust to single points of failure while yielding algorithms that tend to be simpler to implement. However they lack a highly organized network infrastructure, and coordinating the computing nodes becomes a challenge. Consensus algorithms have been analyzed mostly in wireless network settings [5] and provide an elegant distributed solution for coordinating the nodes.

To be of practical interest, a consensus-based distributed optimization algorithm needs to accommodate the constraints imposed by the network. For example, not all (directed) networks admit a doubly stochastic matrix [6]. However, relinquishing double stochasticity can introduce bias in the optimization [7], [8]. Moreover, it is desirable to only rely on one directional communication between nodes because in a bi-directional case where each node blocks until it receives a response, deadlocks can occur when the network has cycles. Finally, an algorithm should be able to converge in the presence of network induced communication delays and should allow for an asynchronous implementation to avoid delaying the whole computation if a particular node is very slow.

In this paper, we present and analyze PS-DDA, an algorithm that addresses all the concerns mentioned above. In the main part of the paper, we prove that a synchronous version of the algorithm converges to the unbiased optimum at a rate $O(T^{-0.5})$ where T is the number of iterations. The result holds for one directional communication using arbitrary column stochastic consensus protocols P that respect the structure of the strongly connected network. In contrast to [8], the bias in the optimization introduced by P is corrected without requiring knowledge of the stationary distribution of P or the network size. In the last part of the paper, we discuss how our convergence result can be extended to accommodate communication delays and a fully asynchronous version of the algorithm.

The rest of the paper is organized as follows. In Sections II and III we describe the standard distributed dual averaging and push-sum algorithms. Section IV introduces push-sum distributed dual averaging and states our convergence theorem. An experimental evaluation follows in Section V. We continue with a discussion of how our algorithm can be extended to handle communication delays and asynchronous communication in Section VI. The detailed convergence proof is given in Section VII before we conclude in Section VIII.

II. Distributed Dual Averaging

To make the paper self-contained, we review some necessary background on the distributed dual averaging algorithm. For more details consult [1]. Suppose we are given a strongly connected network $G = (V, E)$ of $|V| = n$ compute nodes. Each node i knows a convex function $h_i(x) : \mathbb{R}^d \rightarrow \mathbb{R}$. Our

K. I. Tsianos is a PhD candidate at the Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec H3A 2A7, Canada, konstantinos.tsianos@mail.mcgill.ca

S. Lawlor is a Master's student at the Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec H3A 2A7, Canada, sean.lawlor@mail.mcgill.ca

M.G. Rabbat is an Assistant Professor at the Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec H3A 2A7, Canada, michael.rabbat@mcgill.ca

goal is to solve the following minimization problem:

$$\begin{aligned} &\text{minimize} \quad f(x) = \frac{1}{n}\sum_{i=1}^{n} h_i(x) \\ &\text{subject to} \quad x \in \mathcal{X} \end{aligned} \tag{1}$$

where $\mathcal{X}$ is a convex set. We assume that each h_i is convex and L-Lipschitz continuous with respect to norm $\|\cdot\|$; i.e., $|h_i(x) - h_i(y)| \leq L\,\|x-y\|\,, \forall x, y \in \mathcal{X}$. As a consequence, for any $x \in \mathcal{X}$ and any subgradient $g_i \in \partial h_i(x)$ we have $\|g_i\|_* \leq L$ where $\|v\|_* = \sup_{\|u\|=1}\langle u, v\rangle$ is the dual norm.

The algorithm uses a 1-strongly convex *proximal function* $\psi : \mathbb{R}^d \to \mathbb{R}$ such that $\psi(x) \geq 0$ and $\psi(0) = 0$. Also select a non-increasing sequence of positive step sizes $\{a(t)\}_{t=0}^{\infty}$ and a doubly stochastic matrix P respecting the structure of G in the sense that $p_{ij} > 0$ only if $i = j$ or $(i, j) \in E$. The distributed dual averaging algorithm repeats, for each node i in discrete steps t, the following updates:

$$z_i(t+1) = \sum_{j=1}^{n} p_{ij} z_j(t) + g_i(t) \tag{2}$$

$$x_i(t+1) = \Pi_{\mathcal{X}}^{\psi}\left(z_i(t+1), a(t)\right) \tag{3}$$

where $g_i(t) \in \partial h_i(x_i(t))$ is a subgradient of $h_i(x)$ evaluated at $x_i(t)$, and the projection operator $\Pi_{\mathcal{X}}^{\psi}(\cdot,\cdot)$ is defined as

$$\Pi_{\mathcal{X}}^{\psi}\left(z, a\right) = \underset{x \in \mathcal{X}}{\operatorname{argmin}}\{\langle z, x\rangle + \frac{1}{a}\psi(x)\}. \tag{4}$$

In (2),(3) x_i is the local estimate at node i and z_i is a dual variable maintaining an accumulated subgradient. To update z_i, at each iteration each node needs to collect the z-values of its neighbours, form a convex combination of the received information and add its local most recent subgradient $g_i(t)$. In [1] it is proven that using this algorithm, the local running average $\hat{x}_i(T) = \frac{1}{T}\sum_{t=1}^{T} x_i(t)$ converges to the optimum.

III. Push-Sum Consensus

In [8] it is proven that restricting to doubly stochastic consensus protocols in distributed dual averaging is not necessary and it is still possible to converge to the optimum with a general row stochastic protocol P. However there are multiple reasons why using a row stochastic matrix may not be desirable. The bias correction described in [8] requires knowledge of the stationary distribution of P in advance which is restrictive. Moreover, with a time-varying consensus protocol $P(t)$, we may not even be able to specify the stationary distribution beyond its expectation and variance [9] or may only be able to achieve average consensus in expectation [10]. In this paper, we propose the use of a different one directional consensus algorithm called *Push-Sum*. A simple asynchronous version of the algorithm was first analyzed in [2] for complete graphs. In [11] convergence is proven for any graph based on weak ergodicity arguments.

In the simple (synchronous) case, given the topology of the network G, we choose a column stochastic matrix P respecting G; i.e., $p_{ij} = 0$ if there is no directed edge (j, i). If $(j, i) \in E$ we may still have $p_{ij} = 0$ meaning that although the channel is available the protocol chooses not to use it[1]. The initial values at the nodes are stacked in a vector $z(0)$ and we are looking to compute the average $z_{ave} = \frac{\mathbf{1}^T z(0)}{n}$. In Push-Sum, each node i maintains two values, a cumulative estimate of the sum $s_i(t)$ and a weight $w_i(t)$. We initialize

$$s(0) = z(0) \qquad w(0) = \mathbf{1} \tag{5}$$

and the average estimate at each iteration is the ratio $\frac{s_i(t)}{w_i(t)}$. For all iterations, a mass conservation property holds allowing the algorithm to converge to the average. At each iteration, a node j splits its total sum $s_j(t)$ and weight $w_j(t)$ into shares $S_{j\to i}(t) = \big(p_{ij}s_j(t), p_{ij}w_j(t)\big)$ where $\sum_{i=1}^{n} p_{ij} = 1$, and sends to each neighbour i the corresponding share $S_{j\to i}(t)$. A receiving node just sums up all the incoming shares from its neighbours. At each time, the estimate of the average at each node is $\hat{z}_i(t) = \frac{s_i(t)}{w_i(t)}$. In vector form the state evolves as

$$s(t) = Ps(t-1) \qquad w(t) = Pw(t-1) \tag{6}$$

$$\hat{z}(t) = \frac{s(t)}{w(t)} \tag{7}$$

where the division of $s(t)$ and $w(t)$ is element-wise. We can verify that through the updates (6), mass is conserved in the sense that

$$\sum_{i=1}^{n} s_i(t) = \sum_{i=1}^{n} z_i(0) = \mathbf{1}^T x(0) = n z_{ave} \tag{8}$$

$$\sum_{i=1}^{n} w_i(t) = n. \tag{9}$$

To see why Push-sum correctly computes the average, notice that since G is assumed strongly connected and P respects G, matrix P is a scrambling matrix and P^t converges to a rank-1 matrix exponentially fast [12], [13]. Let P^∞ be the limit of P^t as $t \to \infty$. Matrix P^∞ will be column stochastic with all columns the same. At any node i we will have

$$\hat{x}_i(\infty) = \frac{\big[P^\infty s(0)\big]_i}{\big[P^\infty w(0)\big]_i} = \frac{\big[P^\infty x(0)\big]_i}{\big[P^\infty 1\big]_i} = \frac{\sum_{j=1}^{n} p_{ij}^{\infty} x_j(0)}{\sum_{j=1}^{n} p_{ij}^{\infty}} \tag{10}$$

$$= \frac{p_{i1}^{\infty}\sum_{j=1}^{n} x_j(0)}{p_{i1}^{\infty}\sum_{j=1}^{n} 1} = \frac{\sum_{j=1}^{n} x_j(0)}{n} = \frac{\mathbf{1}^T x(0)}{n}. \tag{11}$$

We used the fact that all rows of P^∞ are the same i.e. $p_{ij}^{\infty} = p_{i1}^{\infty}, \forall j$. For a formal proof see [11]. Observe that convergence is achieved without the need to know the stationary distribution of P or the size of the network n at every node. Moreover, from (8) and (7) convergence implies that

$$\frac{s_j(t)}{w_j(t)} \to \frac{1}{n}\sum_{i=1}^{n} s_i(t). \tag{12}$$

Finally, for the case where P remains fixed, we can obtain an eigenvalue bound on the convergence rate of the

[1] As long as that does not break the strong connectivity.

algorithm. In general, matrix P represents a non-reversible, irreducible Markov chain and we have

$$\left\|\pi - \left[P^t\right]_{:,i}\right\|_1 \leq \sqrt{\frac{\lambda_2^t}{\pi_i}} \tag{13}$$

π is the stationary distribution vector of P and λ_2 is the second largest eigenvalue of the *lazy additive reversibilization* of P as explained in [14] and [13]

IV. Push-Sum Distributed Dual Averaging

Equipped with the Push-sum averaging protocol, we formulate the *Push-sum Distributed Dual Averaging* (PS-DDA) algorithm as follows:

$$w_i(t+1) = \sum_{j=1}^{n} p_{ij} w_j(t) \tag{14}$$

$$z_i(t+1) = \sum_{j=1}^{n} p_{ij} z_j(t) + g_i(t) \tag{15}$$

$$x_i(t+1) = \Pi_{\mathcal{X}}^{\psi}\left(\frac{z_i(t+1)}{w_i(t+1)}, a(t)\right) \tag{16}$$

where $g_i(t)$ is a subgradient of $h_i(x)$ at point $x = x_i(t)$ and $a(t)$ is a non-increasing sequence of step sizes. Observe that to retrieve the correct cumulative gradient of standard DDA we need to divide the z variable at every node by the appropriate weight. In section VII we prove that PS-DDA converges to the solution of (1) at the same rate as standard DDA. The end result is the following theorem:

Theorem 1: The PS-DDA algorithm (14)-(16) using a strongly convex function $\psi(x)$ with respect to norm $\|\cdot\|$ and dual norm $\|\cdot\|_*$ such that $\psi(x^*) \leq R^2$ and choosing step sizes

$$a(t) = \frac{R}{L\sqrt{1 + \frac{8+4n}{c\sqrt{\pi^*}(1-\sqrt{\lambda_2})}}} \frac{1}{\sqrt{t}}, \tag{17}$$

converges for every node $j \in V$ to the optimum $x^* \in \mathcal{X}$ of (1) as

$$f(\hat{x}_j(T)) - f(x^*) \leq 2RL\sqrt{1 + \frac{8+4n}{c\sqrt{\pi^*}(1-\sqrt{\lambda_2})}} \frac{1}{\sqrt{T}} \tag{18}$$

Where since G is a strongly connected graph and P respects the structure of G, there exists a value $c > 0$ such that for all t and all i, we have $\sum_{s=1}^{n} [P^t]_{is} \geq c$ and $\pi^* = \min_s\{\pi_s\}$ is the minimum entry in π, the stationary distribution of P. For a fixed network and consensus protocol, the convergence rate is $O(T^{-0.5})$ which is the same as standard DDA. The constant term reveals the dependence on the connectivity and specific consensus protocol through λ_2 as well as the network size n. The dependence on c and π^* is quite pessimistic and could likely be tightened.

V. Experimental Evaluation

In this section we include numerical experiments that complement the theoretical analysis. In the first experiment, we simulate the solution of a small quadratic optimization

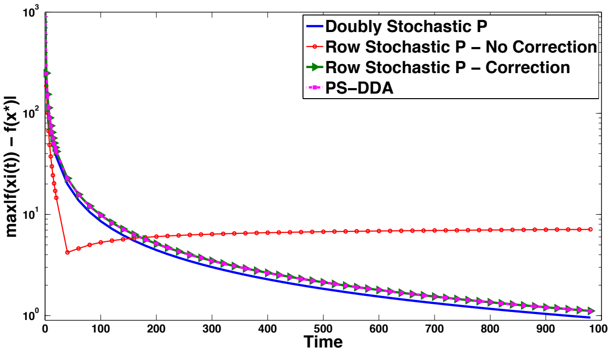

Fig. 1. Minimization of the sum of quadratics (19) with a network of 10 nodes. (Blue) Progress of standard DDA with a doubly stochastic consensus matrix. (Red) DDA using a row stochastic protocol that biases the optimization. (Green) Applying the correction from [8] removes the bias. (Purple) PS-DDA also finds the solution to the unbiased problem using the column stochastic protocol P^T. The purple and green lines in this case are overlapping.

problem for which we know the true optimum exactly. Specifically,

$$f(x) = \frac{1}{10}\sum_{i=1}^{10} h_i(x) = \sum_{i=1}^{10} (x - i\mathbf{1})^T (x - i\mathbf{1}) \tag{19}$$

so each of the $n = 10$ nodes connected using a random network G, holds a quadratic function with a different center. We have $x \in \mathbb{R}^5$ and the solution is $x^* = 5.5\mathbf{1}$ with $f(x^*) = 412.5$. Figure 1 shows the performance of different algorithms. We measure the maximum distance to the optimal function value among the nodes i.e., $\max_i |f(x_i(t)) - f(x^*)|$. The blue curve shows standard distributed dual averaging which uses a doubly stochastic matrix. Next, we generate a row stochastic matrix P that respects G and has a biased stationary distribution i.e., $\pi = [0.08\ 0.07\ 0.08\ 0.05\ 0.07\ 0.11\ 0.24\ 0.14\ 0.09\ 0.07]^T$ so that node 7 gets disproportionately more weight than the rest. By simply using the consensus protocol P we find the solution of the biased optimization problem $\tilde{f}(x) = \sum_{i=1}^{10} \pi_i h_i(x)$ as the red curve shows. Knowing π and n, we apply the bias correction suggested in [8] and retrieve the solution of the unbiased optimization problem (green curve). Finally, to simulate PS-DDA we just use the column stochastic P^T as our consensus protocol. The purple curve corresponding to PS-DDA overlaps in this particular case with the the green curve and also retrieves the unbiased solution.

In our second experiment, we solve a larger problem with a small real cluster. We arrange 15 nodes in an arbitrary network network topology (Figure 3)and assign to each node i a function

$$h_i(x) = \sum_{j=1}^{M}\left(x - \left(i + \frac{j}{2M}\right)\mathbf{1}\right)^T\left(x - \left(i + \frac{j}{2M}\right)\mathbf{1}\right) \tag{20}$$

where $x \in \mathbb{R}^{100}$ and $M = 10,000$. The minimization of a sum of quadratics is typical in machine learning where we are minimizing a quadratic loss function from

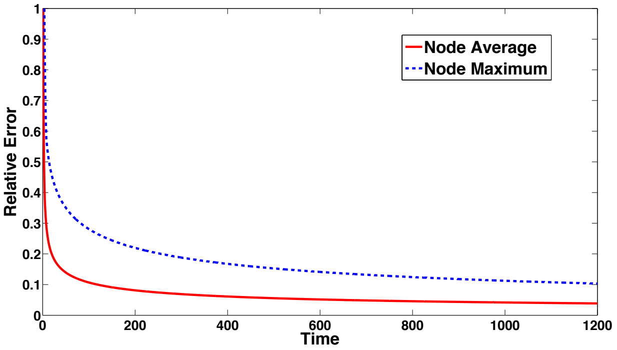

Fig. 2. Minimization of a sum of $150,000$ quadratics among 15 processors in an MPI cluster using PD-DDA. (Blue) The maximum relative error between the estimate $x_i(t)$ at any node i and the true optimum x^*. (Red) Average relative error.

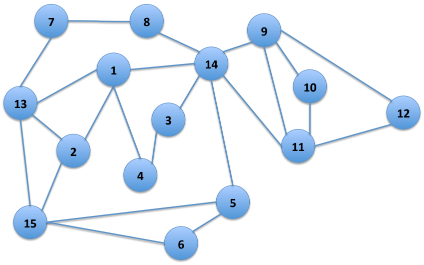

Fig. 3. We organize a small MPI cluster or $n = 15$ nodes as an arbitrary sparse graph for our second experiment. This is a logical topology and does not match the underlying physical network connections.

(in our case) $15M$ data points split evenly among the processors. It is worth emphasizing that for this experiment, although the data is synthetic and we know the solution x^*, the algorithm is running on distributed hardware and operating under real network conditions. The different processors communicate asynchronously using the Message Passing Interface(MPI [15]) over ethernet connections and there is no control over network induced communication delays. Figure 2 shows the evolution of the maximum (red) relative error $\max_i \frac{\|x_i(t)-x^*\|}{\|x^*\|}$ and average relative error $\frac{1}{n}\sum_{i=1}^{n}\frac{\|x_i(t)-x^*\|}{\|x^*\|}$.

VI. Communication Delays and Time Varying Consensus Protocols

The proof of Theorem 1 does not make any specific assumptions about the consensus matrix P other than that it is column stochastic and respects the structure of G. An immediate consequence is that PS-DDA converges in the presence of fixed edge delays. Indeed if we use the fixed delay model from [14] we end up using a column stochastic consensus matrix $Q(P)$ and the proof goes through unaltered. For the analysis we just need to initialize the weights of push-sum corresponding to the virtual delay nodes to zero.

Another important extension comes from allowing the consensus protocol to vary with time i.e. $P = P(t)$. This way we can model the completely asynchronous version of the algorithm. As a sender, node i controls column i of $P(t)$ and at each iteration independently chooses which of its neighbours to contact and also how to share its s_i and w_i values among the neighbours being contacted. In addition, we can also model random communication delays [14] again initializing the delay node weights to zero. With time varying protocols, convergence becomes a question of whether the forward product $\Phi[r,t] = P(r+1)^T P(r+2)^T \cdots P(t)^T, r < t$ is weakly ergodic [11]. In particular when P is selected i.i.d. at each time instant, weak ergodicity obtains if $\mathbb{E}[P]$ is strongly connected so that each node communicates with every other node eventually. Going back to the proof presented here, the only adaptation required is to replace the powers of P with the appropriate forward product i.e., $P^{t-r} \to \Phi[r,t]^T$ to get a general convergence result. For a result similar to Theorem 1 we need a bound for the convergence rate of the forward product such as those found in Chapter 4 of [12].

VII. Proof of Theorem 1

First we restate two lemmas from [1] that still hold in our setup.

Let $\{\hat g(t)\}_{t=1}^{\infty} \subset \mathbb{R}^d$ be an arbitrary sequence of vectors and $\{x(t)\}_{t=1}^{\infty}$ is generated as

$$x(t+1) = \Pi_{\mathcal{X}}^{\psi}\left(\sum_{r=1}^{t}\hat g(r), a(t)\right). \tag{21}$$

Lemma 1: For a non-increasing sequence $\{a(t)\}_{t=0}^{\infty}$ of positive step sizes and $\forall x^* \in \mathcal{X}$

$$\sum_{t=1}^{T}\langle \hat g(t), x(t)-x^*\rangle \le \frac{1}{2}\sum_{t=1}^{T}\hat a(t-1)\left\|\hat g(t)\right\|_*^2 + \frac{1}{a(T)}\psi(x^*). \tag{22}$$

Lemma 2: For any vectors $u, v \in \mathbb{R}^d$, we have $\left\|\Pi_{\mathcal{X}}^{\psi}(u,a) - \Pi_{\mathcal{X}}^{\psi}(v,a)\right\| \le a\left\|u-v\right\|_*$.

We are going to use $\pi^* = \min_s\{\pi_s\}$, the minimum entry in the stationary distribution of P and we know there exists a c such that t and all i, $\sum_{s=1}^{n}[P^t]_{is} \ge c$. Initially $z_i(0) = 0$ and $g_i(0) = 0$ for all i.

We start by computing an expression for the average dual variable $\overline{z}(t) = \frac{1}{n}\sum_{i=1}^{n} z_i(t)$. From (15) after back-substituting in the recursion we derive

$$\begin{aligned}
\overline{z}(t) =& \frac{1}{n}\sum_{i=1}^{n} z_i(t) && (23)\\
=& \frac{1}{n}\sum_{i=1}^{n}\sum_{r=1}^{t-1}\sum_{j=1}^{n}[P^{t-r}]_{ij}g_j(r-1) + g_i(t-1) && (24)\\
=& \sum_{r=1}^{t}\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}[P^{t-r}]_{ij}g_j(r-1) && (25)\\
=& \sum_{r=1}^{t-1}\frac{1}{n}\sum_{j=1}^{n}g_j(r)\sum_{i=1}^{n}[P^{t-r-1}]_{ij} && (26)\\
=& \sum_{r=1}^{t-1}\frac{1}{n}\sum_{j=1}^{n}g_j(r) && (27)
\end{aligned}$$

Where we used the facts that $P^0 = I$ and P^{t-r-1} is column stochastic. We also need the sequence $\{y(t)\}_{t=1}^{\infty}$ defined by the projection of $\overline{z}(t)$:

$$y(t) = \Pi_{\mathcal{X}}^{\psi}\left(\overline{z}(t), a(t-1)\right). \tag{28}$$

Next, we define the running averages $\hat{x}_i(T) = \frac{1}{T}\sum_{t=1}^{T} x_i(t)$ and $\hat{y}(T) = \frac{1}{T}\sum_{t=1}^{T} y(t)$. Using standard convexity arguments and Lemma 4 in [1], we can show that for any $x^* \in \mathcal{X}$ (for a detailed derivation see e.g. [1])

$$\begin{aligned} f(\hat{x}_j(T)) - f(x^*) \leq & \frac{1}{T}\sum_{t=1}^{T}\frac{1}{n}\sum_{i=1}^{n}\Big\langle g_i(t), x_i(t) - x^*\Big\rangle \quad (29) \\ & + \frac{1}{T}\sum_{t=1}^{T}\frac{L}{n}\sum_{i=1}^{n}\|y(t) - x_i(t)\| \quad (30) \\ & + \frac{L}{T}\sum_{t=1}^{T}\|x_j(t) - y(t)\|. \quad (31) \end{aligned}$$

We will bound the three RHS terms separately. For (29) we split the inner product by adding and subtracting $y(t)$:

$$\begin{aligned} \sum_{i=1}^{n}\langle g_i(t), x_i(t) - x^*\rangle = & \sum_{i=1}^{n}\langle g_i(t), y(t) - x^*\rangle \quad (32) \\ & + \sum_{i=1}^{n}\langle g_i(t), x_i(t) - y(t)\rangle \quad (33) \\ = & \left\langle \sum_{i=1}^{n} g_i(t), y(t) - x^*\right\rangle \quad (34) \\ & + \sum_{i=1}^{n}\langle g_i(t), x_i(t) - y(t)\rangle. \quad (35) \end{aligned}$$

To bound (34) we recall the definition (28) of $y(t)$ and the expression (27) to see that

$$y(t) = \Pi_{\mathcal{X}}^{\psi}\left(\sum_{r=1}^{t-1}\frac{1}{n}\sum_{i=1}^{n} g_i(r), a(t-1)\right). \tag{36}$$

Now, we see that

$$\begin{aligned} & \sum_{t=1}^{T}\frac{1}{n}\left\langle \sum_{i=1}^{n} g_i(t), y(t) - x^*\right\rangle \\ & = \sum_{t=1}^{T}\left\langle \frac{1}{n}\sum_{i=1}^{n} g_i(t), \Pi_{\mathcal{X}}^{\psi}\left(\sum_{r=1}^{t-1}\frac{1}{n}\sum_{i=1}^{n} g_i(r), a(t-1)\right) - x^*\right\rangle \end{aligned} \tag{37}$$

and we can invoke Lemma 1 with $\hat{g}(t) = \frac{1}{n}\sum_{i=1}^{n} g_i(t)$ to get

$$\begin{aligned} & \sum_{t=1}^{T}\frac{1}{n}\left\langle \sum_{i=1}^{n} g_i(t), y(t) - x^*\right\rangle \quad (38) \\ & \leq \frac{L^2}{2}\sum_{t=1}^{T} a(t-1) + \frac{1}{a(T)}\psi(x^*). \quad (39) \end{aligned}$$

For term (35), using the gradient magnitude bound, the definition (16) of $x_i(t)$ and Lemma 2 we get

$$\begin{aligned} \sum_{i=1}^{n}\langle g_i(t), x_i(t) - y(t)\rangle & \leq \sum_{i=1}^{n}\|g_i(t)\|_* \|x_i(t) - y(t)\| \quad (40) \\ & \leq \sum_{i=1}^{n} L\left\|\Pi_{\mathcal{X}}^{\psi}\left(\frac{z_i(t)}{w_i(t)}, a(t-1)\right) - \Pi_{\mathcal{X}}^{\psi}\left(\overline{z}(t), a(t-1)\right)\right\| \quad (41) \\ & \leq \sum_{i=1}^{n} La(t-1)\left\|\frac{z_i(t)}{w_i(t)} - \overline{z}(t)\right\|_*. \quad (42) \end{aligned}$$

Terms (30) and (31) are bounded similarly. Using the partial results so far we have shown that

$$\begin{aligned} f(\hat{x}_j(T)) - f(x^*) \leq & \frac{L^2}{2T}\sum_{t=1}^{T} a(t-1) + \frac{1}{Ta(T)}\psi(x^*) \\ & + \frac{2L}{nT}\sum_{t=1}^{T}\sum_{i=1}^{n} a(t-1)\left\|\overline{z}(t) - \frac{z_i(t)}{w_i(t)}\right\|_* \\ & + \frac{L}{T}\sum_{t=1}^{T} a(t-1)\left\|\overline{z}(t) - \frac{z_j(t)}{w_j(t)}\right\|_*. \end{aligned} \tag{43}$$

To complete the proof we thus need to bound each network error term $\left\|\overline{z}(t) - \frac{z_k(t)}{w_k(t)}\right\|_*$ for any node k. From (15), similarly to (27) we derive an expression for $z_k(t)$ as a function of past gradients and from (14) we see that $w_k(t) = \sum_{s=1}^{n}[P^t]_{ks}$. Now we proceed with the bound:

$$\begin{aligned} & \left\|\overline{z}(t) - \frac{z_k(t)}{w_k(t)}\right\|_* \quad (44) \\ & = \left\|\sum_{r=1}^{t-1}\frac{1}{n}\sum_{j=1}^{n} g_j(r) - \frac{\sum_{r=1}^{t-1}\sum_{j=1}^{n}[P^{t-r-1}]_{kj} g_j(r)}{\sum_{s=1}^{n}[P^t]_{ks}}\right\|_* \quad (45) \\ & \leq \sum_{r=1}^{t-1}\sum_{j=1}^{n}\|g_j(r)\|_*\left|\frac{1}{n} - \frac{[P^{t-r-1}]_{kj}}{\sum_{s=1}^{n}[P^t]_{ks}}\right| \quad (46) \\ & \leq L\sum_{r=1}^{t-1}\sum_{j=1}^{n}\left|\frac{[P^{t-r-1}]_{kj}}{\sum_{s=1}^{n}[P^t]_{ks}} - \frac{1}{n}\right|. \quad (47) \end{aligned}$$

We now show that the term in the absolute value remains bounded.

$$\begin{aligned} \left|\frac{[P^{t-r-1}]_{kj}}{\sum_{s=1}^{n}[P^t]_{ks}} - \frac{1}{n}\right| & = \left|\frac{n[P^{t-r-1}]_{kj} - \sum_{s=1}^{n}[P^t]_{ks}}{n\sum_{s=1}^{n}[P^t]_{ks}}\right| \quad (48) \\ & = \left|\frac{\sum_{s=1}^{n}[P^{t-r-1}]_{kj} - \sum_{s=1}^{n}[P^t]_{ks}}{n\sum_{s=1}^{n}[P^t]_{ks}}\right| \quad (49) \\ & = \left|\frac{\sum_{s=1}^{n}\left([P^{t-r-1}]_{kj} - \pi_k + \pi_k - [P^t]_{ks}\right)}{n\sum_{s=1}^{n}[P^t]_{ks}}\right| \quad (50) \\ & \leq \frac{\sum_{s=1}^{n}\left(\left|[P^{t-r-1}]_{kj} - \pi_k\right| + |\pi_k - [P^t]_{ks}|\right)}{n\sum_{s=1}^{n}[P^t]_{ks}} \quad (51) \end{aligned}$$

But the bound (13) gives also an exponential convergence rate for each individual element of P^t, so

$$\left|\frac{[P^{t-r-1}]_{kj}}{\sum_{s=1}^{n}[P^t]_{ks}} - \frac{1}{n}\right| \leq \frac{\sum_{s=1}^{n}\sqrt{\frac{\lambda_2^{t-r-1}}{\pi_j}} + \sum_{s=1}^{n}\sqrt{\frac{\lambda_2^{t}}{\pi_s}}}{n\sum_{s=1}^{n}[P^t]_{ks}} \tag{52}$$

$$\leq \frac{2n\frac{1}{\min_s\{\sqrt{\pi_s}\}}\sqrt{\lambda_2^{t-r-1}}}{n\sum_{s=1}^{n}[P^t]_{ks}} \tag{53}$$

$$\leq \frac{2\sqrt{\lambda_2^{t-r-1}}}{c\sqrt{\pi^*}} \tag{54}$$

where we used the fact that $\lambda_2 < 1$. We thus conclude that

$$\|\overline{z}(t) - z_k(t)\|_* \leq L\sum_{r=1}^{t-1}\sum_{j=1}^{n}\frac{2\sqrt{\lambda_2^{t-r-1}}}{c\sqrt{\pi^*}} \tag{55}$$

$$= \frac{2Ln}{c\sqrt{\pi^*}}\sum_{r=1}^{t-1}\sqrt{\lambda_2^{t-r-1}} \tag{56}$$

$$\leq \frac{2Ln}{c\sqrt{\pi^*}}\frac{1}{1-\sqrt{\lambda_2}} \tag{57}$$

where we used the formula for a finite geometric sum. Now we can go back to (43) to get

$$\begin{aligned} f(\hat{x}_j(T) - f(x^*) \leq & \frac{L^2}{2T}\sum_{t=1}^{T}a(t-1) + \frac{1}{Ta(T)}\psi(x^*) \\ & + \frac{2L}{T}\frac{2L}{c\sqrt{\pi^*}}\frac{1}{1-\sqrt{\lambda_2}}\sum_{t=1}^{T}a(t-1) \\ & + \frac{L}{T}\frac{2Ln}{c\sqrt{\pi^*}}\frac{1}{1-\sqrt{\lambda_2}}\sum_{t=1}^{T}a(t-1). \end{aligned} \tag{58}$$

And assuming that $\psi(x^*) \leq R^2$

$$\begin{aligned} f(\hat{x}_j(T)) - f(x^*) \leq & \frac{L^2}{2T}\sum_{t=1}^{T}a(t-1) + \frac{R^2}{Ta(T)} \\ & + \frac{1}{T}\frac{2L^2(2+n)}{c\sqrt{\pi^*}}\frac{1}{1-\sqrt{\lambda_2}}\sum_{t=1}^{T}a(t-1). \end{aligned} \tag{59}$$

Finally, if we choose $a(t) = \frac{A}{\sqrt{t}}$ and minimize for A, noticing that $\sum_{t=1}^{T}\frac{1}{\sqrt{t}} \leq 2\sqrt{t}$ we arrive at the result in Theorem 1.

VIII. Conclusions and Future Work

In this paper we describe and analyze PS-DDA, a new algorithm for consensus based distributed convex optimization. PS-DDA interleaves local gradient steps with one directional communication using the push-sum consensus protocol. Our proposed algorithm has several appealing properties. It converges to the optimal solution at a rate $O(T^{-0.5})$ which is the same as its predecessor [1] without requiring a doubly stochastic consensus protocol. It works with any column stochastic protocol P respecting the structure of the network G and converges to the unbiased optimum without needing to know the stationary distribution of P or the size of the network at every node. As a direct consequence, PS-DDA also converges in the presence of fixed edge delays. Furthermore, using time varying protocols we can implement asynchronous versions of PS-DDA and model random communication delays. Although we did not discuss those variants here in detail, our experiment on the real cluster exhibits the aforementioned desired properties in practice.

In the future we would like to investigate in more detail the performance of PS-DDA for solving real large scale machine learning problems. In particular, we would like to understand how the algorithm scales with the size of the network in practice. Moreover, we would like to investigate if the asynchronous version can yield speedups when compared to a synchronous algorithm where we have slow nodes that hold back the overall computation. Finally, we would like to extend the theoretical analysis to time varying consensus protocols and obtain bounds with tighter constants.

References

[1] J. Duchi, A. Agarwal, and M. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2011.

[2] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *FOCS, vol. 44. IEEE Computer Society Press, pp. 482–491*, 2003.

[3] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, January 2009.

[4] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM Journal on Control and Optimization*, vol. 20, no. 3, 2009.

[5] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847 – 1864, November 2010.

[6] B. Gharesifard and J. Cortes, "When does a digraph admit a doubly stochastic adjacency matrix?" in *Proceedings of the American Control Conference*, Baltimore, Maryland, 2010, pp. 2440–2445.

[7] S. S. Ram, A. Nedic, 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, 2011.

[8] K. I. Tsianos and M. G. Rabbat, "Distributed dual averaging for convex optimization under communication delays," in *American Control Conference (ACC)*, 2012.

[9] V. M. Preciado, A. Tahbaz-Salehi, and A. Jadbabaie, "On asymptotic consensus value in directed random networks," in *49th IEEE Conference on Decision and Control*, Atlanta, GA, USA, December 2010.

[10] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2748 – 2761, July 2009.

[11] F. Benezit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli, "Weighted gossip: Distributed averaging using non-doubly stochastic matrices," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2010, pp. 1753 – 1757.

[12] E. Seneta, *Non-negative Matrices and Markov Chains.* Springer, 1973.

[13] J. A. Fill, "Eigenvalue bounds on convergence to stationarity for non reversible markov chains, with an application to the exclusion process," *The Annals of Applied Probability*, vol. 1, no. 1, pp. 62–87, 1991.

[14] K. I. Tsianos and M. G. Rabbat, "Distributed consensus and optimization under communication delays," in *49th Allerton Conference on Communication, Control, and Computing*, 2011.

[15] W. Gropp, S. Huss-Lederman, A. Lumsdaine, E. Lusk, B. Nitzberg, W. Saphir, and M. Snir, *MPI—The Complete Reference - Volumes 1,2*. Cambridge, MA: MIT Press, 1998.

Appendix

For completeness here we show how to arrive at the bound (29) which is the starting point of our proof. Using L-Lipschitz continuity, the triangle inequality and convexity of $f(x)$ we have

$$f(\hat{x}_j(T)) - f(x^*) \tag{60}$$

$$= f(\hat{y}(T)) - f(x^*) + f(\hat{x}_j(T)) - f(\hat{y}(T)) \tag{61}$$

$$\leq f(\hat{y}(T)) - f(x^*) + L \left\| \hat{x}_j(T) - \hat{y}(T) \right\| \tag{62}$$

$$\leq \frac{1}{T} \sum_{t=1}^{T} f(y(t)) - f(x^*) + \frac{L}{T} \sum_{t=1}^{T} \left\| x_j(t) - y(t) \right\| . \tag{63}$$

Now we add and subtract $\sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} h_i(x_i(t))$ and use again convexity and L-Lipschitz continuity of the component functions $h_i(x)$ to get

$$\begin{aligned} f(\hat{x}_j(T)) - f(x^*) & \\ \leq & \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} \left[h_i(y(t)) - h_i(x_i(t)) \right] \\ & + \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} \left[h_i(x_i(t)) - h_i(x^*) \right] \\ & + \frac{L}{T} \sum_{t=1}^{T} \left\| x_j(t) - y(t) \right\| \qquad (64) \\ \leq & \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} L \left\| y(t) - x_i(t) \right\| \\ & + \frac{1}{T} \sum_{t=1}^{T} \frac{1}{n} \sum_{i=1}^{n} \langle g_i(t), x_i(t) - x^* \rangle \\ & + \frac{L}{T} \sum_{t=1}^{T} \left\| x_j(t) - y(t) \right\| \qquad (65) \end{aligned}$$

which is precisely the expression (29).