Randomized dual proximal gradient for large-scale distributed optimization

This is the post peer-review accepted manuscript of:

I. Notarnicola and G. Notarstefano, "Randomized dual proximal gradient for large-scale distributed optimization," 2015 54th IEEE Conference on Decision and Control (CDC), Osaka, 2015, pp. 712-717.

The published version is available online at:

https://doi.org/10.1109/CDC.2015.7402313

Randomized dual proximal gradient for large-scale distributed optimization

Ivano Notarnicola and Giuseppe Notarstefano

***Abstract*— In this paper we consider distributed optimization problems in which the cost function is separable (i.e., a sum of possibly non-smooth functions all sharing a common variable) and can be split into a strongly convex term and a convex one. The second term is typically used to encode constraints or to regularize the solution. We propose an asynchronous, distributed optimization algorithm over an undirected topology, based on a proximal gradient update on the dual problem. We show that by means of a proper choice of primal variables, the dual problem is separable and the dual variables can be stacked into separate blocks. This allows us to show that a distributed gossip update can be obtained by means of a randomized block-coordinate proximal gradient on the dual function.**

I. Introduction

Several estimation, learning, decision and control problems arising in cyber-physical networks involve the distributed solution of a constrained optimization problem. Typically, computing processors have only a partial knowledge of the problem (e.g. a portion of the cost function or a subset of constraints) and need to cooperate to compute a global solution of the whole problem. A key challenge to take into account when designing distributed optimization algorithms in peer-to-peer networks is that the communication is time-varying and possibly asynchronous, see, e.g., [1] for a review.

Early references on distributed optimization algorithms involved primal and dual subgradient methods and Alternating Direction Method of Multipliers (ADMM), designed for synchronous communication protocols over fixed graphs. More recently time-varying versions of these algorithmic ideas have been proposed to cope with more realistic peer-to-peer network scenarios. A Newton-Raphson consensus strategy is proposed in [2] to solve unconstrained, convex optimization problems under asynchronous, symmetric gossip communications. In [3] the authors propose accelerated distributed gradient methods for unconstrained problems over symmetric, time-varying networks connected on average. In order to deal with (time-varying) directed graphs, in [4] a push-sum algorithm for average consensus is combined with a primal subgradient method. Paper [5] extends these methods to online distributed optimization. In [6] a novel class of continuous-time, gradient-based distributed algorithms is proposed. A distributed (primal) proximal-gradient method is proposed in [7] for separable optimization problems which can handle only a common constraint. To solve constrained convex optimization problems, in [8] a distributed random projection algorithm is proposed for a balanced time-varying graph.

In [9] a novel asynchronous ADMM-based distributed method is proposed for separable, constrained convex optimization problem. In [10] the author proposes (primal) randomized block-coordinate descent methods for minimizing multi-agent convex optimization problems with linearly coupled constraints over networks. A combination of successive approximations and block-coordinate updates is proposed in [11] to solve separable, non-convex optimization problems in a big-data setting. Another class of algorithms exploits the exchange of active constraints among the nodes to solve general constrained convex programs [12]. The constraint exchange idea has been combined with dual decomposition and cutting-plane methods to solve robust convex optimization problems via polyhedral approximations [13]. These algorithms work under asynchronous, directed and unreliable communication.

It is worth noting that the algorithm in [10] uses a coordinate-descent idea similar to the one we use in this paper, but it works directly on the primal problem. Similarly, in [7] the proximal operator is used to handle the sparsity constraints directly on the primal problem, so that local constraints cannot be simultaneously taken into account. Indeed, in this paper we propose a dual approach to handle both. The optimization set-up in [9] is similar to the one considered in this paper. Differently from our approach, which is a dual method, a primal-dual algorithm is proposed in this reference. This difference results in different algorithm as well as different requirements on the cost functions.

The contribution of the paper is twofold. First, for a fixed graph topology, we develop a distributed optimization algorithm (based on a centralized dual proximal gradient idea introduced in Beck [14]) to minimize a separable strongly convex cost function. The proposed distributed algorithm is based on a proper choice of primal constraints (suitably separating the graph-induced and node-local constraints), that gives rise to a dual problem with a separable structure when expressed in terms of local conjugate functions. Thus, a proximal gradient applied to such a dual problem turns out to be a distributed algorithm where each node updates: (i) its primal variable through a local minimization and (ii) its dual variables through a suitable local proximal gradient step. The algorithm inherits the convergence properties of the centralized one and thus exhibits an $O(1/t)$ rate of convergence in objective value. We point out that the algorithm can be easily accelerated through a Nesterov's scheme, [15], thus

Ivano Notarnicola and Giuseppe Notarstefano are with the Department of Engineering, Università del Salento, Via Monteroni, 73100 Lecce, Italy, name.lastname@unisalento.it. This result is part of a project that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 638992 - OPT4SMART).

obtaining an $O(1/t^2)$ rate.

Second, we propose an asynchronous version of this algorithm for a symmetric gossip communication protocol. In this *event-triggered* communication set-up, a node is in idle mode until its local timer triggers. When in idle, it collects messages from neighboring nodes that are awake and may send information if required. When the local timer triggers, it updates its local (primal and dual) variables and broadcasts them to neighboring nodes. Under mild assumptions on the local triggering timers, the whole algorithm results into a random choice of one active node per iteration. Convergence is proven by showing that the distributed algorithm corresponds to a block-coordinate proximal gradient, as the one proposed in [16], performed on the dual problem. An important feature of the distributed algorithm is that each node can use its own local step-size, based on the Lipschitz constant of its and its neighbors' local cost functions. A key distinctive feature of the algorithm analysis is the combination of duality theory, coordinate-descent methods and properties of the proximal operator when applied to conjugate functions.

The paper is organized as follows. In Section II we set-up the optimization problem and the network model. In Section III we introduce and analyze a distributed dual proximal gradient algorithm for fixed communication graphs, while in Section IV we extend the algorithm to an asynchronous scenario. In Section V we show through a numerical example the convergence properties of the asynchronous algorithm.

Due to space constrains all proofs are omitted in this paper and will be provided in a forthcoming document.

Notation: Given a closed, nonempty convex set X, the indicator function of X is defined as $I_X(x) = 0$ if $x \in X$ and $I_X(x) = +\infty$ otherwise. Let $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$, its conjugate function $f^* : \mathbb{R}^d \to \mathbb{R}$ is defined as $f^*(y) := \sup_x \left\{y^T x - f(x)\right\}$. Let $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ be a closed proper convex function and α a positive scalar, the proximal operator $\mathbf{prox}_{\alpha f} : \mathbb{R}^d \to \mathbb{R}^d$ is defined by $\mathbf{prox}_{\alpha f}(v) := \operatorname{argmin}_x \left\{f(x) + \frac{1}{2\alpha}\|x - v\|^2\right\}$.

II. Problem set-up and network model

We consider the following optimization problem

$$\min_x \sum_{i=1}^n f_i(x) + g_i(x),$$

where $f_i : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ are proper, closed and strongly convex extended real-valued functions with strong convexity parameter $\sigma_i > 0$ and $g_i : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ are proper, closed and convex extended real-valued functions. The next assumption is standard and will guarantee that the dual problem is feasible and equivalent to the primal one (strong duality).

Assumption 2.1 (Constraint qualification): The intersection of the relative interior of $\mathbf{dom} \sum_{i=1}^n f_i$ and the relative interior of $\mathbf{dom} \sum_{i=1}^n g_i$ is non-empty. □

Notice that a convex constrained optimization problem

$$\min_x \sum_{i=1}^n f_i(x)$$
$$\text{subj. to } x \in \bigcap_{i=1}^n X_i \subseteq \mathbb{R}^d.$$

where X_i are convex set, can be modeled in our framework by setting $g_i(x) = I_{X_i}(x)$.

We want this optimization problem to be solved by a network of processors in a distributed way, i.e., by a set of peer processors communicating asynchronously and without the presence of a central coordinator.

Formally, we consider a network of nodes $\{1, \dots, n\}$ communicating according to an asynchronous broadcast protocol. Each node has its own concept of time defined by a local timer that randomly and independently of the other nodes triggers when to awake itself. Between two triggering events the node is in an *idle* mode, i.e., it can receive messages from neighboring nodes. When a trigger occurs, it switches into an *awake* mode in which it updates its local variables and transmits the updated information to its neighbors.

We assume that the asynchronous communication occurs among nodes that are neighbors in a given fixed, undirected and connected graph $\mathcal{G} = (\{1, \dots, n\}, \mathcal{E})$, where $\mathcal{E} \subseteq \{1, \dots, n\} \times \{1, \dots, n\}$ is the set of edges. That is, the edge (i, j) models the fact that node i can receive (respectively send) information from (to) node j when in idle (awake) mode. We denote by $\mathcal{N}_i$ the set of *neighbors* of node i in the fixed graph $\mathcal{G}$, i.e., $\mathcal{N}_i := \{j \in \{1, \dots, n\} \mid (i, j) \in \mathcal{E}\}$, and by $|\mathcal{N}_i|$ its cardinality.

We make the following assumption on the local timers.

Assumption 2.2 (Exponential i.i.d. local timers): The waiting times between consecutive triggering events, T_i, $i \in \{1, \dots, n\}$, are exponential i.i.d. random variables.□

Let $i_t \in \{1, \dots, n\}$, $t = 1, 2, \dots$ be the sequence identifying the generic t-th triggered node. Assumption 2.2 implies that i_t is an i.i.d. uniform process on the alphabet $\{1, \dots, n\}$. Each triggering will induce an iteration of the distributed optimization algorithm, so that t will be a universal, discrete time indicating the t-th iteration of the algorithm itself.

To exploit the sparsity of the underlying graph, we introduce copies of x and a coherence consensus constraint, so that the optimization problem can be equivalently written as

$$\begin{aligned} \min_{x_1, \dots, x_n} \ & \sum_{i=1}^n f_i(x_i) + g_i(x_i) \\ \text{subj. to } & x_i = x_j \quad \forall\, (i, j) \in \mathcal{E} \end{aligned} \tag{1}$$

with $x_i \in \mathbb{R}^d$ for all $i \in \{1, \dots, n\}$. The connectedness of $\mathcal{G}$ guarantees the equivalence.

III. Algorithm for fixed communication graph

In this section we derive and analyze a distributed dual proximal gradient algorithm for a fixed graph.

A. Dual problem derivation

We derive the dual version of the problem that will allow us to design our distributed dual proximal gradient algorithm. To obtain the desired separable structure of the dual problem, we set-up an equivalent formulation of problem (1) by adding a new set of slack variables z_i, $i \in \{1, \dots, n\}$, i.e.,

$$\begin{aligned} \min_{\substack{x_1,\dots,x_n \\ z_1,\dots,z_n}} \ & \sum_{i=1}^{n} f_i(x_i) + g_i(z_i) \\ \text{subj. to } & x_i = x_j \quad \forall\, (i,j) \in \mathcal{E} \\ & x_i = z_i \quad \forall\, i \in \{1, \dots, n\}. \end{aligned} \tag{2}$$

Let $\mathbf{x} = [x_1^T \ \dots \ x_n^T]^T$ and $\mathbf{z} = [z_1^T \ \dots \ z_n^T]^T$, the Lagrangian of the primal problem (2) is given by

$$\begin{aligned} L(\mathbf{x}, \mathbf{z}, \Lambda, \mu) &= \sum_{i=1}^{n} \Bigg(f_i(x_i) + g_i(z_i) \\ &\qquad + \sum_{j \in \mathcal{N}_i} \Big(\lambda_i^j\Big)^T (x_i - x_j) + \mu_i^T (x_i - z_i) \Bigg) \\ &= \sum_{i=1}^{n} \Bigg(f_i(x_i) + \sum_{j \in \mathcal{N}_i} \Big(\lambda_i^j\Big)^T (x_i - x_j) + \mu_i^T x_i \\ &\qquad + g_i(z_i) - \mu_i^T z_i \Bigg), \end{aligned}$$

where Λ and μ are respectively the vectors of the Lagrange multipliers λ_i^j, $(i,j) \in \mathcal{E}$, and μ_i, $i \in \{1, \dots, n\}$, and in the last line we have separated the terms in $\mathbf{x}$ and $\mathbf{z}$. Since $\mathcal{G}$ is undirected, the Lagrangian can be equivalently rewritten as

$$\begin{aligned} L(\mathbf{x}, \mathbf{z}, \Lambda, \mu) = \sum_{i=1}^{n} \Bigg(& f_i(x_i) + x_i^T \Big(\sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) + \mu_i \Big) \\ & + g_i(z_i) - z_i^T \mu_i \Bigg) \end{aligned}$$

where λ_i^j, $j \in \mathcal{N}_i$ and μ_i are variables handled by node i (consistently λ_j^i is handled by node j neighbor of node i).

The dual function is

$$\begin{aligned} q(\Lambda, \mu) &:= \min_{\mathbf{x}, \mathbf{z}} L(\mathbf{x}, \mathbf{z}, \Lambda, \mu) \\ &= \min_{\mathbf{x}} \sum_{i=1}^{n} \Bigg(f_i(x_i) + x_i^T \Big(\sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) + \mu_i \Big) \Bigg) \\ &\qquad + \min_{\mathbf{z}} \sum_{i=1}^{n} \Big(g_i(z_i) - z_i^T \mu_i \Big) \\ &= \sum_{i=1}^{n} \min_{x_i} \Bigg(f_i(x_i) + x_i^T \Big(\sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) + \mu_i \Big) \Bigg) \\ &\qquad + \sum_{i=1}^{n} \min_{z_i} \Big(g_i(z_i) - z_i^T \mu_i \Big) \end{aligned}$$

where we have used the separability of the Lagrangian with respect to each x_i and each z_i. Then, by using the definition of conjugate function, the dual function can be rewritten as

$$q(\Lambda, \mu) = \sum_{i=1}^{n} \Bigg(- f_i^* \Big(- \sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) - \mu_i \Big) - g_i^*(\mu_i) \Bigg).$$

The dual problem of (2) consists of maximizing the dual function with respect to dual variables Λ and μ, i.e.,

$$\max_{\Lambda, \mu} \sum_{i=1}^{n} \Bigg(- f_i^* \Big(- \sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) - \mu_i \Big) - g_i^*(\mu_i) \Bigg). \tag{3}$$

By Assumption 2.1 the dual problem (3) is feasible and strong duality holds, so that (3) can be solved to get a solution of (2).

B. Distributed Dual Proximal Gradient Algorithm

To develop the algorithm, we start rewriting problem (3) by using a more compact notation, and in the equivalent minimization version. We stack the dual variables as $y = [y_1^T \ \dots \ y_n^T]^T$, where

$$y_i = \begin{bmatrix} \Lambda_i \\ \mu_i \end{bmatrix} \in \mathbb{R}^{d|\mathcal{N}_i|+d} \tag{4}$$

with $\Lambda_i \in \mathbb{R}^{d|\mathcal{N}_i|}$ a vector whose block-component associated to neighbor j is $\lambda_i^j \in \mathbb{R}^d$. Thus, the dual problem can be written as

$$\min_{y} \Gamma(y) = F^*(y) + G^*(y), \tag{5}$$

where

$$\begin{aligned} F^*(y) &:= \sum_{i=1}^{n} f_i^* \Big(- \sum_{j \in \mathcal{N}_i} (\lambda_i^j - \lambda_j^i) - \mu_i \Big) \\ G^*(y) &:= \sum_{i=1}^{n} g_i^* \big(\mu_i\big). \end{aligned}$$

The proposed distributed algorithm will be based on a proximal gradient applied to the above formulation of the dual problem. Next, we describe the local update of each node $i \in \{1, \dots, n\}$ and then, in the next subsection, we show its convergence properties.

Node i updates its local dual variables λ_i^j, $j \in \mathcal{N}_i$, and μ_i according to a local proximal gradient step, and its primal variable $x_i^\star$ through a local minimization. The step-size of the proximal gradient step is denoted by α. Then, the updated primal and dual values are exchanged with the neighboring nodes according to a synchronous communication over a fixed undirected graph. The local dual variables at node i are initialized as λ_{i0}^j, $j \in \mathcal{N}_i$, and μ_{i0}. A pseudo-code of the local update at each node of the distributed algorithm is given in Algorithm 1.

Remark 3.1: In order to start the algorithm, a preliminary communication step is needed in which each node i sends to each neighbor j its λ_{i0}^j. This step can be avoided if the nodes agree to set $\lambda_{i0}^j = 0$. □

C. Algorithm analysis

Lemma 3.2 ([17], [18]): Let φ be a closed, strictly convex function and φ^* its conjugate function. Then

$$\nabla \varphi^*(y) = \operatorname*{argmax}_x \big\{ y^T x - \varphi(x) \big\} = \operatorname*{argmin}_x \big\{ \varphi(x) - y^T x \big\}.$$

Moreover, if φ is strongly convex with convexity parameter σ, then $\nabla \varphi^*$ is Lipschitz continuous with Lipschitz constant given by $\frac{1}{\sigma}$. □

Algorithm 1 Distributed Dual Proximal Gradient

Processor states: $x_i^\star$, λ_i^j for all $j \in \mathcal{N}_i$ and μ_i

Initialization: $\lambda_i^j(0) = \lambda_{i0}^j$ for all $j \in \mathcal{N}_i$, $\mu_i(0) = \mu_{i0}$

$x_i^\star(0) = \operatorname{argmin}_{x_i} \Big\{ x_i^T \Big(\sum_{j \in \mathcal{N}_i} \big(\lambda_{i0}^j - \lambda_{j0}^i\big) + \mu_{i0} \Big) + f_i(x_i) \Big\}$

Evolution:

FOR: $t = 1, 2, \dots$

receive $x_j^\star(t-1)$ for each $j \in \mathcal{N}_i$

update

$$\lambda_i^j(t) = \lambda_i^j(t-1) + \alpha \big[x_i^\star(t-1) - x_j^\star(t-1) \big]$$

update

$$\tilde{\mu}_i = \mu_i(t-1) + \alpha\, x_i^\star(t-1)$$

$$\mu_i(t) = \mathbf{prox}_{\alpha g_i^*}(\tilde{\mu}_i) = \tilde{\mu}_i - \alpha\, \mathbf{prox}_{\frac{1}{\alpha} g_i}\left(\frac{\tilde{\mu}_i}{\alpha} \right)$$

receive $\lambda_j^i(t)$ for each $j \in \mathcal{N}_i$

update

$$x_i^\star(t) = \operatorname*{argmin}_{x_i} \left\{ x_i^T \Big(\sum_{j \in \mathcal{N}_i} \Big(\lambda_i^j(t) - \lambda_j^i(t) \Big) + \mu_i(t) \Big) + f_i(x_i) \right\}$$

Lemma 3.3 (Moreau decomposition, [19]): Let $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ be a closed, strictly convex function and f^* its conjugate, then $\forall x \in \mathbb{R}^d$, $x = \mathbf{prox}_f(x) + \mathbf{prox}_{f^*}(x)$. □

Lemma 3.4 (Extended Moreau decomposition): Let $\varphi : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ be a closed, strictly convex function and φ^* its conjugate. Then for any $x \in \mathbb{R}^d$ and $\alpha > 0$, it holds $x = \mathbf{prox}_{\alpha\varphi}\{x\} + \alpha \mathbf{prox}_{\frac{1}{\alpha}\varphi^*}\{\frac{x}{\alpha}\}$.

Lemma 3.5: Let $y = [y_1^T \ \dots \ y_n^T]^T \in \mathbb{R}^{n(D+d)}$ where $y_i = [\Lambda_i^T \ \mu_i^T]^T$ with $\Lambda_i \in \mathbb{R}^D$ and $\mu_i \in \mathbb{R}^d$, $i \in \{1, \dots, n\}$. Let $G^*(y) = \sum_{i=1}^n g_i^*(\mu_i)$, then the proximal operator of αG^* evaluated at y is given by

$$\mathbf{prox}_{\alpha G^*}(y) = \begin{bmatrix} \Lambda_1 \\ \mathbf{prox}_{\alpha g_1^*}(\mu_1) \\ \vdots \\ \Lambda_n \\ \mathbf{prox}_{\alpha g_n^*}(\mu_n) \end{bmatrix}.$$

□

Theorem 3.6: For each $i \in \{1, \dots, n\}$, let f_i be a proper, closed, strongly convex extended real-valued function with strong convexity parameter $\sigma_i > 0$, and let g_i be a proper, convex extended real-valued function. Let $y^\star$ be the minimizer of (5). Suppose that in Algorithm 1 the step-size α is chosen such that $0 < \alpha \leq \frac{1}{\sum_{i=1}^n \frac{1}{\sigma_i}}$. Then the sequence $y(t) = [y_1(t)^T \dots y_n(t)^T]^T$ generated by the Distributed Dual Proximal Gradient (Algorithm 1) converges to $y^\star$ and in objective value satisfies

$$\Gamma(y(t)) - \Gamma(y^\star) \leq \frac{\left(\sum_{i=1}^n \frac{1}{\sigma_i} \right) \|y_0 - y^\star\|^2}{2t},$$

where $y_0 = [y_1(0)^T \ \dots \ y_n(0)^T]^T$ is the initial condition.

Remark 3.7 (Nesterov's acceleration): We can include a Nesterov's *extrapolation step* in the algorithm, which accelerates the algorithm ([15] for further details), attaining a faster $O(1/t^2)$ convergence rate in objective value. □

IV. Asynchronous distributed dual proximal gradient

In this section we present an asynchronous distributed dual proximal gradient and prove its convergence in probability.

We start by describing the local evolution at each node $i \in \{1, \dots, n\}$. First, recall from the network model introduced in Section II that a node can be into two different modes: when in *idle* it continuously listens to incoming messages from its neighbors (and, if needed, may send them auxiliary information back), while when in *awake* it updates its local variables and transmits them to its neighbors. The transition between modes is asynchronously ruled via local timers, $\tau_i \in \mathbb{R}$, $i \in \{1, \dots, n\}$ (they are assumed to have infinite precision). As from Assumption 2.2, timers trigger according to n exponential i.i.d. random variables T_i, $i \in \{1, \dots, n\}$. In the algorithm we make a slight abuse of notation denoting by T_i the realization of the random variables T_i.

Each node i updates its local dual variables λ_i^j, $j \in \mathcal{N}_i$ and μ_i by a local proximal gradient step, and its primal variable $x_i^\star$ through a local minimization. Each node uses a properly chosen, *local* step-size α_i for the proximal gradient step.

Algorithm 2 Asynchronous Distributed Dual Proximal Gradient

Processor states: $x_i^\star$, λ_i^j for all $j \in \mathcal{N}_i$ and μ_i

Initialization: $\lambda_i^j = \lambda_{i0}^j$ for all $j \in \mathcal{N}_i$, $\mu_i = \mu_{i0}$ and

$x_i^\star(0) = \operatorname{argmin}_{x_i} \Big\{ x_i^T \Big(\sum_{j \in \mathcal{N}_i} \big(\lambda_{i0}^j - \lambda_{j0}^i\big) + \mu_{i0} \Big) + f_i(x_i) \Big\}$

set $\tau_i = 0$ and get T_i

Evolution:

IDLE:

WHILE $\tau_i \leq T_i$

receive $x_j^\star$ and/or λ_j^i from each $j \in \mathcal{N}_i$

IF λ_j^i is received THEN compute and broadcast

$$x_i^\star = \operatorname*{argmin}_{x_i} \left\{ x_i^T \Big(\sum_{\ell \in \mathcal{N}_i} \Big(\lambda_i^\ell - \lambda_\ell^i \Big) + \mu_i \Big) + f_i(x_i) \right\}$$

AWAKE:

update and broadcast

$$\lambda_i^{j+} = \lambda_i^j + \alpha_i \big(x_i^\star - x_j^\star \big), \ \forall\, j \in \mathcal{N}_i$$

update

$$\tilde{\mu}_i = \mu_i + \alpha_i\, x_i^\star$$

$$\mu_i^+ = \mathbf{prox}_{\alpha_i g_i^*}(\tilde{\mu}_i) = \tilde{\mu}_i - \alpha_i\, \mathbf{prox}_{\frac{1}{\alpha_i} g_i}\Big(\frac{\tilde{\mu}_i}{\alpha_i} \Big)$$

compute and broadcast

$$x_i^\star = \operatorname*{argmin}_{x_i} \left\{ x_i^T \Big(\sum_{j \in \mathcal{N}_i} \Big(\lambda_i^{j+} - \lambda_j^i \Big) + \mu_i^+ \Big) + f_i(x_i) \right\}$$

set $\tau_i = 0$, get a new T_i and go to ***IDLE***.

Remark 4.1: In order to set the step-size α_i, node i needs a preliminary communication step to receive the convexity parameters from its neighbors. □

From an external, global perspective, the described local asynchronous updates result into an algorithmic evolution, in which at each iteration only one node wakes up randomly, uniformly and independently from previous iterations. This follows from the memoryless property of the exponential distribution. Thus, in this high-level view, we can consider a *universal* (discrete) time-variable t, which counts the iterations of the whole algorithm evolution. This variable will be used in the statement of Theorem 4.2.

Theorem 4.2: For each $i \in \{1, \dots, n\}$, let f_i be a proper, closed and strongly convex extended real-valued function with strong convexity parameter $\sigma_i > 0$, and let g_i be a proper convex extended real-valued function. Let $y^\star$ be the minimizer of (5). Suppose that in Algorithm 2 each local step-size α_i is chosen such that $0 < \alpha_i \leq \frac{1}{L_i}$ with

$$L_i = \sqrt{\frac{1}{\sigma_i^2} + \sum_{j \in \mathcal{N}_i} \left(\frac{1}{\sigma_i} + \frac{1}{\sigma_j}\right)^2}.$$

Then the sequence $y(t) = [y_1(t)^T \dots y_n(t)^T]^T$ generated by the Asynchronous Distributed Dual Proximal Gradient (Algorithm 2) converges in probability to $y^\star$, i.e., for any $\varepsilon \in \big(0, \Gamma(y_0)\big)$, where $y_0 = [y_1(0)^T \dots y_n(0)^T]^T$ is the initial condition, and target confidence $0 < \rho < 1$, there exists $\bar{t}(\varepsilon, \rho) > 0$ such that for all $t \geq \bar{t}$ it holds

$$\mathrm{P}\Big(\Gamma(y(t)) - \Gamma(y^\star) \leq \varepsilon\Big) \geq 1 - \rho.$$

V. Simulations

In this section we provide a numerical example showing the effectiveness of the proposed Asynchronous Distributed Dual Proximal Gradient.

We consider an undirected connected Erdős-Rényi graph $\mathcal{G}$ with parameter 0.2, connecting $n = 15$ nodes. We assume each decision variable $x_i \in \mathbb{R}^2$, $i \in \{1, \dots, n\}$. Let each local objective function f_i be quadratic and randomly generated as

$$f_i(x_i) = x_i^T Q_i x_i + r_i^T x_i$$

where $Q_i \in \mathbb{R}^{2\times 2}$ is diagonal with diagonal elements uniformly distributed in $[1, 2]$ and $r_i \in \mathbb{R}^2$ has elements uniformly randomly distributed in $[-5, 5]$. We let each g_i be the indicator function of a convex polytope $X_i = \{x_i \in \mathbb{R}^2 \mid a_i^T x_i \leq b_i\}$, with components of a_i generated uniformly in $[0, 10]$ and components of b_i in $[-5, 5]$. We initialize to zero the dual variables λ_i^j, $j \in \mathcal{N}_i$, and μ_i for all $i \in \{1, \dots, n\}$, and use a constant step-size $\alpha_i = 1$ for all nodes.

Figure 1 shows the convergence of the primal (and dual) cost to the optimal centralized value. We recall that the primal cost is $-\Gamma(y(t))$, with $\Gamma(y(t))$ being the dual cost in the minimization version (5). In Figure 2 we plot the behavior of the first component of primal variables $x_i^\star(t)$. The horizontal dotted-line is the optimal primal solution. In the inset the first iterations for five selected nodes, $x_i^\star$, $i = 1, 5, 6, 7, 13$, are highlighted, in order to better show the transient, piece-wise constant behavior due to the gossip update.

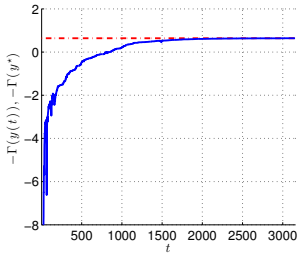

Fig. 1. Cost $-\Gamma(y(t))$ (solid blue) vs optimal cost $-\Gamma(y^\star)$ (dotted red).

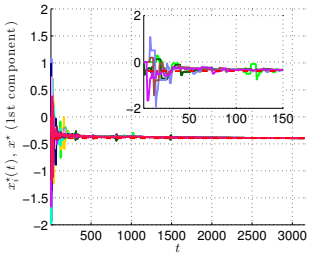

Fig. 2. First component of $x_i^\star(t)$, $i \in \{1, \dots, n\}$, zoom on selected i.

Then we show the evolution of the dual variables. First, note that μ_i is associated to the local constraint X_i of x_i. We obtain that only μ_{13}, the multiplier relative to the only active constraint, converges to a nonzero value, whereas all the other μ_is, associated to the inactive constraints, converge to 0. In Figure 3 the first component of μ_{13} is plotted.

Finally, we plot the evolution of λ_i^j, $j \in \mathcal{N}_i$, for node $i = 5$ (with $\mathcal{N}_i = \{3, 6, 10, 12, 14\}$), see Figure 4 for the first component. As expected the multipliers converge to nonzero values representing the "price" needed to enforce equality constraints on the primal variables x_i and x_j.

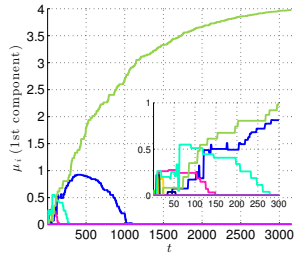

Fig. 3. First component of $\mu_i(t)$, $i \in \{1, \dots, n\}$.

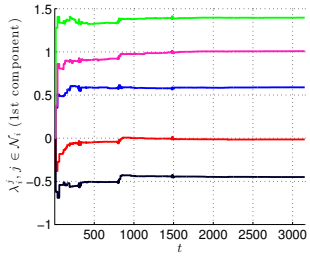

Fig. 4. First component of λ_i^j, $j \in N_i$, with $i = 5$.

VI. Conclusions

In this paper we have proposed an asynchronous, distributed optimization algorithm, based on a block-coordinate dual proximal gradient method to solve separable, constrained optimization problems. The main idea is to construct a suitable, separable dual problem via a proper choice of primal constraints. Then, the dual problem is solved through a proximal gradient algorithm. Thanks to the separable structure of the dual problem in terms of local conjugate functions, the proximal gradient update results into a distributed algorithm, where each node performs a local minimization on its primal variable, and a local proximal gradient update on its dual variables. An asynchronous version of the distributed algorithm is obtained by exploiting a randomized, block-coordinate descent approach.

APPENDIX

A. Randomized coordinate descent for composite functions

Consider the following optimization problem

$$\min_{y\in\mathbb{R}^N} \Gamma(y) := \Phi(y) + \Psi(y) \tag{A.6}$$

where $\Phi : \mathbb{R}^N \to \mathbb{R}$ and $\Psi : \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$ are convex functions.

We decompose the decision variable as $y = [y_1^T \ \dots \ y_n^T]^T$ and, consistently, we decompose the space $\mathbb{R}^N$ into n subspaces as follows. Let $U \in \mathbb{R}^{N\times N}$ be a column permutation of the $N \times N$ identity matrix and, further, let $U = [U_1 \ U_2 \ \dots \ U_n]$ be a decomposition of U into n submatrices, with $U_i \in \mathbb{R}^{N\times N_i}$ and $\sum_i N_i = N$. Thus, any vector $y \in \mathbb{R}^N$ can be uniquely written as $y = \sum_i U_i y_i$ and, viceversa, $y_i = U_i^T y$.

We let problem (A.6) satisfy the following assumptions.

*Assumption A.1 (**Smoothness of** Φ):* The gradient of Φ is block coordinate-wise Lipschitz continuous with positive constants $L_1, \dots, L_n$. That is, for all $y \in \mathbb{R}^N$ and $s_i \in \mathbb{R}^{N_i}$ it holds

$$\|\nabla_i \Phi(y + U_i s_i) - \nabla_i \Phi(y)\| \le L_i \|s_i\|,$$

where $\nabla_i \Phi(y)$ is the i-th block component of $\nabla\Phi(y)$. □

*Assumption A.2 (**Separability of** Ψ):* The function Ψ is block-separable, i.e., it can be decomposed as $\Psi(y) = \sum_{i=1}^n \psi_i(y_i)$, with each $\psi_i : \mathbb{R}^{N_i} \to \mathbb{R}\cup\{+\infty\}$ a proper, closed convex extended real-valued function. □

*Assumption A.3 (**Feasibility**):* The set of minimizers of problem (A.6) is non-empty. □

Algorithm 3 UCDC

Initialization: $y(0) = y_0$

for $t = 0, 1, 2, \dots$ **do**

choose $i_t \in \{1, \dots, n\}$ with probability $\frac{1}{n}$

compute

$$T^{(i_t)}\big(y(t)\big) = \operatorname*{argmin}_{w_{i_t}\in\mathbb{R}^{N_{i_t}}} \Big\{ V_{i_t}(y(t), w_{i_t}) \Big\}$$

where

$$V_{i_t}(y, s_{i_t}) := \nabla_{i_t}\Phi(y)^T s_{i_t} + \frac{L_{i_t}}{2}\|s_{i_t}\|^2 + \psi_{i_t}(y_{i_t} + s_{i_t})$$

update $y(t+1) = y(t) + U_{i_t} T^{(i_t)}\big(y(t)\big)$

The convergence result for UCDC (Algorithm 3) is given in [16, Theorem 5], here reported for completeness.

Theorem A.4 (Theorem 5, [16]): Let Assumptions A.1, A.2 and A.3 hold. Then, for any $\varepsilon \in \Big(0, \Gamma(y_0) - \Gamma(y^\star)\Big)$, there exists $\bar{t}(\varepsilon, \rho) > 0$ such that if $y(t)$ is the random sequence generated by UCDC applied to problem (A.6), then for all $t \ge \bar{t}$ it holds that

$$\mathrm{P}\Big(\Gamma(y(t)) - \Gamma(y^\star) \le \varepsilon\Big) \ge 1 - \rho,$$

where $y^\star$ is a minimizer of problem (A.6), $y_0 \in \mathbb{R}^N$ is the initial condition and $\rho \in (0, 1)$ is the target confidence. □

REFERENCES

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

[2] F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, and L. Schenato, "Asynchronous Newton-Raphson consensus for distributed convex optimization," in *3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, 2012.

[3] D. Jakovetic, J. M. Freitas Xavier, and J. M. Moura, "Convergence rates of distributed Nesterov-like gradient methods on random networks," *IEEE Transactions on Signal Processing*, vol. 62, no. 4, pp. 868–882, 2014.

[4] A. Nedic and A. Olshevsky, "Distributed optimization over time-varying directed graphs," in *IEEE 52nd Annual Conference on Decision and Control (CDC)*. IEEE, 2013, pp. 6855–6860.

[5] M. Akbari, B. Gharesifard, and T. Linder, "Distributed online convex optimization on time-varying directed graphs," in *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*. IEEE, 2014.

[6] S. S. Kia, J. Cortes, and S. Martinez, "Distributed convex optimization via continuous-time coordination algorithms with discrete-time communication," in *arXiv*, 2014.

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

[8] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 221–229, 2013.

[9] E. Wei and A. Ozdaglar, "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.

[10] I. Necoara, "Random coordinate descent algorithms for multi-agent convex optimization over networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 8, pp. 2001–2012, 2013.

[11] F. Facchinei, G. Scutari, and S. Sagratella, "Parallel selective algorithms for nonconvex big data optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 7, pp. 1874–1889, 2015.

[12] G. Notarstefano and F. Bullo, "Distributed abstract optimization via constraints consensus: Theory and applications," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2247–2261, 2011.

[13] M. Bürger, G. Notarstefano, and F. Allgöwer, "A polyhedral approximation framework for convex and robust distributed optimization," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 384–395, 2014.

[14] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[15] Y. Nesterov *et al.*, "Gradient methods for minimizing composite objective function," 2007.

[16] P. Richtárik and M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Mathematical Programming*, vol. 144, no. 1-2, pp. 1–38, 2014.

[17] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[18] A. Beck and M. Teboulle, "A fast dual proximal gradient algorithm for convex minimization and applications," *Operations Research Letters*, vol. 42, no. 1, pp. 1–6, 2014.

[19] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 123–231, 2013.