

New primal-dual proximal algorithm for distributed optimization

Questa è la versione sottoposta a revisione paritaria (postprint) della seguente opera:

Original

New primal-dual proximal algorithm for distributed optimization / Latafat, P.; Stella, L.; Patrinos, P.. - (2016), pp. 1959-1964. (55th IEEE Conference on Decision and Control, CDC 2016 Las Vegas, USA 12-14 December 2016) [10.1109/CDC.2016.7798551].

Availability:

This version is available at: 20.500.11771/32227

Publisher:

IEEE

Published

DOI:10.1109/CDC.2016.7798551

Terms of use:

This publication is made accessible in accordance with the terms for deposit in the institutional repository, as defined by the IMT School for Advanced Studies Lucca's Open Access Policy. (https://library.imtlucca.it/sites/default/files/regolamento-policy-open-access-imtlib_0.pdf).

Si prega di consultare le pagine informative dell'editore relative alle politiche di autoarchiviazione.

(Article begins on next page)

New Primal-Dual Proximal Algorithm for Distributed Optimization

Puya Latafat, Lorenzo Stella, Panagiotis Patrinos

Abstract—We consider a network of agents, each with its own private cost consisting of the sum of two possibly nonsmooth convex functions, one of which is composed with a linear operator. At every iteration each agent performs local calculations and can only communicate with its neighbors. The goal is to minimize the aggregate of the private cost functions and reach a consensus over a graph. We propose a primal-dual algorithm based on *Asymmetric Forward-Backward-Adjoint* (AFBA), a new operator splitting technique introduced recently by two of the authors. Our algorithm includes the method of Chambolle and Pock as a special case and has linear convergence rate when the cost functions are piecewise linear-quadratic. We show that our distributed algorithm is easy to implement without the need to perform matrix inversions or inner loops. We demonstrate through computational experiments how selecting the parameter of our algorithm can lead to larger step sizes and yield better performance.

I. INTRODUCTION

In this paper we deal with the distributed solution of the following optimization problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^N f_i(x) + g_i(C_i x) \quad (1)$$

where for $i = 1, \dots, N$, C_i is a linear operator, f_i and g_i are proper closed convex and possibly nonsmooth functions. We further assume that the *proximal mappings* associated with f_i and g_i are efficiently computable [1]. In a more general case we can include another continuously differentiable term with Lipschitz-continuous gradient in (1) and use [2, Algorithm 3] that includes the algorithm of Vũ and Condat [3], [4] as special case. We do not pursue this here for clarity of exposition.

Problems of this form appear in several application fields. In a distributed model predictive control setting, f_i can represent individual finite-horizon costs for each agent, C_i model the linear dynamics of each agent and possibly coupling constraints that are split through the introduction of extra variables, and g_i model state and input constraints.

In machine learning and statistics the C_i are feature matrices and functions g_i measure the *fitting* of a predicted model with the observed data, while the f_i is *regularization* terms that enforces some prior knowledge in the solution (such as sparsity, or belonging to a certain constraint set).

P. Latafat is with IMT School for Advanced Studies Lucca, Piazza San Francesco 19, 55100 Lucca, Italy; Email: puya.latafat@imtlucca.it

L. Stella and P. Patrinos are with the Department of Electrical Engineering (ESAT-STADIUS) and Optimization in Engineering Center (OPTEC), KU Leuven, Kasteelpark Arenberg 10, 3001 Leuven-Heverlee, Belgium; Emails: lorenzo.stella@esat.kuleuven.be, panos.patrinos@esat.kuleuven.be

For example if g_i is the so-called hinge loss and $f_i = \frac{\lambda}{2} \|\cdot\|_2^2$, for some $\lambda > 0$, then one recovers the standard SVM model. If instead $f_i = \lambda \|\cdot\|_1$ then one recovers the ℓ_1 -norm SVM problem [5].

Clearly problem (1) can be solved in a centralized fashion, when all the data of the problem (functions f_i , g_i and matrices C_i , for all $i \in \{1, \dots, N\}$) are available at one computing node. When this is the case one might formulate and solve the aggregated problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} f(x) + g(Cx),$$

for which algorithms are available [2], [6], [7]. However, such a centralized approach is not realistic in many scenarios. For example, suppose that $g_i(C_i x)$ models least-squares terms and C_1, \dots, C_N are very large features matrices. Then collecting C_1, \dots, C_N into a single computer may be infeasible due to communication costs, or even worse they may not fit into the computer's memory. Furthermore, the exchange of such information may not be possible at all due to privacy issues.

Our goal is therefore to solve problem (1) in a distributed fashion. Specifically, we consider a connected network of N computing agents, where the i -th agent is able to compute proximal mappings of f_i , g_i , and matrix vector products with C_i (and its adjoint operator). We want all the agents to iteratively converge to a *consensus* solution to (1), and to do so by only exchanging variables among neighbouring nodes, i.e. no centralized computations (i.e., existence of a fusion center) are needed during the iterations.

To do so, we will propose a solution based on the recently introduced *Asymmetric Forward-Backward-Adjoint* (AFBA) splitting method [2]. This new splitting technique solves monotone inclusion problems involving three operators, however, in this work we will focus on a special case that involves two terms. Specifically, we develop a distributed algorithm which is based on a special case of AFBA applied to the monotone inclusion corresponding to the primal-dual optimality conditions of a suitable *graph splitting* of (1). Our algorithm involves a nonnegative parameter θ which serves as a tuning knob that allows to recover different algorithms. In particular, the algorithm of [6] is recovered in the special case when $\theta = 2$. We demonstrate how tuning this parameter affects the stepsizes and ultimately the convergence rate of the algorithm.

Other algorithms have been proposed for solving problems similar to (1) in a distributed way. As a reference framework,

all algorithms aim at solving in a distributed way the problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^N F_i(x).$$

In [8] a distributed subgradient method is proposed, and in [9] this idea is extended to the projected subgradient method. More recently, several works focused on the use of ADMM for distributed optimization. In [10] the generic ADMM for consensus-type problems is illustrated. A drawback of this approach is that at every iteration the agents must solve a complicated subproblem that might require an inner iterative procedure. In [11] another formulation is given for the case where $F_i = f_i + g_i$, and only proximal mappings with respect to f_i and g_i are separately computed in each node. Still, when either f_i or g_i is not separable (such as when they are composed with linear operators) these are not trivial to compute and may require inner iterative procedures, or factorization of the data matrices involved. Moreover, in both [10], [11] a central node is required for accumulating each agents variables at every iteration, therefore these formulations lead to *parallel* algorithms rather than distributed. In [12] the optimal parameter selection for ADMM is discussed in the case of distributed quadratic programming problems. In [13]–[15], fully distributed algorithms based on ADMM proposed, assuming that the proximal mapping of F_i is computable, which is impractical in many cases. In [16] the authors propose a variation of the Vũ-Condat algorithm [3], [4], having ADMM as a special case, and show its application to distributed optimization where $F_i = f_i + g_i$, but no composition with a linear operator is involved. Only proximal operations with respect to f_i and g_i and local exchange of variables (*i.e.*, among neighboring nodes) is required, and the method is analyzed in an asynchronous setting.

In this paper we deal with the more general scenario of problem (1). The main features of our approach, that distinguish it from the related works mentioned above, are:

- (i) We deal with F_i that is the sum of two possibly nonsmooth functions one of which is composed with a linear operator.
- (ii) Our algorithm only require local exchange of information, *i.e.*, only neighboring nodes need to exchange local variables for the algorithms to proceed.
- (iii) The iterations involve *direct* operations on the objective terms. Only evaluations of prox_{f_i} , $\text{prox}_{g_i^*}$ and matrix-vector products with C_i and C_i^T are involved. In particular, no inner subproblem needs to be solved iteratively by the computing agents, and no matrix inversions are required.

The paper is organized as follows. [Section II](#) is devoted to a formulation of problem (1) which is amenable to be solved in a distributed fashion by the proposed methods. In [Section III](#) we detail how the primal-dual algorithm in [2, Algorithm 6] together with an intelligent change of variables gives rise to distributed iterations. We then discuss implementation considerations and convergence properties. In [Section IV](#) we illustrate some numerical results for several values of

the constant θ , highlighting the improved performance for $\theta = 1.5$.

II. PROBLEM FORMULATION

Consider problem (1) under the following assumptions:

Assumption 1. For $i = 1, \dots, N$:

- (i) $C_i : \mathbb{R}^n \rightarrow \mathbb{R}^{r_i}$ are linear operators.
- (ii) $f_i : \mathbb{R}^n \rightarrow \overline{\mathbb{R}}$, $g_i : \mathbb{R}^{r_i} \rightarrow \overline{\mathbb{R}}$ are proper closed convex functions, where $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$.
- (iii) The set of minimizers of (1), denoted by S^* , is nonempty.

We are interested in solving problem (1) in a distributed fashion. Specifically, let $G = (V, E)$ be an undirected graph over the vertex set $V = \{1, \dots, N\}$ with edge set $E \subset V \times V$. It is assumed that each node $i \in V$ is associated with a separate agent, and each agent maintains its own cost components f_i, g_i, C_i which are assumed to be private, and its own opinion of the solution $x_i \in \mathbb{R}^n$. The graph imposes communication constraints over agents. In particular, agent i can communicate directly only with its neighbors $j \in \mathcal{N}_i = \{j \in V \mid (i, j) \in E\}$. We make the following assumption.

Assumption 2. Graph G is connected.

With this assumption, we reformulate the problem as

$$\begin{aligned} & \underset{x \in \mathbb{R}^{Nn}}{\text{minimize}} \sum_{i=1}^N f_i(x_i) + g_i(C_i x_i) \\ & \text{subject to } x_i = x_j \quad (i, j) \in E \end{aligned}$$

where $x = (x_1, \dots, x_N)$. Associate any orientation to the unordered edge set E . Let $M = |E|$ and $B \in \mathbb{R}^{N \times M}$ be the *oriented node-arc incidence matrix*, where each column is associated with an edge $(i, j) \in E$ and has $+1$ and -1 in the i -th and j -th entry, respectively. Notice that the sum of each column of B is equal to 0. Let d_i denote the degree of a given vertex, that is, the number of vertices that are adjacent to it. We have $BB^T = \mathcal{L} \in \mathbb{R}^{N \times N}$, where \mathcal{L} is the graph *Laplacian* of G , *i.e.*,

$$\mathcal{L}_{ij} = \begin{cases} d_i & \text{if } i = j, \\ -1 & \text{if } i \neq j \text{ and node } i \text{ is adjacent to node } j, \\ 0 & \text{otherwise.} \end{cases}$$

Constraints $x_i = x_j$, $(i, j) \in E$ can be written in compact form as $Ax = 0$, where $A = B^T \otimes I_n \in \mathbb{R}^{Mn \times Nn}$. Therefore, the problem is expressed as

$$\underset{x \in \mathbb{R}^{Nn}}{\text{minimize}} \sum_{i=1}^N f_i(x_i) + g_i(C_i x_i) + \delta_{\{0\}}(Ax), \quad (2)$$

where δ_X denotes the indicator function of a closed nonempty convex set, X . The dual problem is:

$$\underset{\substack{y_i \in \mathbb{R}^{r_i} \\ w \in \mathbb{R}^{Mn}}}{\text{minimize}} \sum_{i=1}^N f_i^*(-A_i^T w - C_i^T y_i) + g_i^*(y_i), \quad (3)$$

where q^* denotes the Fenchel conjugate of a function q and $A_i \in \mathbb{R}^{Mn \times n}$ are the block columns of A . Let ∂q denote

the subdifferential of a convex function q . The primal-dual optimality conditions are

$$\begin{cases} 0 \in \partial f_i(x_i) + C_i^\top y_i + A_i^\top w, & i = 1, \dots, N \\ C_i x_i \in \partial g_i^*(y_i) & i = 1, \dots, N \\ \sum_{i=1}^N A_i x_i = 0, \end{cases} \quad (4)$$

where $w \in \mathbb{R}^{Mn}$, $y_i \in \mathbb{R}^{r_i}$, for $i = 1, \dots, N$. The following condition will be assumed throughout the rest of the paper.

Assumption 3. *There exist $x_i \in \text{ri dom } f_i$ such that $C_i x_i \in \text{ri dom } g_i$, $i = 1, \dots, N$ and $\sum_{i=1}^N A_i x_i = 0^1$.*

This assumption guarantees that the set of solutions to (4) is nonempty (see [17, Proposition 4.3(iii)]). If $(\mathbf{x}^*, \mathbf{y}^*, w^*)$ is a solution to (4), then \mathbf{x}^* is a solution to the primal problem (2) and (\mathbf{y}^*, w^*) to its dual (3).

III. DISTRIBUTED PRIMAL-DUAL ALGORITHMS

In this section we provide the main distributed algorithm that is based on *Asymmetric Forward-Backward-Adjoint* (AFBA), a new operator splitting technique introduced recently [2]. This special case belongs to the class of primal-dual algorithms. The convergence results include both the primal and dual variables and are based on [2, Propositions 5.4]). However, the convergence analysis here focuses on the primal variables for clarity of exposition, with the understanding that a similar error measure holds for the dual variables.

Our distributed algorithm consists of two phases, a local phase and the phase in which each agent interacts with its neighbors according to the constraints imposed by the communication graph. Each iteration has the advantage of only requiring local matrix-vector products and proximal updates. Specifically, each agent performs 2 matrix-vector products per iteration and transmits a vector of dimension n to its neighbors.

Before continuing we recall the definition of Moreau's proximal mapping. Let U be a symmetric positive-definite matrix. The *proximal mapping* of a proper closed convex function $f : \mathbb{R}^n \rightarrow \bar{\mathbb{R}}$ relative to $\|\cdot\|_U$ is defined by

$$\text{prox}_f^U(x) = \underset{z \in \mathbb{R}^n}{\text{argmin}} f(z) + \frac{1}{2} \|x - z\|_U^2,$$

and when the superscript U is omitted the same definition applies with respect to the canonical norm.

Let $\mathbf{u} = (\mathbf{x}, \mathbf{v})$ where $\mathbf{v} = (\mathbf{y}, w)$ and $\mathbf{y} = (y_1, \dots, y_N)$. The optimality conditions in (4), can be written in the form of the following monotone inclusion:

$$0 \in D\mathbf{u} + M\mathbf{u} \quad (5)$$

with

$$D(\mathbf{x}, \mathbf{y}, w) = (\partial \mathbf{f}(\mathbf{x}), \partial \mathbf{g}^*(\mathbf{y}), 0), \quad (6)$$

¹ $\text{dom } f$ denotes the domain of function f and $\text{ri } C$ is the relative interior of the set C .

and

$$M = \begin{bmatrix} 0 & C^\top & A^\top \\ -C & 0 & 0 \\ -A & 0 & 0 \end{bmatrix},$$

where $\mathbf{f}(\mathbf{x}) = \sum_{i=1}^N f_i(x_i)$, $\mathbf{g}^*(\mathbf{y}) = \sum_{i=1}^N g_i^*(y_i)$, $C = \text{blkdiag}(C_1, \dots, C_N)$. Notice that $Ax = \sum_{i=1}^N A_i x_i$, $A^\top w = (A_1^\top w, \dots, A_N^\top w)$. The operator $D + M$ is maximally monotone [18, Proposition 20.23, Corollary 24.4(i)].

Monotone inclusion (5), i.e., the primal-dual optimality conditions (4), is solved by applying [2, Algorithm 6]. This results in the following iteration:

$$\mathbf{x}^{k+1} = \text{prox}_{\Sigma^{-1}}^{\Sigma^{-1}}(\mathbf{x}^k - \Sigma C^\top \mathbf{y}^k - \Sigma A^\top w^k) \quad (7a)$$

$$\bar{\mathbf{y}}^k = \text{prox}_{\Gamma^*}^{\Gamma^*}(\mathbf{y}^k + \Gamma C(\theta \mathbf{x}^{k+1} + (1-\theta)\mathbf{x}^k)) \quad (7b)$$

$$\bar{w}^k = w^k + \Pi A(\theta \mathbf{x}^{k+1} + (1-\theta)\mathbf{x}^k) \quad (7c)$$

$$\mathbf{y}^{k+1} = \bar{\mathbf{y}}^k + (2-\theta)\Gamma C(\mathbf{x}^{k+1} - \mathbf{x}^k) \quad (7d)$$

$$w^{k+1} = \bar{w}^k + (2-\theta)\Pi A(\mathbf{x}^{k+1} - \mathbf{x}^k) \quad (7e)$$

where matrices Σ, Γ, Π play the role of stepsizes and are assumed to be positive definite. The iteration (7) can not be implemented in a distributed fashion because the dual vector w consists of M blocks corresponding to the edges. The key idea that allows distributed computations is to introduce the sequence

$$(\rho_i^k)_{k \in \mathbb{N}} = (A_i^\top w^k)_{k \in \mathbb{N}}, \quad \text{for } i = 1, \dots, N. \quad (8)$$

This transformation replaces the stacked edge vector w^k with corresponding node vectors ρ_i . More compactly, letting $\boldsymbol{\rho}^k = (\rho_1^k, \dots, \rho_N^k)$, it follows from (7c) and (7e) that

$$\boldsymbol{\rho}^{k+1} = \boldsymbol{\rho}^k + A^\top \Pi A(2\mathbf{x}^{k+1} - \mathbf{x}^k), \quad (9)$$

where $A^\top \Pi A$ is the *weighted graph Laplacian*. Since w^k in (7a) appear as $A^\top w^k$ we can rewrite the iteration:

$$\begin{aligned} \mathbf{x}^{k+1} &= \text{prox}_f^{\Sigma^{-1}}(\mathbf{x}^k - \Sigma C^\top \mathbf{y}^k - \Sigma \boldsymbol{\rho}^k) \\ \bar{\mathbf{y}}^k &= \text{prox}_{\Gamma^*}^{\Gamma^*}(\mathbf{y}^k + \Gamma C(\theta \mathbf{x}^{k+1} + (1-\theta)\mathbf{x}^k)) \\ \mathbf{y}^{k+1} &= \bar{\mathbf{y}}^k + (2-\theta)\Gamma C(\mathbf{x}^{k+1} - \mathbf{x}^k) \\ \boldsymbol{\rho}^{k+1} &= \boldsymbol{\rho}^k + A^\top \Pi A(2\mathbf{x}^{k+1} - \mathbf{x}^k) \end{aligned}$$

Set

$$\begin{aligned} \Sigma &= \text{blkdiag}(\sigma_1 I_n, \dots, \sigma_N I_n), \\ \Gamma &= \text{blkdiag}(\tau_1 I_{r_1}, \dots, \tau_N I_{r_N}), \\ \Pi &= \text{blkdiag}(\pi_1 I_n, \dots, \pi_M I_n), \end{aligned}$$

where $\sigma_i > 0, \tau_i > 0$ for $i = 1, \dots, N$ and $\pi_l > 0$ for $l = 1, \dots, M$. Consider a bijective mapping between $l = 1, \dots, M$ and unordered pairs $(i, j) \in E$ such that $\kappa_{i,j} = \kappa_{j,i} = \pi_l$. Notice that π_l for $l = 1, \dots, M$ are step sizes to be selected by the algorithm and can be viewed as weights for the edges. Thus, iteration (7) gives rise to our distributed algorithm:

Algorithm 1

Inputs: $\sigma_i > 0, \tau_i > 0, \kappa_{i,j} > 0$ for $j \in \mathcal{N}_i, i = 1, \dots, N$, $\theta \in [0, \infty[$, initial values $x_i^0 \in \mathbb{R}^n, y_i^0 \in \mathbb{R}^{r_i}, \rho_i^0 \in \mathbb{R}^n$.

for $k = 1, \dots$ **do**

for each agent $i = 1, \dots, N$ **do**

 Local steps:

$$\begin{aligned} x_i^{k+1} &= \text{prox}_{\sigma_i f_i}(x_i^k - \sigma_i \rho_i^k - \sigma_i C_i^\top y_i^k) \\ \bar{y}_i^k &= \text{prox}_{\tau_i g_i^*}(y_i^k + \tau_i C_i(\theta x_i^{k+1} + (1-\theta)x_i^k)) \\ y_i^{k+1} &= \bar{y}_i^k + \tau_i(2-\theta)C_i(x_i^{k+1} - x_i^k) \\ u_i^k &= 2x_i^{k+1} - x_i^k \end{aligned}$$

 Exchange of information with neighbors:

$$\rho_i^{k+1} = \rho_i^k + \sum_{j \in \mathcal{N}_i} \kappa_{i,j}(u_i^k - u_j^k)$$

Notice that each agent i only requires $u_j^k \in \mathbb{R}^n$ for $j \in \mathcal{N}_i$ during the communication phase. Before proceeding with convergence results, we define the following for simplicity of notation:

$$\bar{\sigma} = \max\{\sigma_1, \dots, \sigma_N\},$$

$$\bar{\tau} = \max\{\tau_1, \dots, \tau_N, \pi_1, \dots, \pi_M\},$$

$$L = \mathcal{L} \otimes I_n + C^\top C, \text{ where } \mathcal{L} \text{ is the graph Laplacian.}$$

It must be noted that the results in this section only provide choices of parameters that are sufficient for convergence. They can be selected much less conservatively by formulating and solving sufficient conditions that they must satisfy as *linear matrix inequalities* (LMIs). Due to lack of space we do not pursue this direction further, instead we plan to consider it in an extended version of this work.

Theorem 1. *Let Assumptions 2 and 3 hold true. Consider the sequence $(\mathbf{x}^k)_{k \in \mathbb{N}} = (x_1^k, \dots, x_N^k)_{k \in \mathbb{N}}$ generated by Algorithm 1. Assume the maximum stepsizes, i.e., $\bar{\sigma}$ and $\bar{\tau}$ defined above, are positive and satisfy*

$$\bar{\sigma}^{-1} - \bar{\tau}(\theta^2 - 3\theta + 3)\|L\| > 0, \quad (10)$$

for a fixed value of $\theta \in [0, \infty[$. Then the sequence $(\mathbf{x}^k)_{k \in \mathbb{N}}$ converges to (x^*, \dots, x^*) for some $x^* \in S^*$. Furthermore, if $\theta = 2$ the strict inequality (10) is replaced with $\bar{\sigma}^{-1} - \bar{\tau}\|L\| \geq 0$.

Proof. Algorithm 1 is an implementation of [2, Algorithm 6]. Thus convergence of $(\mathbf{x}^k)_{k \in \mathbb{N}}$ to a solution of (2) is implied by [2, Proposition 5.4]. Combining this with Assumption 2 yields the result. Notice that in that work the step sizes are assumed to be scalars for simplicity. It is straightforward to adapt the result to the case of diagonal matrices. \square

In Algorithm 1 when $\theta = 2$, we recover the algorithm of Chambolle and Pock [6]. One important observation is that the term $\theta^2 - 3\theta + 3$ in (10) is always positive and achieves its minimum at $\theta = 1.5$. This is a choice of interest for us since it results in larger stepsizes, $\sigma_i, \tau_i, \kappa_{i,j}$, and consequently better performance as we observe in numerical simulations.

Next, we provide easily verifiable conditions for f_i and g_i , under which linear convergence of the iterates can be established. We remark that these are just sufficient and certainly less conservative conditions can be provided but

are omitted for clarity of exposition. Let us first recall the following definitions from [19], [20]:

Definition 1 (Piecewise Linear-Quadratic). *A function $f : \mathbb{R}^n \rightarrow \overline{\mathbb{R}}$ is called piecewise linear-quadratic (PLQ) if its domain can be represented as union of finitely many polyhedral sets, relative to each of which $f(x)$ is given by an expression of the form $\frac{1}{2}x^\top Qx + d^\top x + c$, for some $c \in \mathbb{R}$, $d \in \mathbb{R}^n$, and $D \in \mathbb{R}^{n \times n}$.*

The class of piecewise linear-quadratic functions has been much studied and has many desirable properties (see [19, Chapter 10 and 11]). Many practical applications involve PLQ functions such as quadratic function, $\|\cdot\|_1$, indicator of polyhedral sets, hinge loss, etc. Thus, the R -linear convergence rate that we establish in Theorem 2 holds for a wide range of problems encountered in control, machine learning and signal processing.

Definition 2 (Metric subregularity). *A set-valued mapping $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is metrically subregular at z for z' if $(z, z') \in \text{gra } F$ and there exists $\eta \in [0, \infty[$, a neighborhood \mathcal{U} of z and \mathcal{V} of z' such that*

$$d(x, F^{-1}z') \leq \eta d(z', Fx \cap \mathcal{V}) \text{ for all } x \in \mathcal{U}, \quad (11)$$

where $\text{gra } F = \{(x, u) | u \in Fx\}$ and $d(\cdot, X)$ denotes the distance from set X .

Theorem 2. *Consider Algorithm 1 under the assumptions of Theorem 1. Assume f_i and g_i for $i = 1, \dots, N$, are piecewise linear-quadratic functions. Then the set valued mapping $T = D + M$ is metrically subregular at any z for any z' provided that $(z, z') \in \text{gra } T$. Furthermore, the sequence $(\mathbf{x}^k)_{k \in \mathbb{N}}$ converges R -linearly² to (x^*, \dots, x^*) for some $x^* \in S^*$.*

Proof. Function $\mathbf{f}(x) = \sum_{i=1}^N f_i(x_i)$ is piecewise linear-quadratic since f_i for $i = 1, \dots, N$, are assumed to be PLQ. Similarly, it follows from [19, Theorem 11.14 (b)] that \mathbf{g}^* is piecewise linear-quadratic. The subgradient mapping of a proper closed convex PLQ function is piecewise polyhedral, i.e. its graph is the union of finitely many polyhedral sets [19, Proposition 12.30 (b)]. This shows that D defined in (6) is piecewise polyhedral. Since the image of a polyhedral under affine transformation remains piecewise polyhedral, and M is a linear operator, graph of $T = D + M$ is piecewise polyhedral. Consequently, its inverse T^{-1} is piecewise polyhedral. Thus by [20, Proposition 3H.1] the mapping T^{-1} is calm at any z' for any z satisfying $(z', z) \in \text{gra } T^{-1}$. This is equivalent to the metric subregularity characterization of the operator T [20, Theorem 3H.3]. The second part of the proof follows directly by noting that [2, Algorithm 6] used to derive Algorithm 1 is a special case of [2, Algorithm 1]. Therefore, linear convergence follows from first part of the proof together with [2, Theorem 3.3]. The aforementioned

²The sequence $(x_n)_{n \in \mathbb{N}}$ converges to x^* R -linearly if there is a sequence of nonnegative scalars $(v_n)_{n \in \mathbb{N}}$ such that $\|x_n - x^*\| \leq v_n$ and $(v_n)_{n \in \mathbb{N}}$ converges Q -linearly³ to zero.

³The sequence $(x_n)_{n \in \mathbb{N}}$ converges to x^* Q -linearly with Q -factor given by $\sigma \in]0, 1[$, if for n sufficiently large $\|x_{n+1} - x^*\| \leq \sigma \|x_n - x^*\|$ holds.

theorem guarantees linear convergence for the stacked vector \mathbf{u} in (5), however, here we consider the primal variables only. \square

A. Special Case

Consider the following problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^N f_i(x), \quad (12)$$

where $f_i : \mathbb{R}^n \rightarrow \overline{\mathbb{R}}$ for $i = 1, \dots, N$ are proper closed convex functions. This is a special case of (1) when $g_i \circ C_i \equiv 0$. Since functions g_i are absent, the dual variables y_i in Algorithm 1 vanish and for any choice of θ the algorithm reduces to:

$$\begin{aligned} x_i^{k+1} &= \text{prox}_{\sigma_i f_i}(x_i^k - \sigma_i \rho_i^k) \\ u_i^k &= 2x_i^{k+1} - x_i^k \\ \rho_i^{k+1} &= \rho_i^k + \sum_{j \in \mathcal{N}_i} \kappa_{i,j}(u_i^k - u_j^k). \end{aligned}$$

Thus setting $\theta = 1.5$ in (10) to maximize the stepsizes yields $\bar{\sigma}^{-1} - \frac{3\bar{\tau}}{4}\|\mathcal{L}\| > 0$, where \mathcal{L} is the graph Laplacian.

IV. NUMERICAL SIMULATIONS

We now illustrate experimental results obtained by applying the proposed algorithm to the following problem:

$$\underset{x}{\text{minimize}} \lambda \|x\|_1 + \sum_{i=1}^N \frac{1}{2} \|D_i x - d_i\|_2^2 \quad (13)$$

for a positive parameter λ . This is the ℓ_1 regularized least-squares problem. Problem (13) is of the form (1) if we set for $i = 1, \dots, N$

$$\begin{aligned} f_i(x) &= \frac{\lambda}{N} \|x\|_1, \\ g_i(z) &= \frac{1}{2} \|z - d_i\|_2^2, \\ C_i &= D_i \end{aligned} \quad (14)$$

where $D_i \in \mathbb{R}^{m_i \times n}$, $d_i \in \mathbb{R}^{m_i}$. For the experiments we used graphs of $N = 50$ computing agents, generated randomly according to the Erdős-Renyi model, with parameter $p = 0.05$. In the experiments we used $n = 500$ and generated D_i randomly with normally distributed entries, with $m_i = 50$ for all $i = 1, \dots, N$. Then we generated vector d_i starting from a known solution for the problem and ensuring $\lambda < 0.1 \|\sum_{i=1}^N D_i^\top d_i\|_\infty$.

For the stepsize parameters we set $\sigma_i = \bar{\sigma}$, $\tau_i = \bar{\tau}$, for all $i = 1, \dots, N$, and $\kappa_{i,j} = \kappa_{j,i} = \bar{\tau}$ for all edges $(i, j) \in E$, such that (10) is satisfied. In order to have a fair comparison we selected $\bar{\sigma} = \alpha/\|\mathcal{L}\|$ and $\bar{\tau} = 0.99/(\alpha(\theta^2 - 3\theta + 3))$ with $\alpha = 20$ which was set empirically based on better performance of all the algorithms.

The results are illustrated in Figure 1, for several values of θ , where the distribution of the number of communication rounds required by the algorithms to reach a relative error of 10^{-6} is reported. In Figure 2 the convergence of algorithms is illustrated in one of the instances. It should be noted that the algorithm of Chambolle and Pock, that corresponds to

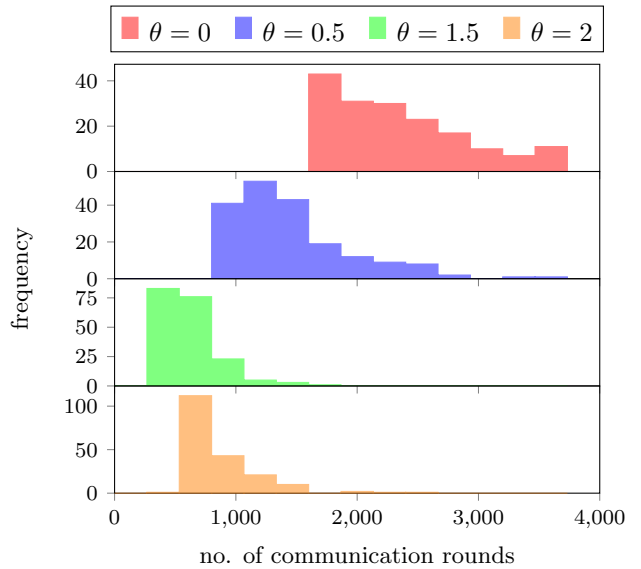


Fig. 1. Distribution of the number of communication rounds required by the algorithms to achieve a relative error of 10^{-6} , for fixed data and 200 randomly generated Erdős-Renyi graphs, with parameter $p = 0.05$.

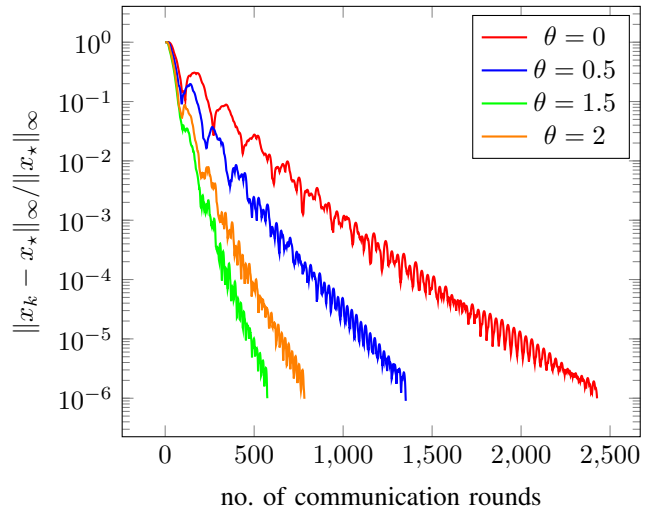


Fig. 2. Convergence of the relative error for the algorithms, in one of the considered instances.

$\theta = 2$, is generally slower than the case $\theta = 1.5$. This is mainly due to the larger stepsize parameters guaranteed by Theorem 1.

V. CONCLUSIONS

In this paper we illustrated how the recently proposed Asymmetric Forward-Backward-Adjoint splitting method (AFBA) can be used for solving distributed optimization problems where a set of N computing agents, connected in a graph, need to minimize the sum of N functions. The resulting Algorithm 1 only involves local exchange of variables (i.e., among neighboring nodes) and therefore no central authority is required to coordinate them. Moreover, the single nodes only require direct computations of the objective terms, and do not need to perform inner iterations and matrix inversions. Numerical experiments highlight that

Algorithm 1 performs generally better when the parameter θ is set equal to 1.5 in order to achieve the largest step-sizes. Future investigations on this topic include the study of how the topological structure of the graph underlying the problem affects the convergence rate of the proposed methods, as well as problem preconditioning in a distributed fashion. Developing asynchronous versions of the algorithm is another important future research direction.

REFERENCES

- [1] P. L. Combettes and I.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering*. Springer, 2011, pp. 185–212.
- [2] P. Latafat and P. Patrinos, "Asymmetric forward-backward-adjoint splitting for solving monotone inclusions involving three operators," *arXiv preprint arXiv:1602.08729*, 2016.
- [3] B. C. Vũ, "A splitting algorithm for dual monotone inclusions involving cocoercive operators," *Advances in Computational Mathematics*, vol. 38, no. 3, pp. 667–681, 2013.
- [4] L. Condat, "A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *Journal of Optimization Theory and Applications*, vol. 158, no. 2, pp. 460–479, 2013.
- [5] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-norm support vector machines," *Advances in neural information processing systems*, vol. 16, no. 1, pp. 49–56, 2004.
- [6] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.
- [7] L. M. Briceño-Arias and P. L. Combettes, "A monotone + skew splitting model for composite monotone inclusions in duality," *SIAM Journal on Optimization*, vol. 21, no. 4, pp. 1230–1250, 2011.
- [8] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.
- [9] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2012.
- [10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.
- [11] N. Parikh and S. Boyd, "Block splitting for distributed optimization," *Mathematical Programming Computation*, vol. 6, no. 1, pp. 77–102, 2014.
- [12] A. Teixeira, E. Ghadimi, I. Shames, H. Sandberg, and M. Johansson, "Optimal scaling of the ADMM algorithm for distributed quadratic programming," in *IEEE 52nd Annual Conference on Decision and Control (CDC)*, 2013, pp. 6868–6873.
- [13] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *IEEE 51st Annual Conference on Decision and Control (CDC)*, 2012, pp. 5445–5450.
- [14] —, "On the $O(1/k)$ convergence of asynchronous distributed alternating direction method of multipliers," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, 2013, pp. 551–554.
- [15] A. Makhdomi and A. Ozdaglar, "Broadcast-based distributed alternating direction method of multipliers," in *52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2014, pp. 270–277.
- [16] P. Bianchi and W. Hachem, "A primal-dual algorithm for distributed optimization," in *IEEE 53rd Annual Conference on Decision and Control (CDC)*, Dec 2014, pp. 4240–4245.
- [17] P. L. Combettes and J.-C. Pesquet, "Primal-dual splitting algorithm for solving inclusions with mixtures of composite, Lipschitzian, and parallel-sum type monotone operators," *Set-Valued and variational analysis*, vol. 20, no. 2, pp. 307–330, 2012.
- [18] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer Science & Business Media, 2011.
- [19] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*. Springer Science & Business Media, 2009, vol. 317.
- [20] A. L. Dontchev and R. T. Rockafellar, "Implicit functions and solution mappings," *Springer Monographs in Mathematics*. Springer, vol. 208, 2009.