A Partition-Based Implementation of the Relaxed ADMM for Distributed Convex Optimization over Lossy Networks

N. Bastianello†, M. Todescato‡, R. Carli†, L. Schenato‡

Abstract—In this paper we propose a distributed implementation of the relaxed Alternating Direction Method of Multipliers (R-ADMM) for optimization of a separable convex cost function, whose terms are stored by a set of interacting agents, one for each agent. Specifically the local cost stored by each node is in general a function of both the state of the node and the states of its neighbors, a framework that we refer to as ‘partition-based’ optimization. This framework presents a great flexibility and can be adapted to a large number of different applications. We show that the partition-based R-ADMM algorithm we introduce is linked to the relaxed Peaceman-Rachford Splitting (R-PRS) operator which, historically, has been introduced in the literature to find the zeros of sum of functions. Interestingly, making use of non expansive operator theory, the proposed algorithm is shown to be provably robust against random packet losses that might occur in the communication between neighboring nodes. Finally, the effectiveness of the proposed algorithm is confirmed by a set of compelling numerical simulations run over random geometric graphs subject to i.i.d. random packet losses.

Index Terms—distributed optimization, partition-based optimization, ADMM, operator theory, splitting methods, Peaceman-Rachford operator

I. INTRODUCTION

Because of the advent of the Internet-of-Things (IoT), we are witnessing the proliferation of large-scale systems in which a multitude of locally networked peers are able to asynchronously and unreliably stream information with neighboring peers. In such networks, many system-wide applications, e.g., in the machine learning field [1], as well as operational requirements, e.g., state estimation and control, can be cast as optimization problems aiming at globally optimal configurations across the entire network. Yet, many of these applications, because of the locally connected nature of such systems, are characterized by local decision functions which are influenced only by local information flows among neighboring peers. Examples owning to the above framework can be found in applications such as state estimation and power flow control in Smart Electric Grids [2] as well as cooperative localization in Wireless Networks [3], just to mention a few. Such class of problems, referred to as partition-based optimization problems, can be formally described as

\[
\min_{x_i, i \in \mathcal{V}} \sum_{i=1}^{N} f_i(x_i, \{x_j\}_{j \in \mathcal{N}_i})
\]

where each peer \(i\), in the set \(\mathcal{V} = \{1, \ldots, N\}\) of all possible peers, is responsible for a local decision function \(f_i\) only affected by its local piece of information \(x_i\) as well as by \(x_j\) of its neighboring peers \(j \in \mathcal{N}_i\).

Regarding partitioned-based optimization, many different approaches have been analyzed in the recent literature. In [3] a dual decomposition approach is proposed. Gradient-based schemes are considered in [4], [2]. In [4], the authors present a Block-Jacobi iteration suitable for quadratic programming while in the more recent [2] a modified generalized scheme is presented for generic convex optimization. Other solutions involve the well-know Alternating Direction Method of Multipliers (ADMM) [5], [6] which has been shown to be particularly suited for parallel and distributed computations. We refer the reader to [7], [8], [9], [10], [11], [12] for an overview of possible applications and convergence results. In particular, to the best of the authors knowledge, a partitioned-based ADMM scheme has been first introduced in [13] to solve for cooperative localization in WSN, while in [14] partition-based ADMM is applied to MPC.

However, when dealing with asynchronous and, in particular, faulty/unreliable communications, i.e., subject to delays and packet drops, while first-order [4], [2] and second-order [15], [16] gradient-based schemes have been proved to be robust, the same results do not apply to ADMM schemes. More specifically, works devoted to the study of asynchronous ADMM implementations can be found. Examples are [17], where convergence of the ADMM is shown when only a subset of coordinates is randomly updated at every time instant, and the recent [18], where a framework for asynchronous operations is proposed. Conversely, literature on robustness analysis of ADMM in the presence of faulty communication is still scarce and usually confined to specific setups such as [19], [20] where only bounded delays and a particular master-slave communication architecture are considered. To the authors knowledge, first steps toward more general results have appeared only recently in [21] where a robust generalized ADMM scheme for consensus optimization is presented.

In this paper we take over from [21]. We leverage nonexpansive operator theory where the underlying idea is to reformulate the original optimization problem into an equivalent form whose solutions corresponds to the fixed points of a suitable
operator. One particular class of such operators is represented by splitting methods, in which the burden of solving for the fixed points is alleviated by breaking down the computations into several steps. Well-known examples are the Peaceman-Rachford Splitting (PRS) [22] with its generalized versions as well as the Douglas-Rachford Splitting (DRS) [23], [24]. Moreover we refer to [25], [26] for more details and possible applications to asynchronous setups.

We start our analysis from the fact that ADMM can be shown to be equivalent to the DRS applied to the Lagrange dual of the original problem [27]. Then, the contribution of the paper is twofold. First, we present a reformulated version of ADMM suitable for partition-based optimization. Second, by resorting to results on stochastic operator theory, we formally prove robustness of the proposed algorithm to faulty communications.

The remainder of the paper is organized as follows. Section II reviews the classical ADMM algorithm and its generalized version, referred to as Relaxed-ADMM. Section III introduces the partition-based framework for consensus optimization and derives the relaxed ADMM applied to this problem. Section IV describes the proposed robust implementation. Section V collects some numerical simulations. Finally, Section VI draws some concluding remarks. Due to space constraints, all the technical proofs can be found in the Appendices.

II. THE RELAXED-ADMM ALGORITHM

Consider the following optimization problem

\[
\min_{x \in \mathcal{X}, y \in \mathcal{Y}} \{ f(x) + g(y) \}
\]

s.t. \( Ax + By = c \) \hspace{1cm} (2)

with \( \mathcal{X} \) and \( \mathcal{Y} \) Hilbert spaces, \( f : \mathcal{X} \rightarrow \mathbb{R} \cup \{+\infty\} \) and \( g : \mathcal{Y} \rightarrow \mathbb{R} \cup \{+\infty\} \) closed, proper and convex functions. In the following we assume that the above problem has solution. Let us define the augmented Lagrangian for the problem \( \text{(2)} \) as

\[
\mathcal{L}_\rho(x, y; w) = f(x) + g(y) - w^T (Ax + By - c) + \frac{\rho}{2} \|Ax + By - c\|^2 \hspace{1cm} (3)
\]

where \( \rho > 0 \) and \( w \) is the vector of Lagrange multipliers.

The Relaxed-ADMM (R-ADMM) algorithm (see [25]) consists in the alternating of the following three steps

\[
g(k + 1) = \arg \min_y \{ \mathcal{L}_\rho(x(k), y; w(k)) \}
\]

\[
w(k + 1) = w(k) - \rho (Ax(k) + By(k) - c) \hspace{1cm} (4)
\]

\[
x(k + 1) = \arg \min_x \mathcal{L}_\rho(x, y(k + 1); w(k + 1)). \hspace{1cm} (5)
\]

The R-ADMM algorithm can be derived applying the relaxed Peaceman-Rachford splitting operator to the Lagrange dual of problem \( \text{(2)} \) [7], [25]. It can be shown that, under the assumptions made on functions \( f \) and \( g \), the convergence of the R-ADMM algorithm is guaranteed if

\[
0 < \alpha < 1, \quad \rho > 0.
\]

We conclude this section by observing that setting \( \alpha = 1/2 \) one can retrieve the classical ADMM algorithm widely analyzed in [7].

III. DISTRIBUTED PARTITION-BASED CONVEX OPTIMIZATION

A. Problem Formulation

We start by formulating the problem we aim at solving.

Let \( \mathcal{G} = (\mathcal{V}, \mathcal{E}) \) be an undirected graph, where \( \mathcal{V} \) denotes the set of \( N \) vertices, labeled \( i \) through \( N \), and \( \mathcal{E} \) the set of edges. For \( i \in \mathcal{V} \), by \( \mathcal{N}_i \) we denote the set of neighbors of node \( i \) in \( \mathcal{G} \), namely,

\[
\mathcal{N}_i = \{ j \in \mathcal{V} : (i, j) \in \mathcal{E} \}.
\]

The state of each node is characterized by the local variable \( x_i \in \mathbb{R}^n \), and we are interested in solving the following optimization problem

\[
\min_{x_i, \mathcal{E}} \sum_{i=1}^{N} f_i(x_i, \{x_j\}_{j \in \mathcal{N}_i}) \hspace{1cm} (7)
\]

where \( f_i : \mathbb{R}^{nN} \rightarrow \mathbb{R} \cup \{+\infty\} \) are closed, proper, and convex functions and where \( f_i \) is known only to node \( i \). Note that writing \( f_i(x_i, \{x_j\}_{j \in \mathcal{N}_i}) \) denotes that \( f_i \) is a function of node \( i \)’s state and the states of its neighbors only. We refer to this framework as to partition-based convex optimization.

In the following we denote by \( x^* \in \mathbb{R}^{nN} \) the optimal solution of \( \text{(7)} \) with components \( x_i^* \in \mathbb{R}^n \).

In this work we seek for iterative and distributed algorithms solving problem in \( \text{(7)} \) where, by distributed, we mean that each node can exchange information only with its neighbors. To this goal, we provide an alternative formulation of \( \text{(7)} \) to which we can apply the relaxed ADMM reviewed in the previous Section.

As first step, we assume that each node \( i \) stores local copies of the variables its cost function depends on, denoted by

\[
x_i^{(i)} \text{ and } x_j^{(i)} \forall j \in \mathcal{N}_i;
\]

precisely, \( x_j^{(i)} \) is the local copy of the variable \( x_j \) stored in memory by node \( i \), while \( x_i^{(i)} = x_i \). Observe that, problem in \( \text{(7)} \) can be equivalently formulated as

\[
\min_{x_i^{(i)}, \mathcal{E}} \sum_{i=1}^{N} f_i(x_i^{(i)}, \{x_j^{(i)}\}_{j \in \mathcal{N}_i}) \hspace{1cm} (8)
\]

s.t. \( x_i^{(i)} = x_j^{(j)}, \forall (i, j) \in \mathcal{E} \)

\[\text{2Note that in general the local variables might have different dimensions, but here for simplicity they are assumed to be all vectors in } \mathbb{R}^n.\]
where the constraints impose that the consensus be reached, that is, all local copies of each vector \( x_i \) be equal.

Now, for each edge \((i,j) \in \mathcal{E}\), we introduce the bridge variables \( y_{i,j}^{(i,j)} \) and \( y_{j,i}^{(i,j)} \), and \( y_{i,j}^{(j,i)} \) and \( y_{j,i}^{(j,i)} \). Notice that the constraints in (8) can be rewritten as

\[
\begin{align*}
x_i^{(i,j)} &= y_{i,j}^{(i,j)}, & x_j^{(i)} &= y_{j,i}^{(i,j)} \\
x_j^{(j,i)} &= y_{i,j}^{(j,i)}, & x_j^{(j)} &= y_{j,i}^{(j,i)} & \forall (i,j) \in \mathcal{E}.
\end{align*}
\]

We define now the vectors

\[
x^{(i)} = \left\{ x_i^{(i,j)} \right\}_{j \in \mathcal{N}_i} \quad \text{and} \quad y^{(i)} = \left\{ y_{i,j}^{(i,j)} \right\}_{j \in \mathcal{N}_i} \quad \text{where} \quad x^{(i)} \in \mathbb{R}^{n(|\mathcal{N}_i|+1)} \quad \text{and} \quad y^{(i)} \in \mathbb{R}^{2n|\mathcal{N}_i|},
\]

and the overall vectors \( x = [x^{(1)} \top, \ldots, x^{(|\mathcal{V}|)} \top] \top \) and \( y = [y^{(1)} \top, \ldots, y^{(|\mathcal{V}|)} \top] \top \). Let \( f_i(x^{(i)}) \) be the local function associated to the constraint \((\mathcal{N}_i, f_i)\), then problem (8) with constraints (9) can be compactly rewritten as

\[
\begin{align*}
\min_{x,y} f(x) \\
\text{s.t.} \quad Ax + y &= 0 \\
y &= Py
\end{align*}
\]

for a suitable \( A \) matrix and with \( P \) being a permutation matrix that swaps \( y_{i,j}^{(i,j)} \) with \( y_{j,i}^{(j,i)} \). Making use of the indicator function \( \mathbb{1}_{(I-P)y=0} \) which is equal to \( 0 \) if \((I-P)y = 0\), and \( +\infty \) otherwise, we can finally rewrite problem (8) as

\[
\begin{align*}
\min_{x,y} \{ f(x) + \mathbb{1}_{(I-P)y} \} \\
\text{s.t.} \quad Ax + y &= 0
\end{align*}
\]

Clearly problem (10) conforms to the formulation of problem (2), therefore we can apply the R-ADMM to solve it, which is the focus of the following Section.

### B. R-ADMM for Partition-Based Optimization

In this Section we propose an implementation of the R-ADMM for the partition-based convex optimization problems introduced in Section III-A.

It is possible to show that the direct application of the R-ADMM algorithm to (10) is amenable of distributed implementation. More precisely, let \( w_{i,j}^{(i,j)} \) be the Lagrange multiplier associated to the constraint \( y_{i,j}^{(i,j)} = y_{i,j}^{(j,i)} \) and let

\[
w^{(i)} = \left[ \begin{array}{c}
w_{i,j}^{(i,j)} \\
w_{j,i}^{(i,j)} \\
\end{array} \right]_{j \in \mathcal{N}_i}.
\]

Assume that node \( i \) stores in memory \( x^{(i)} \), \( y^{(i)} \) and \( w^{(i)} \). Then one can see that these variables can be updated by node \( i \) (according to the updating equations of R-ADMM) only receiving the information \( x^{(j)} \), \( y^{(j)} \) and \( w^{(j)} \), for all \( j \in \mathcal{N}_i \); namely, only communications among neighboring nodes are required. We do not report the explicit equations, because they are quite unwieldy.

However, leveraging the particular structure of problem (10) and considering the fact that we are interested only into the trajectories of the variables \( x^{(i)} \), it is possible to provide a simpler implementation of the R-ADMM algorithm, in terms of memory and communication requirements; in particular, this lighter implementation, beside the variables \( x^{(i)} \) involve only the auxiliary variables \( z^{(i,j)} \), \( i \in \mathcal{V}, \ j \in \mathcal{N}_i \)

We have the following Proposition.

**Proposition 1:** The trajectories generated by the variables \( x^{(i)} \), \( i \in \mathcal{V} \), obtained by applying the R-ADMM algorithm in (4), (5), (6), to the problem in (10), starting from a given initial condition \( x^{(i)}(0), w^{(i)}(0) \), \( y^{(i)}(0) \) are identical to the trajectories generated by iterating the following equations

\[
x^{(i)}(k) = \arg \min_{x^{(i)}} \left\{ f_i(x^{(i)}) \right\} + \frac{\rho}{2} \left\| x^{(i)} - x^{(i)}(k-1) \right\|^2 + \frac{\rho}{2} \sum_{j \in \mathcal{N}_i} \left\| w_{i,j}^{(i,j)}(k) \right\|^2
\]

for all \( i \in \mathcal{V} \), and

\[
z^{(i,j)}(k+1) = (1 - \alpha) z^{(i,j)}(k) - \alpha z^{(j,i)}(k) + 2\alpha \rho \xi^{(i,j)}(k)
\]

\[
z^{(j,i)}(k+1) = (1 - \alpha) z^{(j,i)}(k) - \alpha z^{(i,j)}(k) + 2\alpha \rho \xi^{(j,i)}(k)
\]

for all \( j \in \mathcal{N}_i \), where the auxiliary variables are initialized as \( z^{(i,j)}(0) = w_{i,j}^{(i,j)}(0) + r y_{i,j}^{(i,j)}(0) \).

The proof of the previous Proposition can be found in Appendix I.

**Proposition 1** suggests a straightforward distributed implementation of the R-ADMM in which a node \( i \) locally stores and updates the variables \( x^{(i)} \) and \( z^{(i,j)} \), \( z^{(j,i)} \) for all \( j \in \mathcal{N}_i \). Within this implementation the node requires the auxiliary variables \( z^{(i,j)} \) and \( z^{(j,i)} \) to be sent by each of its neighbors in order to update the local \( x \) variables and hence the auxiliary variables.

An equivalent implementation can be obtained if node \( i \) stores and updates the \( z^{(i,j)} \) and \( z^{(j,i)} \) variables instead of the \( z^{(i,j)} \) and \( z^{(j,i)} \) variables. This implementation is formally described in Algorithm 1.

Observe, that, at the beginning of each iteration node \( i \) updates \( x^{(i)} \) based only on local information. Then it computes the temporary variables \( q^{(i,j)} + q^{(j,i)} \) which are sent to neighbor \( j \). At the same time, it receives the quantities \( q^{(j,i)} + q^{(i,j)} \) from neighbor \( j \) and it uses these information to update \( z^{(i,j)} \) and \( z^{(j,i)} \) as in (14).

The following Proposition characterizes the convergence properties of Algorithm 1 which follows from those of the R-PRS. The proof is available in Appendix I.

**Proposition 2:** For Algorithm 1 let \( (\alpha, \rho) \) be such that \( 0 < \alpha < 1 \) and \( \rho > 0 \). Then, for any initial conditions, the trajectories \( k \to x^{(i)}(k) \) and \( k \to x^{(i)}(k), \ i \in \mathcal{V} \),
Input: Set the termination condition $K > 0$. For each node $i$, initialize $x^{(i)}(0)$ and $\{z_i^{(j,i)}(0), z_j^{(j,i)}(0)\}_{j \in \mathcal{N}_i}$.

$k \leftarrow 0$

while $k < K$ each agent $i$

compute $x^{(i)}(k)$ according to (11);

for all $j \in \mathcal{N}_i$, compute the temporary variables

$$q_i^{(i,j)}(k) = -z_i^{(j,i)}(k) + 2px_i^{(i)}(k)$$

$$q_j^{(j,i)}(k) = -z_j^{(i,j)}(k) + 2pz_j^{(i)}(k)$$

for all $j \in \mathcal{N}_i$, transmit $\{q_i^{(i,j)}, q_j^{(j,i)}\}$ to node $j$; gather $\{q_j^{(j,i)}, q_j^{(j,i)}\}$ from each neighbor $j$;

update the auxiliary variables as

$$z_i^{(j,i)}(k+1) = (1 - \alpha)z_i^{(j,i)}(k) + \alpha q_i^{(i,j)}$$

$$z_j^{(j,i)}(k+1) = (1 - \alpha)z_j^{(j,i)}(k) + \alpha q_j^{(j,i)}$$

$k \leftarrow k + 1$

end

Algorithm 1: Modified partition-based R-ADMM.

$j \in \mathcal{N}_i$, generated by the Algorithm 1 converge to the optimal solution of (7), i.e.,

$$\lim_{k \to \infty} x_i^{(i)}(k) = x_i^* \quad \forall i \in \mathcal{V}, j \in \mathcal{N}_i.$$

Remark 1: The implementation of the R-ADMM presented in Algorithm 1 requires that each node stores and updates locally $3|\mathcal{N}_i| + 1$ variables. Moreover, a node has to transmit only two variables to each of its neighbors at each time instant.

IV. PARTITION-BASED R-ADMM OVER LOSSY NETWORKS

The partition-based algorithm described in the previous Section is proved to converge under the implicit assumption that the communication channels are reliable and, therefore, no packet loss occurs.

The aim of this Section is to prove the convergence of the partition-based R-ADMM in case the communications are unreliable and some transmissions between nodes might fail. Precisely, we make the following assumption.

Assumption 1: During any iteration of Algorithm 1 the communication from node $i$ to node $j$ can be lost with some probability $p$.

To formally describe the communication failures, we associate to each transmission a random variable which is equal to 1 if the packet is lost, 0 otherwise. Specifically we introduce the family of binary random variables $L^{(i,j)}(k)$, $k = 0, 1, 2, \ldots, i \in \mathcal{V}, j \in \mathcal{N}_i$, which are independent for $i, j$ and $k$ that vary, such that

$$\Pr \left[ L^{(i,j)} = 1 \right] = p, \quad \Pr \left[ L^{(i,j)} = 0 \right] = 1 - p.$$  

Accounting for the potential packet losses that we have introduced above, Algorithm 1 is modified as illustrated in Algorithm 2.

Input: Set the termination condition $K > 0$. For each node $i$, initialize $x^{(i)}(0)$ and $\{z_i^{(j,i)}(0), z_j^{(j,i)}(0)\}_{j \in \mathcal{N}_i}$.

$k \leftarrow 0$

while $k < K$ each agent $i$

compute $x^{(i)}(k)$ according to (11);

compute, for $j \in \mathcal{N}_i$, the temporary variables

$$q_i^{(i,j)}(k) = -z_i^{(j,i)}(k) + 2px_i^{(i)}(k)$$

$$q_j^{(j,i)}(k) = -z_j^{(i,j)}(k) + 2pz_j^{(i)}(k)$$

transmit, for $j \in \mathcal{N}_i$, $\{q_i^{(i,j)}, q_j^{(j,i)}\}$ to node $j$; for $j \in \mathcal{N}_i$, if $\{q_j^{(j,i)}, q_j^{(j,i)}\}$ is received do

update the auxiliary variables as

$$z_i^{(j,i)}(k+1) = (1 - \alpha)z_i^{(j,i)}(k) + \alpha q_i^{(i,j)}$$

$$z_j^{(j,i)}(k+1) = (1 - \alpha)z_j^{(j,i)}(k) + \alpha q_j^{(j,i)}$$

$k \leftarrow k + 1$

end

Algorithm 2: Robust partition-based R-ADMM.

In this potentially lossy scenario, during the $k$-th iteration, node $i$ updates the local variables $x_i^{(i)}$ and $\{x_j^{(j,i)}\}_{j \in \mathcal{N}_i}$, according to (11). Then, it computes the temporary variables $\{q_i^{(i,j)}, q_j^{(j,i)}\}$ for each of its neighbors $j \in \mathcal{N}_i$ and transmits them. If neighbor $j$ receives the packet, that is, if $L^{(i,j)}(k) = 0$, then it updates the auxiliary variables $z_i^{(j,i)}$ and $z_j^{(j,i)}$ using the received values $\{q_j^{(j,i)}, q_j^{(j,i)}\}$ according to (14), otherwise it leaves them unchanged. The updates for the auxiliary variables can be described in a compact way as follows

$$z_i^{(j,i)}(k+1) = L^{(i,j)}(k)z_i^{(j,i)}(k) + \left(1 - L^{(i,j)}(k) \right) \left(1 - \alpha \right)z_i^{(j,i)}(k) + \alpha q_i^{(j,i)}$$

$$z_j^{(j,i)}(k+1) = L^{(i,j)}(k)z_j^{(j,i)}(k) + \left(1 - L^{(i,j)}(k) \right) \left(1 - \alpha \right)z_j^{(j,i)}(k) + \alpha q_j^{(j,i)}.$$  

The following Proposition characterizes the convergence properties of Algorithm 2 running in the probabilistic lossy scenario described in Assumption 1.

Proposition 3: Consider Algorithm 2 under Assumption 1. Assume the pair of parameters $(\alpha, \rho)$ be such that $0 < \alpha < 1$ and $\rho > 0$. Then, for any initial conditions, the trajectories $k \to x_i^{(i)}(k)$ and $k \to x_j^{(j,i)}(k)$, $i \in \mathcal{V}, j \in \mathcal{N}_i$ generated by the Algorithm 2 converge almost surely to the optimal solution of (7), that is,

$$\lim_{k \to \infty} x_i^{(i)}(k) = x_i^* \quad \forall i \in \mathcal{V}, j \in \mathcal{N}_i.$$
with probability one.

Proving Proposition 3 is achieved by showing that the randomized partition-based ADMM of Algorithm 2 conforms to the stochastic Peaceman-Rachford splitting introduced in [28], [17] which is provably convergent. The details are available in Appendix [1]. We highlight that allowing the packet loss probability of each edge to be in general different, the partition-based R-ADMM still conforms to the stochastic PRS framework of [28], [17].

Remark 2: Note that we restrict our analysis to the case of synchronous communications and updates, with the aim of investigating the performance of the R-ADMM over faulty networks. The more realistic case of asynchronous communications will be the focus of future research.

Remark 3: Observe that both in the case of reliable communications of Proposition 2 and in the lossy scenario of Proposition 3, the convergence is guaranteed in the same region of the parameters space $(\alpha, \rho)$. In particular Algorithm 1 and the modified version 2 are shown to be provably convergent for $0 < \alpha < 1$ and $\rho > 0$. This is however only a sufficient result and the convergence might hold also in a larger region of the parameter space. Indeed this is verified by the simulations results described in [V] for the case of quadratic cost functions. Moreover, despite what suggested by the intuition, the larger the packet loss probability $p$, the larger the region of convergence (though only slightly). However, this increased region of stability is counterbalanced by a slower convergence rate of the algorithm.

Remark 4: It is worth to remarking that the convergence of the partition-based ADMM can be proved, both in the perfect communication scenario and in the scenario with potential packet losses, also with a time-varying step-size. Therefore it would be possible to design step-size choice criteria that speed up the convergence of the R-ADMM, which will be the object of future works.

V. SIMULATIONS

This Section describes the simulative results obtained testing the effectiveness of the proposed partition-based Algorithm 2. In particular we are interested into the performance of the algorithm in the presence of packet losses due to unreliable communications. Our analysis is restricted to the case of quadratic cost functions defined as

$$ f_1(x^{(i)}_i, \{x^{(i)}_j\}_{j \in N_i}) = \|A_{ii}x^{(i)}_i + \sum_{j \in N_i} A_{ij}x^{(i)}_j - b_i\|^2_{Q_i}, $$

where $A_{ii} \in \mathbb{R}^{r \times n}$, $A_{ij} \in \mathbb{R}^{r \times n}$ for any $j \in N_i$, $b_i \in \mathbb{R}^r$, and $Q_i \in \mathbb{R}^{r \times r}$ is a symmetric and positive definite matrix. Notice that the weighted norm used in the cost function is defined as $\|v\|^2_M = v^\top M v$ for a matrix $M$ of suitable dimensions. In general the matrices $A_{ii}$ and $\{A_{ij}\}_{j \in N_i}$ are different for each node, as well as the cost matrices $Q_i$. Notice that when dealing with quadratic cost functions the solution of Equation (11) can be found in closed form.

Moreover, we consider the family of random geometric graphs with $N = 10$ and communication radius $r = 0.1$[p.u.] in which, that is, two nodes are connected if and only if their relative distance is less than $r$.

All the results are obtained by averaging over a set of 100 Monte Carlo runs of the simulations.

In Figure 1 we depict the evolution of the relative error

$$ \log\sum_{i=1}^N \frac{\|x^{(i)}_i(k) - x^{*}_i\|}{\|x^{*}_i\|} \tag{15} $$

for different values of the packet loss probability, and with fixed step size $\alpha = 0.75$ and penalty parameter $\rho = 3$. The presence of communication failures clearly has the effect of slowing down the convergence rate of the algorithm.

In Figure 2 we report the stability boundaries of the partition-based R-ADMM for different packet loss probabilities as functions of the tunable parameters, the step size $\alpha$ and the penalty $\rho$. In particular each curve in Figure 2 represents the numerical boundary below which the algorithm is found to be convergent, and above which it is found to be divergent. The results are quite interesting and will be a direction of future investigation. As was expected from the convergence
result of Proposition 3, the convergence is guaranteed for any value of the penalty parameter $\rho$. However as the packet loss probability increases, the stability region with respect to the step size broadens. Therefore, somewhat counterintuitively, the greater the probability $p$ is, the larger the stability regions in the $(\rho, \alpha)$ space are; however this phenomenon is balanced by slower convergence rates, as depicted in Figure 1.

The role of the tunable parameters is investigated next. Figure 3 represents the evolution of the relative error (15) for different values of the step size $\alpha$ and with fixed packet loss probability $p = 0.2$ and penalty $\rho = 3$. The use of the relaxed ADMM, instead of the classic ADMM which coincides with the R-ADMM for $\alpha = 0.5$, clearly can be beneficial for the speed of convergence. In particular, inside the convergence region guaranteed by Proposition 3, that is $0 < \alpha < 1$, the rate of convergence results to be larger for values of the step size that are larger than $1/2$. Finally, Figure 4 depicts the relative error (15) for different values of the penalty parameter $\rho$, with step size set to $\alpha = 0$.

Fig. 3: Evolution, in log-scale, of the relative error of Alg. 2 computed w.r.t. the unique optimal solution $x^*$ as function of different values of the step size $\alpha$, with fixed packet loss probability $p = 0.2$ and penalty $\rho = 3$. Average over 100 Monte Carlo runs.

The relative error (15) for different values of the penalty parameter $\rho$, with step size set to $\alpha = 0.75$ and packet loss probability to $p = 0.2$. Recall that by Proposition 3 the convergence of Algorithm 3 is guaranteed when the condition $\rho > 0$ is satisfied. Therefore Figure 4 shows that it is possible to make use of the penalty to speed up the convergence rate of the algorithm.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper we have presented a formulation of the relaxed ADM to distributed convex optimization with partition-based cost functions, that is, the local cost stored by each node depends on both its own state and the states of its neighbors. We have first introduced the framework describing the partition-based scenario, and then we have showed how it is possible to reformulate it so that the R-ADMM can be properly applied. Moreover we have presented an implementation of the R-ADMM that has lower memory and communications requirements than the implementation derived from a straightforward application of the algorithm.

The formulation of the partition-based R-ADMM that we have introduced turns out to be provably robust to random communication failures. In particular, we have rigorously proved that the region of convergence of the algorithm in the lossy scenario does not deteriorate compared to the case of reliable communications. An interesting numerical result shows that the presence of communication failures with larger probability increases the size of the region of convergence in the space of the tunable parameters. The role of the step size and the penalty has been analyzed as well. Future research will deal with to the analysis of the asynchronous case, and with the rigorous mathematical characterization of the convergence regions.

APPENDIX I

PROOF OF PROPOSITION 1

As we showed in Section III-A of the main paper, it is possible to reformulate the partition-based problem (8) so that it conforms to problem

$$
\min_x \{ f(x) + \iota_{(I-P)}(y) \} \\
\text{s.t. } Ax + y = 0
$$

(16)

to which the R-ADMM can be applied. The three update equations (4), (5) and (6) that characterize the R-ADMM applied to problem (16) yield

$$
y(k + 1) = \arg\min_y \{ L_\rho(x(k); y; w(k)) \\
+ \rho(2\alpha - 1)(y, (Ax(k) + y(k))) \} \\
(17)
$$

$$
w(k + 1) = w(k) - \rho(Ax(k) + y(k + 1)) \\
- \rho(2\alpha - 1)(Ax(k) + y(k + 1)) \\
(18)
$$

$$
x(k + 1) = \arg\min_x L_\rho(x, y(k + 1); w(k + 1)) \\
(19)
$$

where $w$ is the vector of Lagrange multipliers and the augmented Lagrangian is

$$
L_\rho(x, y; w) = f(x) + \iota_{(I-P)}(y) - w^\top(Ax + y) \\
+ \frac{\rho}{2} \| Ax + y \|^2.
$$
However, as shown in [25], the R-ADMM for problem (16) can be equivalently characterized with the set of four iterates

\[
y(k) = \arg \min_{y=\rho y} \left\{ -z^T(k)y + \frac{\rho}{2}\|y\|^2 \right\} \\
w(k) = z(k) - \rho y(k) \\
x(k) = \arg \min_x \left\{ f(x) - (2w(k) - z(k))^T Ax + \frac{\rho}{2}\|Ax\|^2 \right\} \\
z(k+1) = (1-2\alpha)z(k) + 2\alpha(w(k) - \rho Ax(k)).
\]

Therefore Equations (11) and (12) can be derived from (26) and (27) using the particular structure of the problem, proving Proposition 1.

**APPENDIX II**

**PROOF OF PROPOSITIONS 2 AND 3**

As was mentioned above, the partition-based problem can be reformulated as (16) which can be solved by the application of the R-ADMM. Therefore both the convergence results of Propositions 2 and 3 follow from those of Propositions 2 and 3 of [21]. Indeed the R-ADMM is guaranteed to converge in both the lossless and lossy scenarios as long as the step-size and penalty parameters are such that \(0 < \alpha < 1\) and \(\rho > 0\). Moreover, the components of the primal variables vector, which in the partition-based case are the subvectors \(x_i^{(i)}\), are guaranteed to converge to the optimum value, that is, each variable \(x_i^{(i)}\) converges to the optimum \(x_i^*\).

**REFERENCES**

[1] K. Slavakis, G. B. Giannakis, and G. Mateos, “Modeling and optimization for big data analytics,” IEEE Signal Processing Magazine, vol. 31, no. 5, pp. 19–31, 2014.

[2] M. Todescato, N. Bof, G. Cavraro, R. Carli, and L. Schenato, “Generalized gradient optimization over lossy networks for partition-based estimation,” arXiv preprint arXiv:1710.10829, 2017.

[3] R. Carli and G. Notarstefano, “Distributed partition-based optimization via dual decomposition,” in Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on. IEEE, 2013, pp. 2979–2984.

[4] M. Todescato, G. Cavraro, R. Carli, and L. Schenato, “A robust block-jacobi algorithm for quadratic programming under lossy communications,” IFAC-PapersOnLine, vol. 48, no. 22, pp. 126–131, 2015.

[5] R. Glowinski and A. Marroco, “Sur l’approximation, par éléments finis d’ordre un, et la résolution, par pénalisation-dualité d’une classe de problèmes de dirichlet non linéaires,” ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique, vol. 9, no. R2, pp. 41–76, 1975. [Online]. Available: http://eudml.org/doc/193269

[6] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximation,” Computers & Mathematics with Applications, vol. 2, no. 1, pp. 17–40, 1976.

[7] 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, 2011.

[8] M. Fukushima, “Application of the alternating direction method of multipliers to separable convex programming problems,” Computational Optimization and Applications, vol. 1, no. 1, pp. 93–111, 1992.

[9] J. Eckstein and M. Fukushima, “Some reformulations and applications of the alternating direction method of multipliers,” in Large scale optimization. Springer, 1994, pp. 115–134.

[10] I. Eckstein and D. P. Bertsekas, “On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators,” Mathematical Programming, vol. 55, no. 1, pp. 293–318, 1992.

[11] G. Chen and M. Teboulle, “A proximal-based decomposition method for convex minimization problems,” Mathematical Programming, vol. 64, no. 1-3, pp. 81–101, 1994.

[12] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, “Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems,” IEEE Transactions on Automatic Control, vol. 60, no. 3, pp. 644–658, March 2015.

[13] T. Erseghe, “A distributed and scalable processing method based upon admm,” IEEE Signal Processing Letters, vol. 19, no. 9, pp. 563–566, 2012.

[14] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, “Distributed optimization with local domains: Applications in mpc and network flows,” IEEE Transactions on Automatic Control, vol. 60, no. 7, pp. 2004–2009, 2015.
[15] R. Carli, G. Notarstefano, L. Schenato, and D. Varagnolo, “Distributed quadratic programming under asynchronous and lossy communications via newton-raphson consensus,” in Control Conference (ECC), 2015 European. IEEE, 2015, pp. 2514–2520.
[16] ———, “Analysis of newton-raphson consensus for multi-agent convex optimization under asynchronous and lossy communications,” in Decision and Control (CDC), 2015 IEEE 54th Annual Conference on. IEEE, 2015, pp. 418–424.
[17] P. Bianchi, W. Hachem, and F. Iutzeler, “A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization,” IEEE Transactions on Automatic Control, vol. 61, no. 10, pp. 2947–2957, 2016.
[18] Z. Peng, Y. Xu, M. Yan, and W. Yin, “Arocks: an algorithmic framework for asynchronous parallel coordinate updates,” SIAM Journal on Scientific Computing, vol. 38, no. 5, pp. A2851–A2879, 2016.
[19] R. Zhang and J. Kwok, “Asynchronous distributed admm for consensus optimization,” in International Conference on Machine Learning, 2014, pp. 1701–1709.
[20] T.-H. Chang, M. Hong, W.-C. Liao, and X. Wang, “Asynchronous distributed admm for large-scale optimization—part i: algorithm and convergence analysis,” IEEE Transactions on Signal Processing, vol. 64, no. 12, pp. 3118–3130, 2016.
[21] N. Bastianello, M. Todescato, R. Carli, and L. Schenato, “Distributed optimization over lossy networks via relaxed peaceman-rachford splitting: a robust admm approach,” in European Control Conference (ECC), 2018. IEEE, 2018.
[22] D. W. Peaceman and H. H. Rachford, Jr, “The numerical solution of parabolic and elliptic differential equations,” Journal of the Society for industrial and Applied Mathematics, vol. 3, no. 1, pp. 28–41, 1955.
[23] J. Douglas and H. H. Rachford, “On the numerical solution of heat conduction problems in two and three space variables,” Transactions of the American mathematical Society, vol. 82, no. 2, pp. 421–439, 1956.
[24] P.-L. Lions and B. Mercier, “Splitting algorithms for the sum of two nonlinear operators,” SIAM Journal on Numerical Analysis, vol. 16, no. 6, pp. 964–979, 1979.
[25] D. Davis and W. Yin, “Convergence rate analysis of several splitting schemes,” in Splitting Methods in Communication, Imaging, Science, and Engineering. Springer, 2016, pp. 115–163.
[26] R. Hannah and W. Yin, “On unbounded delays in asynchronous parallel fixed-point algorithms,” arXiv preprint arXiv:1609.04746, 2016.
[27] J. Eckstein and W. Yao, “Augmented lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results,” RUTCOR Research Reports, vol. 32, 2012.
[28] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, “Asynchronous distributed optimization using a randomized alternating direction method of multipliers,” in Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on. IEEE, 2013, pp. 3671–3676.