Convergence of the Robust Block-Jacobi Algorithm in Presence of Packet Losses

M. Todescato * **G. Cavraro** * **R. Carli** * **L. Schenato** *

* *Department of Information Engineering, Padova, 35131, Italy.*
(e-mail: todescat|cavraro|carlirug|schenato @ dei.unipd.it).

1. MATHEMATICAL PRELIMINARIES

In this technical note, $\mathcal{G}(\mathcal{V},\mathcal{E})$ denotes a directed graph where $\mathcal{V}=\{1,...,N\}$ is the set of vertices and $\mathcal{E}$ is the set of directed edges, i.e., a subset of $\mathcal{V}\times\mathcal{V}$. More precisely the edge (i,j) is incident on node i and node j and is assumed to be directed away from i and directed toward j. The graph $\mathcal{G}$ is said to be bidirected if $(i,j)\in\mathcal{E}$ implies $(j,i)\in\mathcal{E}$. Given a directed graph $\mathcal{G}(\mathcal{V},\mathcal{E})$, a directed path in $\mathcal{G}$ consists of a sequence of vertices $(i_1,i_2,\dots,i_r)$ such that $(i_j,i_j+1)\in\mathcal{E}$ for every $j\in\{1,\dots,r-1\}$. The directed graph $\mathcal{G}$ is said to be *strongly connected* if for any pair of vertices (i,j) there exists a directed path connecting i to j. Given the directed graph $\mathcal{G}$, the set of neighbors of node i, denoted by $\mathcal{N}_i$, is given by $\mathcal{N}_i=\{j\in\mathcal{V}\,|\,(i,j)\in\mathcal{E}\}$. Moreover, $\mathcal{N}_i^+=\mathcal{N}_i\cup i$. Given a directed graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ with $|\mathcal{E}|=M$ let the *incidence matrix* $\mathcal{A}\in\mathbb{R}^{M\times N}$ of $\mathcal{G}$ be defined as $\mathcal{A}=[a_{ei}]$, where $a_{ei}=1,-1,0$, if edge e is incident on node i and directed away from it, is incident on node i and directed toward it, or is not incident on node i, respectively. Given a vector v with v^T we denote its transpose. Moreover, we denote with A_d the *adjacency matrix* or *laplacian matrix* of the graph which is defined as $\mathcal{A}_d:=\mathcal{A}^T\mathcal{A}$, which has the property that $[\mathcal{A}_d]_{ij}\neq 0$ if and only if $(i,j)\in\mathcal{E}$. If we associate to each edge a weight different from one, then it is possible to define the *weighted laplacian matrix* as $\mathcal{L}=\mathcal{A}^TW\mathcal{A}$ where $W\in\mathbb{R}^{M\times M}$ represents the diagonal matrix containing in its ℓ-th element the weight associated to the ℓ-th edge. Given a vector v with the symbols $\Re(v)$ and $\Im(v)$ we denote its real and imaginary parts, respectively. Finally, with the symbols $\mathbb{E}$ and $\mathbb{P}$ we denote, respectively, the expectation operator and the probability of an event.

2. PROBLEM FORMULATION

Consider the set of N agents $\mathcal{V}=\{1,\dots,N\}$, where each agent i is described by its state vector $x_i\in\mathbb{R}^{n_i}$. Assume the agents can communicate among themselves through a bidirected strongly connected *communication graph* $\mathcal{G}(\mathcal{V},\mathcal{E})$. Assume each agent collects a set of measurements $b_i\in\mathbb{R}^{m_i}$ which are noisy linear combinations of its own state and those of its neighboring agents, i.e.,

$$b_i=\sum_{j\in\mathcal{N}_i^+}A_{ij}x_j+w_i=A_{ii}x_i+\sum_{j\in\mathcal{N}_i}A_{ij}x_j+w_i$$

where $A_{ij}\in\mathbb{R}^{m_i\times n_j}$ and where w_i is white noise of zero mean and variance R_i independent of the other w_j. We consider the problem of estimating the entire state of the network from the knowledge of the noisy measurements. By collecting all the agents state and measurements in the vectors $x=[x_1^T,\dots,x_N^T]^T\in\mathbb{R}^n$ (where $n=\sum_i n_i$) and $b:=[b_1^T,\dots,b_N^T]\in\mathbb{R}^m$ (where $m=\sum_i m_i$), respectively, it is possible to formulate the problem as a classical *weighted least square* problem. That is

$$\min_x J(x), \tag{1}$$

where

$$J(x)=\frac{1}{2}(Ax-b)^TR^{-1}(Ax-b)\,. \tag{2}$$

The matrix $A\in\mathbb{R}^{m\times n}$ represents the measurements matrix, whose ij-th block is simply defined as $[A]_{ij}=A_{ij}$, while $R\in\mathbb{R}^{m\times m}$ is the block diagonal matrix defined as

$$R=\text{blkdiag}\{R_1,\dots,R_N\}\,,$$

which represents the noise variance.
From now on, we assume that $n\leq m$ and that A is full rank. Under these assumptions, it is well known that the solution of (1) is unique and given by

$$x^*=(A^TR^{-1}A)^{-1}A^TR^{-1}b\,. \tag{3}$$

To compute the value of x^* directly as in (3), one needs all the measurements, the matrix A and the noise variance R, i.e., full knowledge of the network is required. On the contrary, we aim at solving Problem (1) in a distributed fashion. To this end, note that it is possible to rewrite Problem (1) as

$$\min_{x_1,\dots,x_N}\ \sum_{i=1}^N J_i(x_i,\{x_j\}_{j\in\mathcal{N}_i})\,, \tag{4}$$

where

$$J_i(x_i,\{x_j\}_{j\in\mathcal{N}_i})=\frac{1}{2}(A_{ii}x_i+\sum_{j\in\mathcal{N}_i}A_{ij}x_j-b_i)^TR_i^{-1}(A_{ii}x_i+\sum_{j\in\mathcal{N}_i}A_{ij}x_j-b_i)\,.$$

The above equation highlights the local dependence of each cost function J_i on information regarding only agent i and its neighbors $j\in\mathcal{N}_i$. In next section, we present a distributed algorithm to solve optimization problems in the separable quadratic form (4). Firstly, we review an implementation which requires synchronous communications among the agents and basically coincides with a *Block-Jacobi* algorithm. Secondly, we present a modified version which is amenable for a distributed implementation and is robust to the presence of packet losses in the communication channel.

3. BLOCK JACOBI ALGORITHM

3.1 Distributed Block Jacobi Algorithm

Consider the *generalized gradient* descent strategy

$$x(t+1) = x(t) - \epsilon D^{-1} \nabla J(x(t)) \tag{5}$$

where $\nabla J(x(t))$ is the gradient of J, i.e., $\nabla J(x(t)) = [\partial J(x(t))/\partial x]^T$, D is a generic positive definite matrix and ϵ a suitable positive constant, usually referred as *step size*. The algorithm we propose is a particular case of (5), where D is a block diagonal matrix whose i-th diagonal block is defined as

$$D_i = \sum_{j \in \mathcal{N}_i^+} A_{ji}^T R_j^{-1} A_{ji}. \tag{6}$$

With a little abuse of notation, let us denote the Hessian of the cost function as $\nabla^2 J(x)$. From standard algebraic computations, it follows that $\nabla^2 J(x) = A^T R^{-1} A$. The matrix $\nabla^2 J(x)$ can be partitioned as an $N \times N$ block matrix, where the i,j-th block $\left[\nabla^2 J(x)\right]_{ij}$ is given by $\frac{\partial J(x)}{\partial x_i \partial x_j}$. One can see that the block $[\nabla^2 J(x)]_{ij}$ is different from zero either if $j \in \mathcal{N}_i^+$ or if i and j are *two step neighbors* (i.e. there exists a agent k such that k is neighbor of both i and j). Furthermore, it can be shown that $D_i = [\nabla^2 J(x)]_{ii}$ is the i-th diagonal block of the cost function Hessian.

From (2), we can compute the gradient of the cost function

$$\nabla J(x(t)) = A^T R^{-1}(Ax(t) - b), \tag{7}$$

whose component associated with the i-th agent is

$$[\nabla J(x(t))]_i = \sum_{j \in \mathcal{N}_i^+} A_{ji}^T R_j^{-1} n_j(t+1) \tag{8}$$

where the variable $n_j(t+1)$ is defined as

$$n_j(t+1) = \sum_{k \in \mathcal{N}_j^+} A_{jk} x_k(t) - b_j. \tag{9}$$

By plugging (6), (8) into (5), we can write the updating step performed by agent i as

$$x_i(t+1) = x_i(t) - \epsilon D_i^{-1} \sum_{j \in \mathcal{N}_i^+} A_{ji}^T R_j^{-1} n_j(t+1), \tag{10}$$

which, in vector form, leads to

$$x(t+1) = (I - \epsilon D^{-1} A^T R^{-1} A) x(t) + \epsilon D^{-1} A^T R^{-1} b. \tag{11}$$

Observe that agent i, in order to perform (10), needs information coming from the neighbours of its neighbors, i.e. the two-step neighbours. As so, to each iteration of the algorithm it is necessary to perform two communications, the first to compute the $n_i(t+1)$'s and the second to compute the $x_i(t+1)$'s. The *distributed Block Jacobi algorithm* (denoted hereafter as the BJ algorithm) for quadratic functions is formally described as in Algorithm 1. Next, the convergence properties of the BJ algorithm are established.

Lemma 1. Consider Problem (1) and the BJ algorithm. Assume

$$\epsilon \leq \frac{2}{\|D^{-\frac{1}{2}} A^T R^{-1} A D^{-\frac{1}{2}}\|}. \tag{12}$$

Then, for any $x(0) \in \mathbb{R}^n$, the trajectory $x(t)$, generated by the BJ algorithm, converges exponentially fast to the minimizer of Problem (1), i.e.,

$$\|x(t) - x^*\| \leq C\rho^t$$

for some constants $C > 0$ and $0 < \rho < 1$.

Proof. Consider the change of variables $\tilde{x} = x - x^*$. The cost function becomes

$$f(\tilde{x}) = \tilde{x}^T \frac{A^T R^{-1} A}{2} \tilde{x} + c,$$

while the evolution of $\tilde{x}$ is given by

$$\tilde{x}(t+1) = (I - \epsilon D^{-1} A^T R^{-1} A)\tilde{x}(t)$$

By imposing $f(\tilde{x}(t+1)) - f(\tilde{x}(t)) < 0$, after some simple computations, it turns out that if equation (12) holds, then the algorithm reaches the minimizer of (1).

Algorithm 1 Distributed Block Jacobi algorithm.

Require: $\forall i \in \mathcal{V}$, store $A_{ij}, A_{ji}, R_j, j \in \mathcal{N}_i^+$.

1: **for** $t \in \mathbb{N}$ each $i \in \mathcal{V}$ **do**
2: sends $x_i(t)$ to $j \in \mathcal{N}_i$;
3: receives $x_j(t)$ from $j \in \mathcal{N}_i$;
4: updates $n_i(t)$ by using (9)
5: sends $n_i(t)$ to $j \in \mathcal{N}_i$;
6: receives $n_j(t)$ from $j \in \mathcal{N}_i$;
7: updates $x_i(t)$ by using (10)
8: **end for**

3.2 Robust Block Jacobi Algorithm

Algorithm 1 has been designed for the ideal case with no lossy communication. In the following, we generalize Algorithm 1 for the case with lossy communication, e.g. agent i could not receive information sent by some of its neighbors, due to communication failures. The modification of the algorithm is apparently naive, since we simple perform the same algorithm by using the last received data from its neighbours if a packet is not received.

To model the packet losses, it is convenient to introduce the indicator function

$$\gamma_j^{(i)}(t) = \begin{cases} 1 & \text{if } i \text{ received the information sent by } j \\ 0 & \text{otherwise.} \end{cases}$$

with the assumption that $\gamma_i^{(i)}(t) = 1$, since node i has always access to its local variables $n_i(t)$ and $x_i(t)$. We assume the following property.

Assumption 2. There exists a constant T such that, for all $t \geq 0$, for all $i \in \mathcal{V}$ and for all $j \in \mathcal{N}_i$,

$$\mathbb{P}\left[\{\gamma_j^{(i)}(t), \ldots, \gamma_j^{(i)}(t+T)\} = \{0, \ldots, 0\}\right] = 0.$$

Roughly speaking, Assumption 2 states that agent i receives, at least once, information coming from agent j within any window of T iterations of the algorithm. Observe that, if agent i does not receive some of the packets transmitted by its neighbors, then it does not have the necessary information to perform the updates (9) and (10). To overcome this fact, we assume agent i stores in memory the auxiliary variables $x_j^{(i)}, n_j^{(i)}$, $j \in \mathcal{N}_i$, which are equal, respectively, to the last packets x_j and n_j received by agent i from agent j; specifically, the dynamics of $x_j^{(i)}, n_j^{(i)}$ are

$$n_j^{(i)}(t+1) = \begin{cases} n_j(t) & \text{if } \gamma_j^{(i)}(t) = 1 \\ n_j^{(i)}(t) & \text{if } \gamma_j^{(i)}(t) = 0 \end{cases} \tag{13}$$

$$x_j^{(i)}(t+1) = \begin{cases} x_i(t) & \text{if } \gamma_j^{(i)}(t) = 1 \\ x_j^{(i)}(t) & \text{if } \gamma_j^{(i)}(t) = 0 \end{cases}. \tag{14}$$

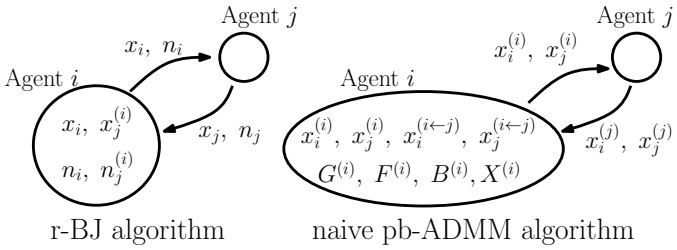

Fig. 1. Communication scheme for the BJ algorithm (left) and for the ADMM algorithm (right).

As mentioned in the previous section, Algorithm 1 requires two communication rounds every iteration. In a lossy environment, in order to reduce the communication burden and the number of communication failures, we modify Algorithm 1 by letting the agents to communicate just once every iteration, transmitting together the x_i's and the n_i's. Agents i exploit $x_j^{(i)}(t)$ and $n_j^{(i)}(t)$ to update n_i and x_i as

$$n_i(t+1) = \sum_{j\in\mathcal{N}_i^+} A_{ij}x_j^{(i)}(t) - b_i \tag{15}$$

$$x_i(t+1) = x_i(t) - \epsilon D_i^{-1} \sum_{j\in\mathcal{N}_i^+} A_{ji}^T R_j^{-1} n_j^{(i)}(t) \tag{16}$$

As so, even in the scenario with no packet losses, this new algorithm does not exactly coincide with Algorithm 1, since a one-step delay is introduced in the computation of the variables $n_i(t)$. The *robust block Jacobi algorithm* (hereafter referred to as r-BJ algorithm) for quadratic functions is formally described as in Algorithm 2. The

Algorithm 2 Robust block Jacobi algorithm.

Require: $\forall i \in \mathcal{V}$, store $A_{ij}, A_{ji}, R_j, j \in \mathcal{N}_i^+$.
1: **for** $t \in \mathbb{N}$ each $i \in \mathcal{V}$ **do**
2: sends $x_i(t), n_i(t)$ to $j \in \mathcal{N}_i$;
3: **if** $\gamma_j^{(i)}(t) = 1$ **then**
4: receives $x_j(t)$ and $n_j(t)$ from $j \in \mathcal{N}_i$
5: **end if**
6: updates $n_i(t)$ by using (15)
7: updates $n_j^{(i)}(t)$ by using (13)
8: updates $x_j^{(i)}(t)$ by using (14)
9: updates $x_i(t)$ by using (16)
10: **end for**

left panel of Figure 1 provides a pictorial representation of the stored and communicated variables by each node. The convergence properties of the r-BJ algorithm are next established.

Theorem 3. Let Assumption 2 hold. Consider Problem (1) and the r-BJ algorithm. There exists $\bar{\epsilon}$ such that, if $0 < \epsilon < \bar{\epsilon}$, then, for any $x(0) \in \mathbb{R}^n$, the trajectory $x(t)$, generated by the BJ algorithm, converges exponentially fast to the minimizer of Problem (1), i.e.,

$$\|x(t) - x^*\| \leq C\rho^t$$

for some constants $C > 0$ and $0 < \rho < 1$.

The proof of Theorem 3 relies on the time scale separation of the dynamic of the x_i's and of the auxiliary variables $x_j^{(i)}$'s, n_i's and $n_j^{(i)}$'s, and fully exploits the following

Lemma 4. Consider the dynamical system

$$\begin{bmatrix} x(t+1) \\ y(t+1) \end{bmatrix} = \begin{bmatrix} I & \epsilon B \\ C(t) & F(t) \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} \tag{17}$$

Let the following assumptions hold

(1) $\forall x, \exists! y : y = C(t)x + F(t)y, \forall t$. As a consequence, we can write $y = Gx$;
(2) the system

$$z(t+1) = F(t)z(t) \tag{18}$$

is asymptotically stable;
(3) the system

$$\dot{\xi}(t) = -BGx(t) \tag{19}$$

is asymptotically stable.

Then, there exists $\bar{\epsilon}$, with $0 < \epsilon < \bar{\epsilon}$ such that the origin is an asymptotically stable equilibrium for the system (17).

Proof. [Proof of Theorem 3] Let us define $n^* = Ax^* - b$, $n_j^{(i)*} = n_j^*$ and $x_j^{(i)*} = x_j^*$, and consider the change of variables

$$\begin{aligned} \tilde{x} &= x - x^* \\ \tilde{x}_j^{(i)} &= x_j^{(i)} - x_j^* \\ \tilde{n} &= n - n^* \\ \tilde{n}_j^{(i)} &= n_j^{(i)} - n_j^* \end{aligned} \tag{20}$$

Let us collect all the auxiliary variables $\tilde{x}_j^{(i)}$'s, $\tilde{n}$'s and $\tilde{n}_j^{(i)}$'s in the vector $\tilde{y}$. Then, Algorithm 2 dynamic can be expressed as

$$\begin{bmatrix} \tilde{x}(t+1) \\ \tilde{y}(t+1) \end{bmatrix} = \begin{bmatrix} I & \epsilon B \\ C(t) & F(t) \end{bmatrix} \begin{bmatrix} \tilde{x}(t) \\ \tilde{y}(t) \end{bmatrix} \tag{21}$$

The proof's aim is to show that the system (21) satisfies the hypotheses of Lemma 17. In (21), $\tilde{x}$ will be the variable with a slow dynamic, while $\tilde{y}$ will be the variable with a fast dynamic.

Now, fix the slow variable value $\tilde{x} = \bar{x}$. It can be shown that the vector $\tilde{y}(\bar{x})$ that stacks the values $\tilde{x}_j^{(i)} = \bar{x}_j$, $\tilde{n} = A\bar{x}$, $\tilde{n}_j^{(i)} = \tilde{n}_j$, satisfies the condition $\tilde{y}(\bar{x}) = C(t)\bar{x} + D(t)\tilde{y}(\tilde{x})$, $\forall t, \bar{x}$. Furthermore, it can be easily found a matrix G such that $\tilde{y}(\bar{x}) = G\bar{x}$.

For what concerns the dynamic of the fast variables $\tilde{y}$, because of Assumption 2, we have that after a time lower or equal to $2T + 1$, $\tilde{y}$ will reach the value $\tilde{y}(\bar{x}) = G\bar{x}$. In fact, exploiting (13), (14) and the change of variables (20), in the worst case, T iteration are necessary to have $\tilde{x}_j^{(i)} = \bar{x}_j, \forall i$. After that, one iteration is necessary to compute $\tilde{n} = A\bar{x}$ and finally T iteration are necessary to have $\tilde{n}_j^{(i)} = \tilde{n}_j, \forall i$. As a result, the fast variable dynamic is exponentially stable, reaching the equilibrium in a finite number of iteration. That is, fixed $\bar{x}$, we have

$$\tilde{y}(t) = G\bar{x},\ \forall t \geq 2T + 1 \tag{22}$$

Furthermore, from (17) we have that

$$\begin{aligned}\tilde{y}(t) &= \prod_{k=0}^{t-1} F(k)y(0) + \prod_{k=1}^{t-1} F(k)C(0)\bar{x} + \dots \\ &\quad + \prod_{k=2}^{t-1} F(k)C(1)\bar{x} + \dots C(0)\bar{x} \\ &= \prod_{k=0}^{t-1} F(k)y(0) + \Phi\bar{x} \end{aligned} \tag{23}$$

Since equation (22) is satisfied for every initial condition $y(0)$, it turns out that

$$\prod_{k=0}^{t-1} F(k)y(0) = 0$$

for every $y(0)$ and for every sequence $F(0), \dots, F(t-1)$. As a consequence, the system (18) is asymptotically stable.

Now, consider the dynamical system

$$\dot{\xi}(t) = -BG\xi(t). \tag{24}$$

It can be shown that

$$BG = D^{-1}A^T R^{-1} A$$

and thus, choosing as a Lyapunov function

$$V(\xi) = \xi^T \frac{A^T R^{-1} A}{2} \xi,$$

we can see that system (24) is asymptotically stable.

As a result, system (21) satisfies the hypotheses of Lemma 17, and thus there exists $\bar{\epsilon}$, with $0 < \epsilon < \bar{\epsilon}$ such that, by using the robust block Jacobi Algorithm 2,

$$\lim_{t\to\infty} x(t) = x^*.$$

Remark 5. We would like to emphasize that this general model of packet losses includes as special cases asynchronous updates. In fact, asynchronous updates where only one node i updates its local variables based on the information received form its neighbours can be recovered by our algorithm assuming that all packets are lost except those from the neighbours of node i to node i itself. Also, our algorithm allows for multiple agents to communicate and perform updates at the same time, thus requiring no coordination. Finally, broadcast communication can be used since nodes do not need to establish reliable bidirectional communication as in gossip protocols.