Average Consensus with Asynchronous Updates and Unreliable Communication (with proofs)

Nicoletta Bof [*] **Ruggero Carli** [*] **Luca Schenato** [*]

[*] *All authors are with the Department of Information Engineering, University of Padova, Via Gradenigo 6/a,35131 Padova, Italy. (e-mails:*{`bofnicol,carlirug,schenato`}*@dei.unipd.it)*

Abstract: In this work we introduce an algorithm for distributed average consensus which is able to deal with asynchronous and unreliable communication systems. It is inspired by two algorithms for average consensus already present in the literature, one which deals with asynchronous but reliable communication and the other which deals with unreliable but synchronous communication. We show that the proposed algorithm is exponentially convergent under mild assumptions regarding the nodes update frequency and the link failures. The theoretical results are complemented with numerical simulations.

Keywords: Multi-agent systems, Distributed control and estimation, Control and estimation with data loss, Asynchronous algorithms, Average consensus

1. INTRODUCTION

In recent years, substantial effort has been dedicated towards the design of distributed algorithms for large-scale systems. The main driver for this is that, nowadays, the availability of small and cheap computational units is becoming widespread. As a consequence, it is affordable to develop extended systems to monitor and control many different environments. However, due to the size of this systems, it is not always possible to collect all the information in a single computational unit and sometimes it is neither advisable, since some of the information collected by the system could be sensible. Moreover, each unit is endowed with some computational power which will not be fully exploited otherwise. Another additional advantage of using a distributed approach is that the whole system is in a way safer, since it does not rely on a single unit but is assigned to many different ones. Distributed algorithms present two major drawbacks: memory and computational constraints and need of a reliable communication system. The first is due to the fact that the units which compose the system can be quite limited in their computational and storage capabilities, the second is intrinsic to the distributed setting, since each unit can exchange information only with its neighbours in order to perform a global task.

It is therefore fundamental to always consider the properties of the communication system adopted. One very important characteristic is the communication protocol, which can be synchronous or asynchronous. Synchronous protocols require substantial coordination between the nodes, and when the number of agents in the system increases, this coordination can become difficult to achieve. An asynchronous communication protocol, on the other hand, has no coordination requirements, but an algorithm which uses such a protocol could in general require more iterations because in each iteration of the algorithm only a subset of the nodes in the networks are activated. Another important feature of the communication system is its reliability due to the possibility that packets are lost during the transmission. Obviously, the system should not lose packets but perfect reliability could be difficult or too expensive to enforce; to deal with possible packet losses, either an acknowledgement scheme is developed or the algorithm is implemented in such a way that the loss of a packet is not detrimental for the convergence.

In this paper we describe and study a distributed algorithm for average consensus. Basically, each unit has a given scalar quantity and the aim for each node is to compute the average of all these quantities. Sensor networks represent a remarkable domain where the evaluation of the average of the measured quantities is required in several applications Xiao et al. (2005), Bolognani et al. (2010), Garin and Schenato (2010). However, differently from the rich literature on this topic, we adopt an asynchronous and unreliable communication system, and we allow the communication not to be bi-directional, that is if a unit communicates with another one, the converse is not assured. In a synchronous and reliable communication scenario, important works are Boyd et al. (2004), Olshevsky and Tsitsiklis (2009), Oreshkin et al. (2010), Domínguez-García and Hadjicostis (2011). When unreliability in the communication is introduced, some works have adopted the acknowledgement scheme Chen et al. (2010) or assumed that each unit can determine whether the communication works Patterson et al. (2007), Xiao et al. (2005). However, an acknowledgement scheme requires additional secondary transmissions, which slow down the entire algorithm and consume extra energy. Therefore, in context where the energy consumption is constrained, the latter scheme is not adoptable and the transmission has to be reduced only to essential information. In an asynchronous setting, Bénézit et al. (2010) introduce an algorithm that reaches average consensus using the so-called ratio consensus. A very interesting idea is introduced in Dominguez-Garcia et al. (2011) and Vaidya et al. (2011) where the adopted communication is synchronous and unreliable. In these works a robust and synchronous algorithm inspired by Bénézit et al. (2010) is introduced.

Adopting the idea of mass transfer given in Vaidya et al. (2011), but developing an asynchronous algorithm as done in Bénézit et al. (2010), we describe an algorithm for average consensus which is provably convergent to the average in an asynchronous and unreliable communication scenario. The convergence proof relies on the introduction of two assumptions concerning the communication scheme, one regarding the frequency of waking up of each node and the other regarding how many consecutive times a given link can fail.

Our interest in this algorithm is justified by its possible execution in more complex algorithms. In fact, even though it is an interesting stand-alone algorithm, there are some algorithms which need average consensus algorithm as a building block, e.g. the Newton-Raphson algorithm for convex optimization (see Varagnolo et al. (2016)), some distributed versions of the Kalman filter (see Cattivelli and Sayed (2010)) or some algorithms for energy resources distribution in power grids (see Dominguez-Garcia and Hadjicostis (2010)). However, to be used in such algorithms, the average consensus has to be exponentially convergent. The aforementioned works by Bénézit et al. (2010) and Vaidya et al. (2011), for example, do not show whether exponential convergence is guaranteed in their algorithms By proving this kind of convergence for our algorithm, we make possible to use it in more advanced algorithms which could then be applied in a realistic communication scenarios (i.e. asynchronous and unreliable channels).

2. NOTATION & COMMUNICATION PROTOCOLS

Given a scalar $x \in \mathbb{R}$, $|x|$ denotes its absolute value. Given a matrix $A \in \mathbb{R}^{N\times N}$, $[A]_{ij}$ denotes its $(i,j)-$th element, and $A^\top$ indicate its transpose. A vector $\mathbf{x}$ is strictly positive if $x_i > 0$, $\forall i \in \{1,\dots,N\}$. Given two vectors $\mathbf{x},\mathbf{z} \in \mathbb{R}^N$, with x_i or $[\mathbf{x}]_i$ we denote its $i-$th element and with $\mathbf{x}/\mathbf{z}$ the Hadamard division of the two vectors. I_N indicates the $N\times N$ identity matrix. A graph $\mathcal{G}$ is represented by the couple $(\mathcal{V},\mathcal{E})$, with $\mathcal{V}$ the set of nodes $\{1,\dots,N\}$ and $\mathcal{E} \subset \mathcal{V}\times\mathcal{V}$ the set of edges. The number of edges in the graph is E. We consider directed, strongly connected and static graphs, with all the nodes having a self loop. Given a node $i \in \mathcal{V}$, the set $\mathcal{N}^i_{\text{in}}$ contains all the neighbours which communicate to i, that is $\mathcal{N}^i_{\text{in}} = \{j \,|\, j \in \mathcal{V}, i \neq j,\, (j,i) \in \mathcal{E}\}$, while the set $\mathcal{N}^i_{\text{out}}$ contains all the neighbours to which i communicates, that is $\mathcal{N}^i_{\text{out}} = \{j \,|\, j \in \mathcal{V}, i \neq j,\, (i,j) \in \mathcal{E}\}$. For a set $\mathcal{A} \in \mathcal{V}$, $|\mathcal{A}|$ denotes the cardinality of the set. A matrix $P \in \mathbb{R}^{N\times N}$ is row stochastic if $P\mathbb{1}_N = \mathbb{1}_N$, where $\mathbb{1}_N$ is the all-ones vector of dimension N. Finally, if $a,b \in \mathbb{R}$, $a < b$, with $[a,b]$ we indicate the interval between a and b, extremes included.

In the following we briefly describe some of the communication protocols which are usually adopted in wireless sensors networks given a pre-assigned communication graph, namely, the *synchronous* protocols, as opposed to the

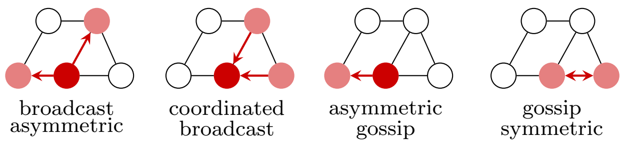

Fig. 1. Asynchronous communication protocols: the opaque red node is the one that wakes up (for the first 3 protocols), the other highlighted nodes are those which exchange information with it. In gossip symmetric there is no hierarchy in the nodes selected.

asynchronous ones like *broadcast asymmetric*, *coordinated broadcast*, *gossip asymmetric* and *gossip symmetric*.

In a **synchronous** protocol all the nodes activate at the same time instants and perform the updating and communication operations (almost) synchronously. This protocol requires a common notion of time among the nodes; indeed all the agents have to wake up simultaneously and so a perfect coordination is required. If the graph is moderately small, requiring synchronization may not be a great deal, but as the dimension of the network increases, synchronization can become an issue.

Instead, in **asynchronous** protocols, during each iteration, only a small subsets of all the nodes in the network perform the communication and updating steps. Specifically, in the **broadcast asymmetric** protocol, at each iteration, there is only one node transmitting information to its out-neighbours, which, based on the received messages, update their internal variables. At a given iteration, the (unique) transmitter node is said to be the one that wakes up (or turns on). The same terminology is applied to the node that performs the first step of the communication in the two protocols we describe next. The **coordinate broadcast** can be considered as the dual protocol of the broadcast asymmetric. Indeed, at each iteration, there is only one node which wakes up, but, instead of sending information, it polls all its in-neighbours in order to receive from them some desired messages. In **asymmetric gossip** again only one node wakes up but it sends information to only one of its out-neighbours, typically randomly chosen. Finally, the **symmetric gossip** is a protocol that requires bidirectional communication, that is the communication graph $\mathcal{G}$ has to be undirected (implying $\mathcal{N}^i_{\text{in}} = \mathcal{N}^i_{\text{out}}$ for all $i \in \mathcal{V}$); during each iteration an edge of the graph is selected and only the two nodes which are pointed by this edge exchange information with each other. Figure 1 gives a pictorial description of the asynchronous protocols just described.

3. PROBLEM FORMULATION

Consider N agents (also called nodes) which can communicate with each other according to a graph $\mathcal{G}$ and throughout some asynchronous communication protocol. We assume the communications to be unreliable, that is, some packet losses might occur during the transmissions of the messages. Each node $i \in \{1,\dots,N\}$ has a private scalar quantity [1] $v_i \in \mathbb{R}$, which can be collected in vector $\mathbf{v} \in \mathbb{R}^N$, and the problem to solve is the evaluation of the mean of these v_i, that is of $\bar{v} = \sum_i v_i/N$. The evaluation has to be carried out by each node in a distributed way,

[1] The algorithm can be modified to manage multidimensional quantities.

and the nodes can exchange information among themselves only if they are neighbours in the graph $\mathcal{G}$.
Formally, denoted by $x_i(k)$ the estimate of the mean $\bar{v}$ stored in memory by node i at time k, and introducing the vector $\mathbf{x}(k) = [x_1(k), \ldots, x_N(k)]^T \in \mathbb{R}^N$, the problem is to develop an algorithm such that

$$\lim_{k\to\infty} x_i(k) = \bar{v},\ i \in \{1, \ldots, N\} \ \equiv\ \lim_{k\to\infty} \mathbf{x}(k) = \bar{v}\mathbb{1}_N$$

and such that the update of $x_i(k)$ depends only on quantities that belong to the neighbours of node i in $\mathcal{N}^i_{\text{in}}$.
Distributed algorithms to solve the given problem already exists if the communication is synchronous, and the real challenge we want to face is represented by the fact that the communication is asynchronous and not reliable.
In this paper we assume the agents communicate with each other through a broadcast asymmetric communication protocol; however the analysis we propose in next sections can be adapted to the other communication protocols we have described in the previous Section.
Due to the communication scenario, some assumptions are needed concerning how many times a given node wakes up and how unreliable the communication is. We make the following (deterministic) assumptions

Assumption 1. (Communications are persistent). For any time instant $k \in \mathbb{N}$ there exists a positive integer $\tilde{\tau}$ such that each node performs at least one broadcast transmission within the interval $[k, k + \tilde{\tau}]$.

Assumption 2. (Packet losses are bounded). There exists a positive integer L such that the number of consecutive communication failures over every directed edge in the communication graph is smaller than L.

A direct consequence of these assumptions is that, considering any instant $k \geq 0$, in the interval $k, k+1, \ldots, k + L(\tilde{\tau}+1) - 1$ each link of the graph $\mathcal{G}$ is successfully used at least once.

4. ASYNCHRONOUS AND ROBUST CONSENSUS

The asynchronous and robust Average Consensus algorithm (*ra-AC*) we present takes inspiration from the algorithm presented in Vaidya et al. (2011) but under asynchronous communication. In particular we adopt a broadcast asymmetric communication protocol, that is we only allow one node and, in a second moment, all its out-neighbours that receive information, to update part of their variables at each iteration. In Vaidya et al. (2011), instead, all the nodes at each iteration perform some computations. Thanks to our assumptions on the communication, we are able to prove that the convergence of the algorithm is exponential.

As the algorithm in Vaidya et al. (2011), also the *ra-AC* algorithm is based on the average ratio consensus introduced in Bénézit et al. (2010). According to the ratio consensus, variable $x_i \in \mathbb{R}$ of node i that reaches consensus on the mean of the vector $\mathbf{v}$ is obtained as the ratio of two appropriate scalar quantities y_i and s_i; the update of y_i and s_i are made by node i as a linear combination of its own variable and of the companion variables of its neighbours. However, differently from Bénézit et al. (2010)), where the communications were assumed to be reliable, in our communication scenario the packets exchanged between two nodes can be lost. In this case, we

Algorithm 1 *ra-AC*(time k, node h wakes up)

```
1:  % Node $h$ updates its variables
2:  $y_h(k+1) = \frac{y_h(k)}{|\mathcal{N}^h_{\text{out}}|+1}$;
3:  $s_h(k+1) = \frac{s_h(k)}{|\mathcal{N}^h_{\text{out}}|+1}$;
4:  $x_h(k+1) = \frac{y_h(k+1)}{s_h(k+1)}$;
5:  $\sigma^{(y)}_h(k+1) = \sigma^{(y)}_h(k) + y_h(k+1)$;
6:  $\sigma^{(s)}_h(k+1) = \sigma^{(s)}_h(k) + s_h(k+1)$;
7:  % Node $h$ broadcasts variable $\sigma^{(y)}_h(k+1)$ and $\sigma^{(s)}_h(k+1)$
        to all $j \in \mathcal{N}^h_{\text{out}}$
8:  if node $j$ receives $\sigma^{(y)}_h(k+1)$ and $\sigma^{(s)}_h(k+1)$ then
9:      $y_j(k+1) = \sigma^{(y)}_h(k+1) - \rho^{(h,y)}_j(k) + y_j(k)$;
10:     $s_j(k+1) = \sigma^{(s)}_h(k+1) - \rho^{(h,s)}_j(k) + s_j(k)$;
11:     $x_j(k+1) = \frac{y_j(k+1)}{s_j(k+1)}$;
12:     $\rho^{(h,y)}_j(k+1) = \sigma^{(y)}_h(k+1)$;
13:     $\rho^{(h,s)}_j(k+1) = \sigma^{(s)}_h(k+1)$;
14: end if
15: % The variables of the other nodes are not changed
```

need to ensure that all the information sent by node i to its neighbour j is received by j at least every once in a while. The remarkable idea that allows to meet this requirement is that of introducing the use of counters: in particular node i has a counter $\sigma^{(y)}_i(k)$ ($\sigma^{(s)}_i(k)$ respectively) to keep track of the total y-mass (total s-mass) [2] sent by itself to its neighbours from time 0 to time k, while node j has a counter $\rho^{(i,y)}_j(k)$ ($\rho^{(i,s)}_j(k)$ resp.) to take into account the total y-mass (total s-mass) received from its neighbour i from time 0 to time k (one such variable for all $i \in \mathcal{N}^j_{\text{in}}$). In this way, if at time k node j receives information from node i, the information coming from node i used in the update of the variable $y_j(k)$ ($s_j(k)$ resp.) will be $\sigma^{(y)}_i(k) - \rho^{(i,y)}_j(k)$ ($\sigma^{(s)}_i(k) - \rho^{(i,s)}_j(k)$ resp.); in this way the information sent by an agent but not received due to packet losses is only delayed and not lost.
We have inherited the idea of using counters from the algorithm in Vaidya et al. (2011). Our *ra-AC* algorithm, taking inspiration from the latter ideas, carries out a ratio consensus according to an asynchronous communication protocol, and the generic k-th iteration is described in Algorithm 1. To be implemented, each node $i \in \{1, \ldots, N\}$ in the network has to keep in memory the following scalar quantities: $y_i(k), s_i(k), \sigma^{(y)}_i(k), \sigma^{(s)}_i(k)$ and $\rho^{(i,y)}_j(k), \rho^{(i,s)}_j(k), \forall (i,j) \in \mathcal{E}$, while the quantity of interest $x_i(k)$ is evaluated as $y_i(k)/s_i(k)$. We collect variables $y_i(k)$ and $s_i(k)$ resp. in the N-dimensional vectors $\mathbf{y}(k)$ and $\mathbf{s}(k)$.
Suppose that at a given iteration node h wakes up. Then, the main steps of *ra-AC* are the following: first node h updates its variables y_h and s_h dividing their previous value by the cardinality of its out-neighbours set augmented by 1 (steps 2-3). Note that this operation leaves in fact unchanged the value of variable x_h. Then it updates the

[2] As in Vaidya et al. (2011), we interchangeably use the word mass for information, since the physical idea of the transferring of mass quantities can be helpful in understanding how the algorithm works.

counters $\sigma_h^{(y)}$ and $\sigma_h^{(s)}$ (steps 5-6) and sends these updated values to its out-neighbours. Now, if node $j \in \mathcal{N}_{\text{out}}^h$ receives the packet from node h, it updates the variables y_j and s_j as described in steps 10-11, then it adjourns x_j and it finally stores in memory the new values for $\rho_j^{(h,y)}$ and $\rho_j^{(h,s)}$ (steps 11-12-13).
The following Theorem shows that, with a proper initialization of the variables, the *ra-AC* algorithm works as an average consensus algorithm, that is, the variables x_i, $i \in \{1,\dots,N\}$, which are updated distributively and iteratively, converge to the average of the N components of the vector $\mathbf{v}$.

Theorem 3. Under Assumptions 1 and 2 and under the following initialization for the variables

$$\begin{aligned}&\mathbf{y}(0) = \mathbf{v}, \qquad \mathbf{s}(0) = \mathbb{1}_N,\\ &\sigma_i^{(y)}(0) = \sigma_i^{(s)}(0) = 0, \ \forall i\{1,\dots,N\},\\ &\rho_j^{(i,y)}(0) = \rho_j^{(i,s)}(0) = 0, \ \forall (i,j) \in \mathcal{E},\end{aligned}$$

the evolution, obtained using *ra-AC* algorithm, of the variable $\mathbf{x}(k)$ exponentially converges to $\bar{v}\mathbb{1}_N$, $\bar{v} = \mathbf{v}^\top \mathbb{1}_N/N$, that is, there exist suitable constants $C > 0$, $0 < d < 1$ such that

$$\|\mathbf{x}(k) - \bar{v}\mathbb{1}_N\|^2 \le C \left(d^{\frac{1}{\tau}}\right)^k \|x(0) - \bar{v}\mathbb{1}_N\|^2, \tag{1}$$

where $\tau = NL(\tilde{\tau} + 1)$.

Since the proof of the Theorem is quite involved and consists of several steps we postpone it in the next Section.

The bound in (1) depends on our communication scenario through τ. For a fixed number of nodes N, τ might increases, either because each node wakes up less often, or because each communication link may fail for a longer period of time (or both), which implies that the dissemination of information may become more difficult. In Equation (1), if τ increases the upper bound becomes larger and larger, which is coherent with the fact that the information is spread through the network in a slower way.

Remark 4. If no packet losses occur, the variables $\sigma_i^{(y)}(k)$, $\sigma_i^{(s)}(k)$, $\rho_i^{(j,y)}(k)$ and $\rho_i^{(j,s)}(k)$ can be discarded (and variables y_i and s_i of the nodes that receive the information are updated directly using the packets they receive). The algorithm we obtain in this case is subsumed by those presented in (Bénézit et al., 2010).

5. PROOF OF CONVERGENCE

The proof of Theorem 3 is based on the theory of ergodic coefficients for positive matrices Seneta (2006), applied to the particular case of stochastic matrices. In the first part, we follow what is done in Vaidya et al. (2011). The Assumptions 1 and 2 allow us to state the results in Vaidya et al. (2011) without resorting to probability theory. However, we exploit the ergodicity theory in a way such that it allow us to conclude the exponential convergence of the algorithm. To proceed with the proof, we first rewrite the algorithm iteration in a matrix form, then we study the property of the matrices involved and we finally exploit ergodicity theory to prove the convergence of the algorithm.

Matrix form for *ra-AC* We start by introducing the indicator variables $\chi_i(k)$ and $\chi_{(i,j)}(k)$, $i,j \in \{1,\dots,N\}$. The variable $\chi_i(k)$ is equal to 1 if node i wakes up at time k, otherwise is 0; at this regard, recall that since we adopt a broadcast asymmetric protocol only one node turns on at each iteration. Concerning $\chi_{(i,j)}(k)$, the variable is 1 if node i wakes up at time k, if $j \in \mathcal{N}_{\text{out}}^i$ and if the edge $(i,j) \in \mathcal{E}$ is reliable at time k, while it is 0 otherwise. Formally

$$\chi_i(k) = \begin{cases} 1 & \text{if node } i \text{ wakes up at time } k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and

$$\chi_{(i,j)}(k) = \begin{cases} 1 & \text{if } \chi_i(k)=1, (i,j)\in\mathcal{E} \text{ active at time } k \\ 0 & \text{otherwise}^{3} \end{cases} \tag{3}$$

Observe that $\sum_{i=1}^N \chi_i(k) = 1$. Now, we only analyze the matrices which describe the evolution of variable $\mathbf{y}(k)$, since the same matrices drive the evolution of variable $\mathbf{s}(k)$. Using the indicator variables we can rewrite the update for the total sent-mass counter $\sigma_i^{(y)}(k)$ and for the total received-mass counter $\rho_j^{(i,y)}(k)$ as

$$\sigma_i^{(y)}(k+1) = \sigma_i^{(y)}(k) + \chi_i(k)\frac{y_i(k)}{|\mathcal{N}_{\text{out}}^i| + 1} \tag{4}$$

$$\begin{aligned}\rho_j^{(i,y)}(k+1) = \rho_j^{(i,y)}(k)-&\\ - \chi_i(k)\chi_{(i,j)}(k)&\left(\rho_j^{(i,y)}(k) - \sigma_i^{(y)}(k+1)\right)\end{aligned} \tag{5}$$

Let us introduce variables

$$\nu_{(i,j)}^{(y)}(k) = \sigma_i^{(y)}(k) - \rho_j^{(i,y)}(k), \ \forall (i,j) \in \mathcal{E}.$$

These variables indicate how much of the mass sent by node i is still to be received by node j. If at time $k-1$ node i turns on and the communication between node i and j (where j is a neighbour of i) is successful, then $\nu_{(i,j)}^{(y)}(k)$ is 0, otherwise it contains the information missing in node j. Using equations (4) and (5) the update of these variables can be written as

$$\begin{aligned}\nu_{(i,j)}^{(y)}(k+1) &= \left[1-\chi_i(k)\chi_{(i,j)}(k)\right]\left[\sigma_i^{(y)}(k+1)-\rho_j^{(i,y)}(k)\right]\\ &= \left[1 - \chi_i(k)\chi_{(i,j)}(k)\right]\left[\chi_i(k)\frac{y_i(k)}{|\mathcal{N}_{\text{out}}^i| + 1} + \nu_{(i,j)}^{(y)}(k)\right].\end{aligned} \tag{6}$$

We will exploit these variables to rewrite the update of vector $\mathbf{y}(k)$ in a matrix form. Note that these quantities are not actually computed by the nodes, and are just auxiliary variables used to enable the matrix version of the update.

To rewrite the update for the $y_i(k)$ variable, we have to consider three different cases:

- if $\chi_i(k) = 1$, it holds $y_i(k+1) = \frac{y_i(k)}{|\mathcal{N}_{\text{out}}^i|+1}$;
- if $\chi_j(k) = 1$ and $i \in \mathcal{N}_{\text{in}}^j$ and $\chi_{(j,i)}(k) = 1$, then
$$\begin{aligned}y_i(k+1) &= \sigma_j^{(y)}(k+1) - \rho_i^{(j,y)}(k) + y_i(k)\\ &= \nu_{(j,i)}^{(y)}(k) + \frac{y_j(k)}{|\mathcal{N}_{\text{out}}^j| + 1} + y_i(k);\end{aligned}$$
- if $\chi_j(k) = 1$ and $i \in \mathcal{N}_{\text{in}}^j$ and $\chi_{(j,i)}(k) = 0$ or if $i \notin \mathcal{N}_{\text{in}}^j$, then it holds
$$y_i(k+1) = y_i(k).$$

[3] $\chi_{(i,j)}(k)$ is considered identically 0 for all k, if $(i,j) \notin \mathcal{E}$

The above three cases are all captured by the following update

$$y_i(k+1) = \chi_i(k)\frac{y_i(k)}{|\mathcal{N}^i_{\text{out}}|+1} + [1-\chi_i(k)]\cdot$$
$$\cdot\left\{\sum_{j\neq i}\left[\chi_j(k)\chi_{(j,i)}(k)\left(\nu^{(y)}_{(j,i)}(k)+\frac{y_j(k)}{|\mathcal{N}^j_{\text{out}}|+1}\right)\right]+y_i(k)\right\}. \quad (7)$$

Now let us introduce the column vector $\boldsymbol{\nu}^{(y)}(k) = [\nu^{(y)}_{(i,j)}(k)] \in \mathbb{R}^E$, which collects all different $\nu^{(y)}_{(i,j)}(k)$. Moreover let us define the row vector

$$\boldsymbol{\phi}^{(y)}(k) = [\mathbf{y}(k)^\top \ \boldsymbol{\nu}^{(y)}(k)^\top] \in \mathbb{R}^{N+E}.$$

Our aim is to find matrix $M(k) \in \mathbb{R}^{(N+E)\times(N+E)}$ according to which it holds

$$\boldsymbol{\phi}^{(y)}(k+1) = \boldsymbol{\phi}^{(y)}(k)M(k). \quad (8)$$

Let us first consider the $i-$th row of matrix $M(k)$, with $i \in \{1,\dots,N\}$. The element $[M(k)]_{ii}$ indicates how $y_i(k)$ influences $y_i(k+1)$, so

$$[M(k)]_{ii} = \frac{\chi_i(k)}{|\mathcal{N}^i_{\text{out}}|+1} + [1-\chi_i(k)].$$

The element $[M(k)]_{ij},\ j \in \{1,\dots,N\}\setminus\{i\}$ indicates how $y_i(k)$ influences $y_j(k+1)$. It holds

$$[M(k)]_{ij} = [1-\chi_j(k)]\left[\frac{\chi_i(k)\chi_{(i,j)}(k)}{|\mathcal{N}^i_{\text{out}}|+1}\right].$$

Finally, if $\ell \in \{N+1,\dots,N+E\}$ is such that $[\boldsymbol{\phi}^{(y)}(k)]_\ell = \nu^{(y)}_{(r,j)}(k)$, the element $[M(k)]_{i\ell}$ indicates how $y_i(k)$ influences $\nu^{(y)}_{(r,j)}(k)$. It holds

$$[M(k)]_{i\ell} = \begin{cases} \left[1-\chi_i(k)\chi_{(i,j)}(k)\right]\left[\frac{\chi_i(k)}{|\mathcal{N}^i_{\text{out}}|+1}\right] & \text{if } r=i \\ 0 & \text{if } r\neq i \end{cases}$$

We now analyse the $h-$th row of $M(k)$, with $h \in \{N+1,\dots,N+E\}$. Let us suppose that $[\boldsymbol{\phi}^{(y)}(k)]_h = \nu^{(y)}_{(r,\ell)}(k)$. Reasoning as before, we have

$$[M(k)]_{hh} = 1-\chi_r(k)\chi_{(r,\ell)}(k),$$
$$[M(k)]_{h\ell} = [1-\chi_\ell(k)]\left[\chi_r(k)\chi_{(r,\ell)}(k)\right].$$

and all the other elements in the $h-$th row are 0.

Using the matrices $M(k)$ just defined and introducing variables $\nu^{(s)}_{(i,j)}(k) = \sigma^{(s)}_i(k) - \rho^{(i,s)}_j(k),\ \forall (i,j) \in \mathcal{E}$ and $\boldsymbol{\phi}^{(s)}(k) = [\mathbf{s}(k)^\top \ \boldsymbol{\nu}^{(s)}(k)^\top]$, the evolution of $\boldsymbol{\phi}^{(y)}(k)$ and $\boldsymbol{\phi}^{(s)}(k)$ is given by

$$\begin{cases} \boldsymbol{\phi}^{(y)}(k+1) = \boldsymbol{\phi}^{(y)}(k)M(k) \\ \boldsymbol{\phi}^{(s)}(k+1) = \boldsymbol{\phi}^{(s)}(k)M(k) \end{cases} \quad (9)$$

We recall that the first N elements of vectors $\boldsymbol{\phi}^{(y)}(k)$ and $\boldsymbol{\phi}^{(s)}(k)$ corresponds respectively to $\mathbf{y}(k)$ and $\mathbf{s}(k)$.

Properties of matrices $M(k)$**.** Introducing the set $\mathcal{M}$, which collects all possible matrices $M(k)$, the following lemma holds true.

Lemma 5. The set of matrices $\mathcal{M}$ satisfies

(1) $\mathcal{M}$ is a finite set;
(2) each $M \in \mathcal{M}$ is a row-stochastic matrix;
(3) each positive element in any matrix $M \in \mathcal{M}$ is lower bounded by a positive constant c;
(4) given $\tau = NL(\tilde{\tau}+1)$, for all $k \geq 0$, the stochastic matrix
$V^{(\tau)}(k) = M(k)M(k+1)\cdots M(k+\tau-1),\ M(t) \in \mathcal{M}$,
is such that its first N columns have all the elements which are strictly positive.

Proof. (1) Each matrix $M \in \mathcal{M}$ depends on which node wakes up and on which communication links from this node to its neighbours work. Since the number of all possible combinations is finite (and in particular equal to $\sum_{i=1}^N 2^{|\mathcal{N}^i_{\text{out}}|}$) the property is verified.
(2) Consider first the $i-$th row of M, with $i \in \{1,\dots,N\}$. Then, either $\chi_i(k) = 0$, from which it follows

$$\begin{cases} [M(k)]_{ii} = 1 \\ [M(k)]_{ij} = 0 \text{ if } j \in \{1,\dots,N\}\setminus\{i\} \\ [M(k)]_{ij} = 0 \text{ if } j \in \{N+1,\dots,N+E\} \end{cases},$$

or $\chi_i(k) = 1$ (and all other $\chi_j(k) = 0, j \neq i$), implying

$$\begin{cases} [M(k)]_{ii} = \frac{1}{|\mathcal{N}^i_{\text{out}}|+1} \\ [M(k)]_{ij} = \frac{\chi_{(i,j)}(k)}{|\mathcal{N}^i_{\text{out}}|+1} \qquad \text{if } j \in \{1,\dots,N\}\setminus\{i\} \end{cases}$$

and

$$[M(k)]_{i\ell} = \frac{1-\chi_{(i,j)}(k)}{|\mathcal{N}^i_{\text{out}}|+1}$$

for those $\ell \in \{N+1,\dots,N+E\}$ for which there exists a $j \in \{1,\dots,N\}\setminus\{i\}$ such that $\psi_\ell(k) = \nu^{(y)}_{(i,j)}(k)$ and $[M(k)]_{i\ell} = 0$ otherwise. Note that in both cases the sum of the row is 1.

Consider now the $h-$th row of matrix $M(k)$, with h such that $[\boldsymbol{\phi}^{(y)}(k)]_h = \nu^{(y)}_{(r,\ell)}(k)$. If $\chi_r(k) = 0$ it holds

$$\begin{cases} [M(k)]_{hh} = 1 \\ [M(k)]_{h\ell} = 0 \text{ if } \ell \in \{1,\dots,N+E\}\setminus\{i\} \end{cases}.$$

On the other hand, if $\chi_r(k) = 1$

$$\begin{cases} [M(k)]_{hh} = 1-\chi_{(r,\ell)}(k) \\ [M(k)]_{h\ell} = \chi_{(r,\ell)}(k) \\ [M(k)]_{hj} = 0 \text{ if } j \in \{1,\dots,N+E\}\setminus\{i,\ell\} \end{cases}$$

In both cases the row sums up to 1.
(3) This directly follows from the construction of $M(k)$.
(4) Let us define $V^{(h)}(k) = M(k)M(k+1)\dots M(k+h-1), k \geq 0, h \geq 1, V^{(0)}(k) = I_N, k \geq 0$, which can be divided as

$$V^{(h)}(k) = \begin{bmatrix} A^{(h)}_{11}(k) & A^{(h)}_{12}(k) \\ A^{(h)}_{21}(k) & A^{(h)}_{22}(k) \end{bmatrix},$$

with $A^{(h)}_{11}(k) \in \mathbb{R}^{N\times N}, A^{(h)}_{22}(k) \in \mathbb{R}^{E\times E},\ A^{(h)}_{12}(k) \in \mathbb{R}^{N\times E}$ and $A^{(h)}_{21}(k) \in \mathbb{R}^{E\times N}$. Since every matrix $M \in \mathcal{M}$ is such that $[M]_{ii} > 0$ if $i \in \{1,\dots,N\}$, it holds, for $h \geq 1$, that if in the product which yields $V^{(h)}(k)$ there exists a matrix with the element in position (i,j) strictly greater than 0, then also $[V^{(h)}(k)]_{ij} > 0$. Due to Assumptions 1 and 2, after $L(\tilde{\tau}+1)$ iterations, all the links in graph $\mathcal{G}$ have successfully transmitted at least once. Moreover, if at time $k+\Delta,\ 0 \leq \Delta \leq L\tilde{\tau}$, the communication link of the edge $(i,j) \in \mathcal{E}$ is reliable, then considering index s such that $[\boldsymbol{\phi}^{(y)}(\cdot)]_s = \nu^{(y)}_{(i,j)}(\cdot)$, it holds $[M(k+\Delta)]_{sj} > 0$. As a consequence, each row of $A^{(L\tilde{\tau})}_{21}(k)$ has at least one non zero element. Using a similar reasoning, for all $(i,j) \in \mathcal{E}$, it holds that $[V^{(L\tilde{\tau})}(k)]_{ij} > 0$. Since graph $\mathcal{G}$ is connected, it holds that all the elements of $A_{11}^{((N-1)L(\tilde{\tau}+1))}(k)$ are strictly positive. Due to the last two properties, choosing $\tau = NL(\tilde{\tau}+1)$, matrix $V^{(\tau)}(k)$ has the first N columns with all the elements strictly positive. $\square$

Remark 6. The constant τ has been evaluated in the worst possible scenario. As a matter of fact we assumed that in graph $\mathcal{G}$ there are at least two nodes that communicate with each other in no less than $N-1$ steps and we also assumed that the communication along one link fails $L-1$ times consecutively. This implies that in a random network $\mathcal{G}$, where the diameter of the graph is usually much smaller than the number of nodes, the actual constant τ according to which the first N columns of $V^{(\tau)}(k)$ are strictly positive will be, in general, much smaller.

Ergodicity theory and convergence of *ra-AC* We will briefly recall some useful concepts of ergodicity theory to be later applied to prove the convergence of the algorithm. An exhaustive explanation for ergodicity theory can be found in Seneta (2006).
Given a stochastic matrix $P \in \mathbb{R}^{N\times N}$, a coefficient of ergodicity for P quantifies how much its rows are different from each other . Two well-known coefficients of ergodicity for a stochastic matrix P are

$$\delta(P) := \max_j \max_{i_1,i_2} \left|[P]_{i_1 j} - [P]_{i_2 j}\right|,$$
$$\lambda(P) := 1 - \min_{i_1,i_2} \sum_j \min\left\{[P]_{i_1 j}, [P]_{i_2 j}\right\}.$$

These coefficients are proper (that is $\delta(P) = 0$ and $\lambda(P) = 0$ if and only if $P = \mathbb{1}_n \mathbf{w}^\top$, with $\mathbf{w}$ such that $\mathbf{w}^\top \mathbb{1}_n = 1$), and, as all the coefficients of ergodicity, $0 \le \delta(P) \le 1$ and $0 \le \lambda(P) \le 1$.
Consider now a stochastic matrix P such that $\delta(P) < \psi$. Selecting two elements in any column of P, the difference between these two elements is necessarily smaller than ψ. Consider now a vector $\mathbf{y} \in \mathbb{R}^N$ which sums to 0, that is $\mathbb{1}_N{}^\top \mathbf{y} = 0$. Let us define the related quantities

$$y_{\text{pos}} = \sum_{i|y_i>0} y_i \ge 1, \quad y_{\text{neg}} = \sum_{i|y_i<0} y_i \le 0, \quad y_{\text{pos}} + y_{\text{neg}} = 0,$$

and also, for any $j \in \{1,\dots,N\}$, the quantities

$$\overline{P}_j = \max_h [P]_{hj}, \quad \underline{P}_j = \min_h [P]_{hj}, \quad \overline{P}_j - \psi \le \underline{P}_j \le \overline{P}_j.$$

Suppose now that[4] $y_{\text{pos}} > 0$. We aim at finding an upper and lower bound for $[\mathbf{y}^\top P]_j$, for any $1 \le j \le N$. The maximum value for this product is achieved in case the positive elements of vector $\mathbf{y}$ are multiplied by $\overline{P}_j$ and the negative elements of the same vector are multiplied by $\underline{P}_j$, that is

$$[\mathbf{y}^\top P]_j \le y_{\text{pos}} \overline{P}_j + y_{\text{neg}} \underline{P}_j \le y_{\text{pos}} \overline{P}_j + y_{\text{neg}} \overline{P}_j - y_{\text{neg}} \psi$$

which reduces to $[\mathbf{y}^\top P]_j \le -y_{\text{neg}}\psi$. The minimum value of $[\mathbf{y}^\top P]_j$ is instead produced if the negative elements are multiplied by $\overline{P}_j$ and the positive ones by $\underline{P}_j$, i.e.

$$[\mathbf{y}^\top P]_j \ge y_{\text{neg}} \overline{P}_j + y_{\text{pos}} \underline{P}_j \le y_{\text{neg}} \overline{P}_j + y_{\text{pos}} \overline{P}_j - y_{\text{pos}} \psi$$

which implies $[\mathbf{y}^\top P]_j \ge -y_{\text{pos}}\psi$. From these two bounds we obtain

$$\left|[\mathbf{y}^\top P]_j\right| \le \psi \sum_{i=1}^N |y_i| \tag{10}$$

This bound will be used to prove the convergence of the algorithm. In particular, the stochastic matrix involved will be the forward product of the matrices that define the evolution of the algorithm as seen in (9), that is matrix $T(k) = M(0)M(1)\cdots M(k)$. This matrix allows to evaluate $\boldsymbol{\phi}^{(y)}(k+1)$ given $\boldsymbol{\phi}^{(y)}(0)$ (supposing the initial time is 0). In order to evaluate the coefficient $\delta(T(k))$, we will exploit an important property that holds for $\delta(\cdot)$ and $\lambda(\cdot)$ when we are dealing with the product of row stochastic matrices: given h stochastic matrices $P_1,\dots,P_h$, then

$$\delta(P_1 P_2 \cdots P_h) \le \prod_{i=1}^h \lambda(P_i). \tag{11}$$

As a consequence if some of the matrices P_i are such that $\lambda(P_i) < 1$, then also $\delta(P_1P_2\cdots P_h)$ will be strictly less than 1. A stochastic matrix P such that $\lambda(P) < 1$ is called scrambling, and a sufficient condition for P to be scrambling is that at least one column is strictly positive,as can be verified by the definition of $\lambda(\cdot)$.
Let us apply this theory to our forward product of matrices, starting by defining the following

$$W(h) = \prod_{k=(h-1)\tau}^{h\tau-1} M(k), \;\; h \ge 1, \;\; M(k) \in \mathcal{M}$$

which, by Lemma 5, have strictly positive columns. As a consequence, $\lambda(W(h)) < 1$ for all $h \ge 1$. Moreover, the number of different $W(h)$ is finite since matrices $M(k)$ are finite and the Assumptions 1 and 2 have to be satisfied. Collecting all $W(h)$ in set $\mathcal{W}$, it is possible to define value

$$d = \max_{W\in\mathcal{W}} \lambda(W),$$

which is strictly smaller than 1.
The following lemma holds

Lemma 7. The constant $\beta = d^{1/(2\tau)}$, $0 < \beta < 1$, is such that $\delta(T(k)) \le \beta^k$ for $k \ge \tau$.

Proof. If $k \ge \tau$, $T(k)$ can be rewritten as

$$T(k) = W(1)\cdots W(h) M(h\tau) M(h\tau+1)\cdots M(h\tau+\Delta)$$

with $h = \lfloor k/\tau \rfloor$ and $\Delta = k - h\tau$, $0 \le \Delta \le \tau - 1$. As a consequence, using Formula (11)

$$\delta(T(k)) \le \lambda(W(1))\cdots\lambda(W(h))\lambda\left(\prod_{j=0}^{\Delta} M(h\tau+j)\right) \le d^h.$$

Since $h \ge k/(2\tau)$, $d^h \le d^{k/(2\tau)}$, so choosing $\beta = d^{1/(2\tau)}$, $\delta(T(k)) \le \beta^k$. $\square$

Lemma 7 implies that the coefficient of ergodicity for $T(k)$ converges to 0 as k goes to infinity.
Before showing the convergence of *ra-AC*, we need to show that each component of $\mathbf{s}(k)$ is lower bounded by a constant $\mu > 0$, since the variable $\mathbf{x}(k)$ is obtained through the Hadamard division by $\mathbf{s}(k)$. Note that if $k \ge \tau$, the elements of the first N columns of $T(k)$ are strictly bigger than c^τ. As a consequence, $s_i(k+1) \ge c^\tau \sum_{j=1}^N s_j(0) \ge c^\tau$. On the other hand, since the first N elements of the diagonals of matrices $M(k)$ are strictly positive, if $1 \le k \le \tau - 1$, $s_i(k+1) \ge c^k s_i(0) \ge c^\tau$, since $0 < c < 1$ and

[4] It is possible to verify that the property we are interested in, that is the bound in Equation 10, is still verified if $y_i = 0 \; \forall i$.

$\mathbf{s}(0) = \mathbb{1}_N$. Therefore we take $\mu = c^\tau$.
Let us finally prove convergence: we will first prove the exponential convergence in case vector $\mathbf{v}$ is zero mean, $\bar{v} = 0$, and then we will generalize to the case, $\bar{v} \neq 0$. For $k \geq \tau$, by Lemma 7 we have $\delta(T(k)) \leq \beta^k$, where $T(k)$ is such that $\boldsymbol{\phi}^{(y)}(k+1) = \boldsymbol{\phi}^{(y)}(0)T(k)$. Starting from Formula (10), and remembering that the first N elements of $\boldsymbol{\phi}^{(y)}(0)$ are $\mathbf{y}(0)$ and the other elements are 0, for all $1 \leq i \leq N$ it holds

$$\left|[\boldsymbol{\phi}^{(y)}(k+1)]_i\right| = \left|[\boldsymbol{\phi}^{(y)}(0)T(k+1)]_i\right| \leq \beta^k \sum_i |y_i|$$

Now, since for $1 \leq i \leq N$, $y_i(k+1) = [\boldsymbol{\phi}^{(y)}(k+1)]_i$ and the elements of $\mathbf{s}(k)$ are strictly greater than μ, we have

$$|y_i(k+1)| = \left|\frac{s_i(k+1)}{s_i(k+1)} y_i(k+1)\right| \leq \beta^k \sum_i |y_i|$$

which implies

$$|x_i(k+1)| = \left|\frac{y_i(k+1)}{s_i(k+1)}\right| \leq \frac{1}{s_i(k)} \beta^k \sum_i |y_i| \leq \frac{1}{\mu} \beta^k \sum_i |y_i|$$

and then

$$\|\mathbf{x}(k+1)\|^2 \leq \frac{N}{\mu^2 \beta^2} (\beta^2)^{k+1} \left(\sum_i |y_i|\right)^2 \leq \frac{N^2}{\mu^2 \beta^2} (\beta^2)^{k+1} \|\mathbf{x}(0)\|^2$$

where the last inequality is a consequence of the Cauchy-Schwarz inequality and the fact that $\mathbf{x}(0) = \mathbf{y}(0)$.
Defining $C_\infty := \frac{N^2}{\mu^2 \beta^2}$ the latter becomes

$$\|\mathbf{x}(k)\|^2 \leq C_\infty (d^{1/\tau})^k \|\mathbf{x}(0)\|^2, \; k \geq \tau + 1.$$

If $0 \leq k \leq \tau, T(k)$ is still stochastic and it surely holds that $\delta(T(k)) \leq 1$, so applying a similar reasoning we have

$$\|\mathbf{x}(k)\|^2 \leq C_k (d^{1/\tau})^k \|\mathbf{x}(0)\|^2, \; C_k = \frac{N^2}{\mu^2 (d^{1/\tau})^k}.$$

Introducing $C = \max\{C_1, \ldots, C_\tau, C_\infty\} = C_\tau$ we have

$$\|\mathbf{x}(k)\|^2 \leq C (d^{1/\tau})^k \|\mathbf{x}(0)\|^2, \; k \geq 0, \qquad (12)$$

that is we have the exponential convergence of the algorithm when vector $\mathbf{v}$ is 0, since the mean $\bar{v}$ is 0, and the vector $\mathbf{x}(k)$ is converging to $0\mathbb{1}_N$.
Let us finally generalize to the case in which $\mathbf{v}$ is such that $\bar{v} \neq 0$. Introducing vector $\mathbf{v}_0 = \mathbf{v} - \bar{v}\mathbb{1}_N$, we consider two evolutions of the algorithm, one initialized using $\mathbf{v}_0$ and the other initialized to $\mathbf{v}$. At each time step k the same matrix $M(k)$ is applied for both initializations. We use the subscript 0 to indicate the variables of the evolution starting from the zero-mean vector $\mathbf{v}_0$ and the subscript $\bar{v}$ to indicate those starting from vector $\mathbf{v}$ (vectors $\mathbf{s}(k)$ and $\boldsymbol{\phi}^{(s)}(k)$ do not have a subscript since they are the same in both evolutions). Remembering that $\mathbf{s}(0) = \mathbb{1}_N$, we have

$$\mathbf{x}_0(0) = \frac{\mathbf{y}_0(0)}{\mathbf{s}(0)}, \; \mathbf{x}_{\bar{v}} = \frac{\mathbf{y}_{\bar{v}}(0)}{\mathbf{s}(0)} = \frac{\mathbf{y}_0(0) + \bar{v}\mathbb{1}_N}{\mathbf{s}(0)} = \frac{\mathbf{y}_0(0)}{\mathbf{s}(0)} + \bar{v}\mathbb{1}_N$$

so $\mathbf{x}_{\bar{v}}(0) = \mathbf{x}_0(0) + \bar{v}\mathbb{1}_N$. Moreover, it is possible to verify that $\boldsymbol{\phi}_{\bar{v}}^{(y)}(0) = \boldsymbol{\phi}_0^{(y)}(0) + \bar{v}\boldsymbol{\phi}^{(s)}(0)$, according to which we have for all $1 \leq i \leq N$

$$[\mathbf{x}_0(k)]_i = \left[\frac{\boldsymbol{\phi}_0^{(y)}(k)}{\boldsymbol{\phi}^{(s)}(k)}\right]_i = \left[\frac{\boldsymbol{\phi}_0^{(y)}(0)T(k-1)}{\boldsymbol{\phi}^{(s)}(0)T(k-1)}\right]_i$$

$$\begin{aligned} [\mathbf{x}_{\bar{v}}(k)]_i &= \left[\frac{\boldsymbol{\phi}_{\bar{v}}^{(y)}(k)}{\boldsymbol{\phi}^{(s)}(k)}\right]_i = \left[\frac{\boldsymbol{\phi}_{\bar{v}}^{(y)}(0)T(k-1)}{\boldsymbol{\phi}^{(s)}(0)T(k-1)}\right]_i \\ &= \left[\frac{\boldsymbol{\phi}_0^{(y)}(0)T(k-1)}{\boldsymbol{\phi}^{(s)}(0)T(k-1)}\right]_i + \left[\bar{v}\frac{\boldsymbol{\phi}^{(s)}(0)T(k-1)}{\boldsymbol{\phi}^{(s)}(0)T(k-1)}\right]_i \\ &= [\mathbf{x}_0(k)]_i + \bar{v}. \end{aligned}$$

We have just proved that $\mathbf{x}_{\bar{v}}(k)$ can be always obtained as $\mathbf{x}_0(k) + \bar{v}\mathbb{1}_N$ for all $k \geq 0$, and therefore, since for $\mathbf{x}_0(k)$ Equation (12) holds, we also have

$$\|\mathbf{x}_{\bar{v}}(k) - \bar{v}\mathbb{1}_N\|^2 \leq C(d^{1/\tau})^k \|\mathbf{x}_{\bar{v}}(0) - \bar{v}\mathbb{1}_N\|, \; k \geq 0,$$

that is the exponential convergence of $\mathbf{x}$ to $\bar{v}\mathbb{1}_N$ holds for any vector $\mathbf{v} \in \mathbb{R}^N$.

Remark 8. It is possible to adequately modify the algorithm in order to work also in the other asynchronous communication scenarios introduced in Section 2. If some deterministic assumptions equivalent to the ones in this paper hold, the proof of convergence of the modified algorithm can be obtained by the one just described, introducing appropriate modifications in the constructions of matrices $M(k)$ and showing that the properties in Lemma 5 are still verified.

Remark 9. The idea of using a consensus algorithm with an augmented state in order to prove the convergence of this particular ratio consensus is taken from Vaidya et al. (2011). However, in the latter the communication is synchronous, that is at each iteration all the nodes perform some updates, and moreover the results concerning the convergence are given in probability. In the set-up illustrated in this paper, the algorithm is asynchronous and the convergence result is stated considering a worst-case scenario. This is a consequence of the two assumptions we made, which, remarkably, also allow us to prove that the convergence is exponential.

6. SIMULATIONS

In this section we show the results of some simulations done for *ra-AC*. The set-up of the simulations is the following: the number of agents considered is $N = 50$, the underlying communication graph is random geometric, with the agents arranged in a squared environment of edge equal to 1 and with maximum distance between neighbouring nodes equal to r. In addition, in order to work on directed graphs, some of the links have been forced to be unidirectional. The value of $\tilde{\tau}$ and L for Assumptions 1 and 2 are respectively 75 and 10. In particular, we consider a probability $0 < p < 1$ of losing a given packet, but if the link that is selected has failed to transmit for $L-1$ previous consecutive times, then the link is forced to be reliable without considering the packet loss probability. In Table 1 we give the averaged root mean squared error (ARMSE) of the results. In particular, for each value of d and p selected, we run $M = 500$ Monte Carlo runs (MCR) for different graph realizations. Denoting with $\mathbf{x}_{\{i\}}(k)$ the value $\mathbf{x}(k)$ obtained in the $i-$th MCR, then

$$\text{ARMSE}(k) = \frac{1}{M} \sum_{i=1}^{M} \left[\frac{1}{\sqrt{N}} \|\mathbf{x}_{\{i\}}(k) - \bar{v}\mathbb{1}_N\|_2\right]$$

ARMSE(2000)	$p = 20\%$	$p = 50\%$	$p = 80\%$
$r = 0.25$	0.395	0.635	1.22
$r = 0.33$	0.033	0.131	0.45
$r = 0.5$	$1.62\,10^{-5}$	$9.17\,10^{-4}$	0.0319

Table 1. Values of $ARMSE$ at time $k = 2000$, computed over $M = 500$ Monte Carlo runs for different values of r and p.

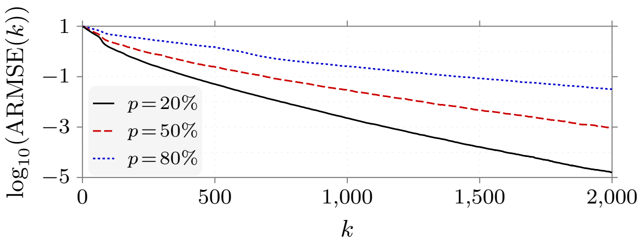

Fig. 2. ARMSE as a function of time for 3 different values for the packet loss probability, evaluated over $M = 500$ MCR. The value for the maximum distance r between nodes is $1/3$.

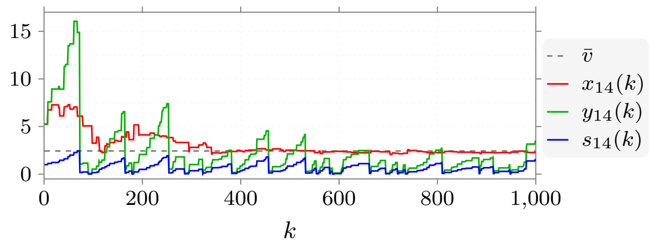

Fig. 3. Time evolution for the scalar variables $x_{14}(k)$, $y_{14}(k)$ and $s_{14}(k)$ for one run of the algorithm. In this simulation $r = 1/3$ and $p = 20\%$.

The results of the simulations show that the more connected the graph is, the faster the convergence is. On the other hand, the packet loss probability, as expected, makes the convergence slower. Note that for $r = 0.25$, even at iteration 2000 the convergence is still not good. However, even in the best case, at iteration 2000 all the nodes have woken up at most 40 times, and so, due to the presence of packet losses and the fact that each node have only few neighbours, this is not surprising.
Figure 2 shows the time evolution for the ARMSE(k), in case $r = 1/3$. For all the different values of the packet loss probability it is possible to appreciate the exponential convergence of the algorithm. Finally, Figure 3 shows the time evolution of the variables of a single node in the network. It is interesting to see that, while the ratio between $y_{14}(k)$ and $s_{14}(k)$ exponentially converges to the mean $\bar{v}$, the single variables do not converge but keep oscillating. This behaviour is typical in the ratio consensus algorithm in presence of unidirectional links and packet losses.

7. CONCLUSIONS

In this paper we presented an algorithm which allows each node in the network to reach consensus on the mean of some private constants which belong to each agent. We gave the proof for the exponential convergence of the algorithm under mild conditions and we carried out some simulations to further verify its performance.
As future research, we want to apply *ra-AC* as the fundamental block for consensus in the Newton-Raphson algorithm presented in Varagnolo et al. (2016). Newton-Raphson is an algorithm that allows to distributively minimize the sum of convex cost functions and one of the steps of its iterations is a consensus. Introducing *ra-AC* in its implementation would make Newton-Raphson an asynchronous and robust algorithm, where the latter features are important in real applications.

REFERENCES

Bénézit, F., Blondel, V., Thiran, P., Tsitsiklis, J., and Vetterli, M. (2010). Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *Proceedings ISIT" 10*, EPFL-CONF-148711. IEEE.

Bolognani, S., Favero, S.D., Schenato, L., and Varagnolo, D. (2010). Consensus-based distributed sensor calibration and least-square parameter identification in wsns. *International Journal of Robust and Nonlinear Control*, 20(2), 176–193.

Boyd, S., Diaconis, P., and Xiao, L. (2004). Fastest mixing markov chain on a graph. *SIAM review*, 46(4), 667–689.

Cattivelli, F.S. and Sayed, A.H. (2010). Diffusion strategies for distributed Kalman filtering and smoothing. *IEEE Transactions on automatic control*, 55(9), 2069–2084.

Chen, Y., Tron, R., Terzis, A., and Vidal, R. (2010). Corrective consensus: Converging to the exact average. In *49th IEEE Conference on Decision and Control (CDC)*, 1221–1228. IEEE.

Dominguez-Garcia, A.D. and Hadjicostis, C.N. (2010). Coordination and control of distributed energy resources for provision of ancillary services. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, 537–542. IEEE.

Domínguez-García, A.D. and Hadjicostis, C.N. (2011). Distributed strategies for average consensus in directed graphs. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2124–2129. IEEE.

Dominguez-Garcia, A.D., Hadjicostis, C.N., and Vaidya, N.H. (2011). Distributed algorithms for consensus and coordination in the presence of packet-dropping communication links-part I: Statistical moments analysis approach. *arXiv preprint arXiv:1109.6391*.

Garin, F. and Schenato, L. (2010). A survey on distributed estimation and control applications using linear consensus algorithms. In *Networked Control Systems*, 75–107. Springer.

Olshevsky, A. and Tsitsiklis, J.N. (2009). Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1), 33–55.

Oreshkin, B.N., Coates, M.J., and Rabbat, M.G. (2010). Optimization and analysis of distributed averaging with short node memory. *Signal Processing, IEEE Transactions on*, 58(5), 2850–2865.

Patterson, S., Bamieh, B., and El Abbadi, A. (2007). Distributed average consensus with stochastic communication failures. In *Decision and Control, 2007 46th IEEE Conference on*, 4215–4220. IEEE.

Seneta, E. (2006). *Non-negative matrices and Markov chains*. Springer Science & Business Media.

Vaidya, N.H., Hadjicostis, C.N., and Dominguez-Garcia, A.D. (2011). Distributed algorithms for consensus and coordination in the presence of packet-dropping communication links-part II: Coefficients of ergodicity analysis approach. *arXiv preprint arXiv:1109.6392*.

Varagnolo, D., Zanella, F., Cenedese, A., Pillonetto, G., and Schenato, L. (2016). Newton-Raphson consensus for distributed convex optimization. *IEEE Transactions on Automatic Control*, 61(4), 994–1009.

Xiao, L., Boyd, S., and Lall, S. (2005). A scheme for robust distributed sensor fusion based on average consensus. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, 63–70. IEEE.