Cloud-aided collaborative estimation by ADMM-RLS algorithms for connected vehicle prognostics

Technical Report TR-2017-01

Valentina Breschi, Ilya Kolmanovsky, Alberto Bemporad

September 22, 2017

Abstract

As the connectivity of consumer devices is rapidly growing and cloud computing technologies are becoming more widespread, cloud-aided techniques for parameter estimation can be designed to exploit the theoretically unlimited storage memory and computational power of the "cloud", while relying on information provided by multiple sources.
With the ultimate goal of developing monitoring and diagnostic strategies, this report focuses on the design of a Recursive Least-Squares (RLS) based estimator for identification over a group of devices connected to the "cloud". The proposed approach, that relies on Node-to-Cloud-to-Node (N2C2N) transmissions, is designed so that: (*i*) estimates of the unknown parameters are computed locally and (*ii*) the local estimates are refined on the cloud. The proposed approach requires minimal changes to local (pre-existing) RLS estimators.

1 Introduction

With the increasing connectivity between devices, the interest in distributed solutions for estimation [24], control [10] and machine learning [9] has been rapidly growing. In particular, the problem of parameter estimation over networks has been extensively studied, especially in the context of Wireless Sensor Networks (WSNs). The methods designed to solve this identification problem can be divided into three groups: incremental approaches [18], diffusion approaches [5] and consensus-based distributed strategies [20]. Due to the low communication power of the nodes in WSNs, research has mainly been devoted to obtain fully distributed approaches, i.e. methods that allow exchanges of information between neighbor nodes only. Even though such a choice enables to reduce multi-hop transmissions and improve robustness to node failures, these strategies allows only neighbor nodes to communicate and thus to reach consensus.

*Valentina Breschi and Alberto Bemporad are with the IMT School for Advanced Studies Lucca, Piazza San Francesco 19, 55100 Lucca, Italy. valentina.breschi@imtlucca.it; alberto.bemporad@imtlucca.it

†Ilya Kolmanovsky is with the Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA. ilya@umich.edu

‡The results in the report have been partially presented in a paper submitted to ACC 2018.

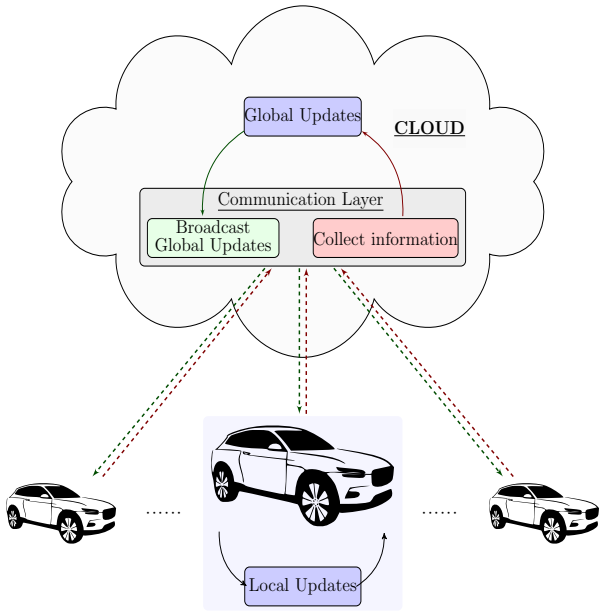


Figure 1: Cloud-connected vehicles.

As a consequence, to attain consensus on the overall network, its topology has to be chosen to enable exchanges of information between the different groups of neighbor nodes.

At the same time, with recent advances in cloud computing [22] it has now become possible to acquire and release resources with minimum effort so that each node can have on-demand access to shared resources, theoretically characterized by unlimited storage space and computational power. This motivates to reconsider the approach towards a more centralized strategy where some computations are performed at the node level, while the most time and memory consuming ones are executed "on the cloud". This requires the communication between the nodes and a fusion center, i.e. the "cloud", where the data gathered from the nodes are properly merged.

Cloud computing has been considered for automotive vehicle applications in [13]-[14] and [25]. As motivating example for another possible automotive application, consider a vehicle fleet with vehicles connected to the "cloud" (see Figure 1). In such a setting, measurements taken on-board of the vehicles can be used for cloud-based diagnostics and prognostics purposes. In particular, the measurements can be used to estimate parameters that may be common to all vehicles, such as parameters in components wear models or fuel consumption models, and parameters that may be specific to individual vehicles. References [32] and [12] suggest potential applications of such approaches for prognostics of automotive fuel pumps and brake pads. Specifically, the component wear rate as a function of the workload (cumulative fuel flow or energy dissipated in the brakes) can be common to all vehicles or at least to all vehicles in the same class.

A related distributed diagnostic technique has been proposed in [3]. However it relies on a fully-distributed scheme, introduced to reduce long distance transmissions and to avoid the presence of a "critic" node in the network, i.e. a node

whose failure causes the entire diagnostic strategy to fail.

In this report a centralized approach for recursive estimation of parameters in the least-squares sense is presented. The method has been designed under the hypothesis of (i) ideal transmission, i.e. the information exchanged between the cloud and the nodes is not corrupted by noise, and the assumption that (ii) all the nodes are described by the same model, which is supposed to be known a priori. Differently from what is done in many distributed estimation methods (e.g. see [20]), where the nodes estimate common unknown parameters, the strategy we propose allows to account for more general consensus constraint. As a consequence, for example, the method can be applied to problems where only a subset of the unknowns is common to all the nodes, while other parameters are purely local, i.e. they are different for each node.
Our estimation approach is based on defining a separable optimization problem which is then solved through the Alternating Direction Method of Multipliers (ADMM), similarly to what has been done in [20] but in a somewhat different setting. As shown in [20], the use of ADMM leads to the introduction of two time scales based on which the computations have to be performed. In particular, the local time scale is determined by the nodes' clocks, while the cloud time scale depends on the characteristics of the resources available in the center of fusion and on the selected stopping criteria, used to terminate the ADMM iterations.
The estimation problem is thus solved through a two-step strategy. In particular: (i) local estimates are recursively retrieved by each node using the measurements acquired from the sensors available locally; (ii) global computations are performed to refine the local estimates, which are supposed to be transmitted to the cloud by each node. Note that, based on the aforementioned characteristics, back and forth transmissions to the cloud are required. A transmission scheme referred to as Node-to-Cloud-to-Node (N2C2N) is thus employed.
The main features of the proposed strategies are: (i) the use of recursive formulas to update the local estimates of the unknown parameters; (ii) the possibility to account for the presence of both purely local and global parameters, that can be estimated in parallel; (iii) the straightforward integration of the proposed techniques with pre-existing Recursive Least-Squares (RLS) estimators already running on board of the nodes.

The report is organized as follows. In Section 2 ADMM is introduced, while in Section 3 is devoted to the statement of the considered problem. The approach for collaborative estimation with full consensus is presented in Section 4, along with the results of simulation examples that show the effectiveness of the approach and its performance in different scenarios. In Section 5 and Section 6 the methods for collaborative estimation with partial consensus and for constrained collaborative estimation with partial consensus are described, respectively. Results of simulation examples are also reported. Concluding remarks and directions for future research are summarized in Section 7.

1.1 Notation

Let $\mathbb{R}^{\mathsf{n}}$ be the set of real vectors of dimension n and $\mathbb{R}^{+}$ be the set of positive real number, excluding zero. Given a set $\mathcal{A}$, let $\breve{\mathcal{A}}$ be the complement of $\mathcal{A}$.

Given a vector $a \in \mathbb{R}^{\mathsf{n}}$, $\|a\|_2$ is the Euclidean norm of a. Given a matrix $A \in \mathbb{R}^{\mathsf{n}\times\mathsf{p}}$, A' denotes the transpose of A. Given a set $\mathcal{A}$, let $\mathcal{P}_{\mathcal{A}}$ denote the Euclidean projection onto $\mathcal{A}$. Let I_{n} be the identity matrix of size n and 0_{n} be an n-dimensional column vector of ones.

2 Alternating Direction Method of Multipliers

The Alternating Direction Method of Multipliers (ADMM) [4] is an algorithm tailored for problems in the form

$$\begin{array}{ll} \text{minimize} & f(\theta)+ \\ \text{subject to} & A\theta + Bz = c, \end{array} \tag{1}$$

where $\theta \in \mathbb{R}^{n_\theta}$, $z \in \mathbb{R}^{n_z}$, $f : \mathbb{R}^{n_\theta} \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^n_z \to \mathbb{R} \cup \{+\infty\}$ are closed, proper, convex functions and $A \in \mathbb{R}^{p\times n_\theta}$, $B \in \mathbb{R}^{p\times n_z}$, $c \in \mathbb{R}^p$.

To solve Problem (1), the ADMM iterations to be performed are

$$\theta^{(k+1)} = \underset{\theta}{\operatorname{argmin}}\ \mathcal{L}(\theta, z^{(k)}, \delta^{(k)}), \tag{2}$$

$$z^{(k+1)} = \underset{z}{\operatorname{argmin}}\ \mathcal{L}(\theta^{(k+1)}, z, \delta^{(k)}), \tag{3}$$

$$\delta^{(k+1)} = \delta^{(k)} + \rho(A\theta^{(k+1)} + Bz^{(k+1)} - c), \tag{4}$$

where $k \in \mathbb{N}$ indicates the ADMM iteration, $\mathcal{L}$ is the augmented Lagrangian associated to (1), i.e.

$$\mathcal{L}(\theta, z, \delta) = f(\theta) + g(z) + \delta' (A\theta + Bz - c) + \frac{\rho}{2} \|A\theta + Bz - c\|_2^2, \tag{5}$$

$\delta \in \mathbb{R}^p$ is the Lagrange multiplier and $\rho \in \mathbb{R}^+$ is a tunable parameter (see [4] for possible tuning strategies). Iterations (2)-(4) have to be run until a stopping criteria is satisfied, e.g. the maximum number of iterations is attained.

It has to be remarked that the convergence of ADMM to high accuracy results might be slow (see [4] and references therein). However, the results obtained with a few tens of iterations are usually accurate enough for most of applications. For further details, the reader is referred to [4].

2.1 ADMM for constrained convex optimization

Suppose that the problem to be addressed is

$$\begin{array}{ll} \min\limits_{\theta} & f(\theta) \\ \text{s.t.} & \theta \in \mathcal{C}, \end{array} \tag{6}$$

with $\theta \in \mathbb{R}^{n_\theta}$, $f : \mathbb{R}^{n_\theta} \to \mathbb{R} \cup \{+\infty\}$ being a closed, proper, convex function and $\mathcal{C}$ being a convex set, representing constraints on the parameter value.
As explained in [4], (6) can be recast in the same form as (1) through the

introduction of the auxiliary variable $z \in \mathbb{R}^{n_\theta}$ and the indicator function of set $\mathcal{C}$, i.e.

$$g(z) = \begin{cases} 0 & \text{if } z \in \mathcal{C} \\ +\infty & \text{otherwise} \end{cases}. \tag{7}$$

In particular, (6) can be equivalently stated as

$$\begin{aligned} \min_{\theta,z} \quad & f(\theta) + g(z) \\ \text{s.t.} \quad & \theta - z = 0. \end{aligned} \tag{8}$$

Then, the ADMM scheme to solve (8) is

$$\theta^{(k+1)} = \underset{\theta}{\operatorname{argmin}}\ \mathcal{L}(\theta, z^{(k)}, \delta^{(k)}), \tag{9}$$

$$z^{(k+1)} = \mathcal{P}_{\mathcal{C}}(\theta^{(k+1)} + \delta^{(k)}), \tag{10}$$

$$\delta^{(k+1)} = \delta^{(k)} + \rho(\theta^{(k+1)} - z^{(k+1)}) \tag{11}$$

with $\mathcal{L}$ equal to

$$\mathcal{L}(\theta, z, \delta) = f(\theta) + g(z) + \delta'(\theta - z) + \frac{\rho}{2}\|\theta - z\|_2^2$$

2.2 ADMM for consensus problems

Consider the optimization problem given by

$$\min_{\theta^g} \sum_{n=1}^{N} f_n(\theta^g), \tag{12}$$

where $\theta^g \in \mathbb{R}^{n_\theta}$ and each term of the objective, i.e. $f_n : \mathbb{R}^{n_\theta} \to \mathbb{R} \cup \{+\infty\}$, is a proper, closed, convex function.
Suppose that N processors are available to solve (12) and that, consequently, we are not interested in a centralized solution of the consensus problem. As explained in [4], ADMM can be used to reformulate the problem so that each term of the cost function in (12) is handled by its own.
In particular, (12) can be reformulated as

$$\begin{aligned} \text{minimize} \quad & \sum_{n=1}^{N} f_n(\theta_n) \\ \text{subject to} \quad & \theta_n - \theta^g = 0 \qquad n = 1, \ldots, N. \end{aligned} \tag{13}$$

Note that, thanks to the introduction of the consensus constraint, the cost function in (13) is now separable.
The augmented Lagrangian correspondent to (13) is given by

$$\mathcal{L}(\{\theta_n\}_{n=1}^N, \theta^g, \{\delta_n\}_{n=1}^N) = \sum_{n=1}^{N} \left(f_n(\theta_n) + \delta_n'(\theta_n - \theta^g) + \frac{\rho}{2}\|\theta_n - \theta^g\|_2^2 \right), \tag{14}$$

and the ADMM iterations are

$$\theta_n^{(k+1)} = \underset{\theta_n}{\operatorname{argmin}}\ \mathcal{L}_n(\theta_n, \delta_n^{(k)}, \theta^{g,(k)}),\ n = 1, \dots, N \tag{15}$$

$$\theta^{g,(k+1)} = \frac{1}{N}\sum_{n=1}^{N}\left(\theta_n^{(k+1)} + \frac{1}{\rho}\delta_n^{(k)}\right), \tag{16}$$

$$\delta_n^{(k+1)} = \delta_n^{(k)} + \rho\left(\theta_n^{(k+1)} - \theta^{g,(k+1)}\right),\ n = 1, \dots, N \tag{17}$$

with

$$\mathcal{L}_n = f_n(\theta_n) + (\delta_n)'(\theta_n - \theta^g) + \frac{\rho}{2}\|\theta_n - \theta^g\|_2^2.$$

Note that, on the one hand (15) and (17) can be carried out independently by each agent $n \in \{1, \dots, N\}$, (16) depends on all the updated local estimates. The global estimate should thus be updated in a "fusion center", where all the local estimates are collected and merged.

3 Problem statement

Assume that (i) measurements acquired by N agents are available and that (ii) the behavior of the N data-generating systems is described by the same known model. Suppose that some parameters of the model, $\theta_n \in \mathbb{R}^{n_\theta}$ with $n = 1, \dots, N$, are unknown and that their value has to to be retrieved from data. As the agents share the same model, it is also legitimate to assume that (iii) there exist a set of parameters $\theta^g \in \mathbb{R}^{n_g}$, with $n_g \leq n_\theta$, common to all the agents.

We aim at (i) retrieving *local* estimates of $\{\theta_n\}_{n=1}^N$, employing information available at the local level only, and (ii) identifying the *global* parameter θ^g at the "cloud" level, using the data collected from all the available sources. To accomplish these tasks (i) N local processors and (ii) and a "cloud", where the data are merged are needed.

The considered estimation problem can be cast into a separable optimization problem, given by

$$\begin{aligned} \min_{\theta_n} \quad & \sum_{n=1}^{N} f_n(\theta_n) \\ \text{s.t.} \quad & F(\theta_n) = \theta^g, \\ & \theta_n \in \mathcal{C}_n,\ n = 1, \dots, N \end{aligned} \tag{18}$$

where $f_n : \mathbb{R}^{n_\theta} \to \mathbb{R} \cup \{+\infty\}$ is a closed, proper, convex function, $F : \mathbb{R}^{n_\theta} \to \mathbb{R}^{n_g}$ is a nonlinear operator and $\mathcal{C}_n \subset \mathbb{R}^{n_\theta}$ is a convex set representing constraints on the parameter values. Note that, constraints on the value of the global parameter can be enforced if $\mathcal{C}_n = \mathcal{C} \cup \{\mathcal{C}_n \cap \check{\mathcal{C}}\}$, with $\theta \in \mathcal{C}$.
Assume that the available data are the output/regressor pairs collected from each agent $n \in \{1, \dots, N\}$ over an horizon of length $T \in \mathbb{N}$, i.e. $\{y_n(t), X_n(t)\}_{t=1}^T$. Relying on the hypothesis that the regressor/output relationship is well modelled as

$$y_n(t) = X_n(t)'\theta_n + e_n(t), \tag{19}$$

with $e_n(t) \in \mathbb{R}^{n_y}$ being a zero-mean additive noise independent of the regressor $X_n(t) \in \mathbb{R}^{n_\theta \times n_y}$, we will focus on developing a recursive algorithm to solve (18) with the local cost functions given by

$$f_n(\theta_n) = \frac{1}{2} \sum_{t=1}^{T} \lambda_n^{T-t} \left\| y_n(t) - X_n(t)' \theta_n \right\|_2^2 . \tag{20}$$

The forgetting factor $\lambda_n \in (0, 1]$ is introduced to be able to estimate time-varying parameters. Note that different forgetting factors can be chosen for different agents.

Remark 1 ***ARX models***
Suppose that an AutoRegressive model with eXogenous inputs (ARX) has to be identified from data. The input/output relationship is thus given by

$$\begin{aligned} y(t) = \theta_1 y(t-1) + \ldots + \theta_{n_a} y(t-n_a) + \\ + \theta_{n_a+1} u(t-n_k-1) + \ldots + \theta_{n_a+n_b} u(t-n_k-n_b) + e(t) \end{aligned} \tag{21}$$

where u is the deterministic input, $\{n_a, n_b\}$ indicate the order of the system, n_k is the input/output delay.
Note that (21) *can be recast as the output/regressor relationship with the regressor defined as*

$$X(t) = \begin{bmatrix} y(t-1)' & \ldots & y'(t-n_a) & u(t-n_k-1)' & \ldots & u(t-n_k-n_b)' \end{bmatrix}' \tag{22}$$

It is worth to point out that, in the considered framework, the parameters n_a, n_b and n_k are the same for all the N agents, as they are supposed to be described by the same model. ■

4 Collaborative estimation for full consensus

Suppose that the problem to be solve is (12), i.e. we are aiming at achieving full consensus among N agents. Consequently, the consensus constraint in (18) has to be modified as

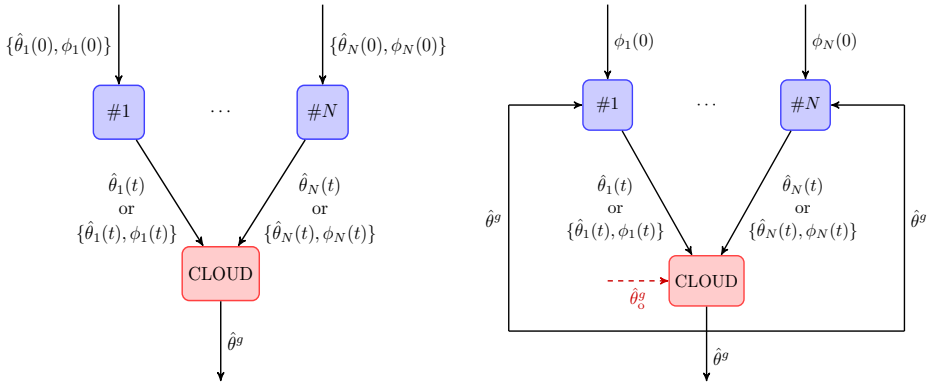
$$F(\theta_n) = \theta^g \rightarrow \theta_n = \theta^g$$

and $\mathcal{C}_n = \mathbb{R}^{n_\theta}$, so that $\theta_n \in \mathcal{C}_n$ can be neglected for $n = 1, \ldots, N$. Moreover, as we are focusing on the problem of collaborative least-squares estimation, we are interested in the particular case in which the local cost functions in (13) are equal to (20) .

Even if the considered problem can be solved in a centralized fashion, our goal is to obtain estimates of the unknown parameters both (i) at a local level and (ii) on the "cloud". With the objective of distributing the computation among the local processors and the "cloud", we propose 5 approaches to address (13).

4.1 Greedy approaches

All the proposed 'greedy' approaches rely on the use, by each local processor, of the standard Recursive Least-Squares (RLS) method (see [17]) to update the



(a) S-RLS and SW-RLS (b) M-RLS and MW-RLS

Figure 2: Greedy approaches. Schematic of the information exchanges between the agents and the "cloud".

local estimates, $\{\hat{\theta}_n\}_{n=1}^N$. Depending on the approach, $\{\hat{\theta}_n\}_{n=1}^N$ are then combined on the "cloud" to update the estimate of the global parameter.

The first two methods that are used to compute the estimates of the unknown parameters both (i) locally and (ii) on the "cloud" are:

1. **Static RLS (S-RLS)** The estimate of the global parameter is computed as
$$\hat{\theta}^g = \frac{1}{N}\sum_{n=1}^{N} \hat{\theta}_n(t). \tag{23}$$

2. **Static Weighted RLS (SW-RLS)** Consider the matrices $\{\phi_n\}_{n=1}^N$, obtained applying standard RLS at each node (see [17]), and assume that $\{\phi_n\}_{n=1}^N$ are always invertible. The estimate $\hat{\theta}^g$ is computed as the weighted average of the local estimates
$$\hat{\theta}^g = \left(\sum_{n=1}^{N} \phi_n(t)^{-1}\right)^{-1} \left(\sum_{n=1}^{N} \phi_n(t)^{-1}\hat{\theta}_n(t)\right). \tag{24}$$
Considering that ϕ_n is an indicator of the accuracy of the nth local estimate, (24) allows to weight more the "accurate " estimates then the "inaccurate" ones.

S-RLS and SW-RLS allow to achieve our goal, i.e. (i) obtain a local estimate of the unknowns and (ii) compute $\hat{\theta}^g$ using all the information available. However, looking at the scheme in Figure 2(a) and at Algorithm 1, it can be noticed that the global estimate is not used at a local level.

Thanks to the dependence of $\hat{\theta}^g$ on all the available information, the local use of the global estimate might enhance the accuracy of $\{\hat{\theta}_n\}_{n=1}^N$. Motivated by this observation, we introduce two additional methods:

4. **Mixed RLS (M-RLS)**

Algorithm 1 S-RLS and SW-RLS

Input: Sequence of observations $\{X_n(t), y_n(t)\}_{t=1}^{T}$, initial matrices $\phi_n(0) \in \mathbb{R}^{n_\theta \times n_\theta}$, initial estimates $\hat{\theta}_n(0) \in \mathbb{R}^{n_\theta}$, $n = 1, \ldots, N$

1. **for** $t = 1, \ldots, T$ **do**

   <u>Local</u>

   1.1. **for** $n = 1, \ldots, N$ **do**

   1.1.1. **compute** $K_n(t)$, $\phi_n(t)$ and $\hat{\theta}_n(t)$ with standard RLS [17];

   1.2. **end for**;

   <u>Global</u>

   1.1. **compute** $\hat{\theta}^g$;

2. **end**.

Output: Local estimates $\{\hat{\theta}_n(t)\}_{t=1}^{T}$, $n = 1, \ldots, N$, estimated global parameters $\{\hat{\theta}^g(t)\}_{t=1}^{T}$.

Algorithm 2 M-RLS and MW-RLS

Input: Sequence of observations $\{X_n(t), y_n(t)\}_{t=1}^{T}$, initial matrices $\phi_n(0) \in \mathbb{R}^{n_\theta \times n_\theta}$, $n = 1, \ldots, N$, initial estimate $\hat{\theta}_0^g$.

1. **for** $t = 1, \ldots, T$ **do**

   <u>Local</u>

   1.1. **for** $n = 1, \ldots, N$ **do**

   1.1.1. **set** $\hat{\theta}_n(t-1) = \hat{\theta}^g(t-1)$;

   1.1.2. **compute** $K_n(t)$, $\phi_n(t)$ and $\hat{\theta}_n(t)$ with standard RLS [17];

   1.2. **end for**;

   <u>Global</u>

   1.1. **compute** $\hat{\theta}^g$;

2. **end**.

Output: Local estimates $\{\hat{\theta}_n(t)\}_{t=1}^{T}$, $n = 1, \ldots, N$, estimated global parameters $\{\hat{\theta}^g(t)\}_{t=1}^{T}$.

5. Mixed Weighted RLS (MW-RLS)

While M-RLS relies on (23), in MW-RLS the local estimates are combined as in (24). However, as shown in Figure 2(b) and outlined in Algorithm 2, the global estimate $\hat{\theta}^g$ is fed to the each local processor and used to update the local estimates instead of their values at the previous step. Note that, especially at the beginning of the estimation horizon, the approximation made in M-RLS and MW-RLS might affect negatively some of the local estimates, e.g. the ones obtained by the agents characterized by a relatively small level of noise.

Remark 2 *While S-RLS and M-RLS require the local processors to transmit to the "cloud" only* $\{\hat{\theta}_n\}_{n=1}^N$*, the pairs* $\{\hat{\theta}_n, \phi_n\}_{n=1}^N$ *have to be communicated to the "cloud" with both SW-RLS and MW-RLS (see* (23) *and* (24)*, respectively). Moreover, as shown in Figure 2, while S-RLS and SW-RLS require Node-to-Cloud-to-Node (N2C2N) transmissions, M-RLS and MW-RLS are based on a Node-to-Cloud (N2C) communication policy.* ■

4.2 ADMM-based RLS (ADMM-RLS) for full consensus

Instead of resorting to greedy methods, we propose to solve (12) with ADMM. Note that the same approach has been used to develop a fully distributed scheme for consensus-based estimation over Wireless Sensor Networks (WSNs) in [20]. However, our approach differs from the one introduced in [20] as we aim at exploiting the "cloud" to attain consensus and, at the same time, we want local estimates to be computed by each node.
As the problem to be solved is equal to (13), the ADMM iterations to be performed are (15)-(17), i.e.

$$\hat{\theta}_n(T)^{(k+1)} = \underset{\theta_n}{\operatorname{argmin}} \left\{ f_n(\theta_n) + (\delta_n^{(k)})'(\theta_n - \hat{\theta}^{g,(k)}) + \frac{\rho}{2}\|\theta_n - \hat{\theta}^{g,(k)}\|_2^2 \right\},$$

$$\hat{\theta}^{g,(k+1)} = \frac{1}{N}\sum_{n=1}^{N}\left(\theta_n^{(k+1)} + \frac{1}{\rho}\delta_n^{(k)}\right),$$

$$\delta_n^{(k+1)} = \delta_n^{(k)} + \rho\left(\hat{\theta}_n^{(k+1)}(T) - \hat{\theta}^{g,(k+1)}\right),\ n = 1, \ldots, N$$

with the cost functions f_n defined as in (14) and where the dependence on T of the local estimates is stressed to underline that only the updates of $\hat{\theta}_n$ are directly influenced by the current measurements. Note that the update for $\hat{\theta}^g$ is a combination of the mean of the local estimates, i.e. (23), and the mean of the Lagrange multipliers.
As (16)-(17) are independent from the specific choice of $f_n(\theta_n)$, we focus on the update of the local estimates, i.e. (15), with the ultimate goal of finding recursive updates for $\hat{\theta}_n$.

Thanks to the characteristics of the chosen local cost functions, the closed-form solution for the problem in (15) is given by

$$\hat{\theta}_n^{(k+1)}(T) = \phi_n(T)\left(\mathcal{Y}_n(T) - \delta_n^{(k)} + \rho\hat{\theta}^{g,(k)}\right), \tag{25}$$

$$\mathcal{Y}_n(t) = \sum_{\tau=1}^{t}\lambda_n^{t-\tau}X_n(\tau)y_n(\tau), \quad t = 1, \ldots, T, \tag{26}$$

$$\phi_n(t) = \left(\sum_{\tau=1}^{t}\lambda_n^{t-\tau}X_n(\tau)(X_n(\tau))' + \rho I_{n_\theta}\right)^{-1}, \quad t = 1, \ldots, T. \tag{27}$$

With the aim of obtaining recursive formulas to update $\hat{\theta}_n$, consider the local estimate obtained at $T-1$, which is given by

$$\hat{\theta}_n(T-1) = \phi_n(T-1)\left(\mathcal{Y}_n(T-1) + \rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right), \tag{28}$$

with $\delta_n(T-1)$ and $\hat{\theta}^g(T-1)$ denoting the Lagrange multiplier and the global estimate computed at $T-1$, respectively. It has then to be proven that $\hat{\theta}_n^{(k)}(T-1)$ can be computed as a function of $\hat{\theta}(T-1)$, $y_n(T)$ and $X_n(T)$.

Consider the inverse matrix ϕ_n (27), given by

$$\phi_n(T)^{-1} = \mathcal{X}_n(T) + \rho I_{n_\theta},$$
$$\mathcal{X}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau)(X_n(\tau))'.$$

Based on (27), it can be proven that $\phi_n(T)^{-1}$ can be computed as a function of $\phi_n(T-1)^{-1}$. In particular:

$$\begin{aligned}
&\phi_n(T)^{-1} = \mathcal{X}_n(T) + \rho I_{n_\theta} = \\
&= \lambda_n \mathcal{X}_n(T-1) + X_n(T)(X_n(T))' + \rho I_{n_\theta} = \\
&= \lambda_n \left[\mathcal{X}_n(T-1) + \rho I_{n_\theta}\right] + X_n(T)(X_n(T))' + (1-\lambda_n)\rho I_{n_\theta} = \\
&= \lambda_n \phi_n(T-1)^{-1} + X_n(T)(X_n(T))' + (1-\lambda_n)\rho I_{n_\theta}. \qquad (29)
\end{aligned}$$

Introducing the extended regressor vector $\tilde{X}_n(T)$

$$\tilde{X}_n(T) = \begin{bmatrix} X_n(T) & \sqrt{(1-\lambda_n)\rho} I_{n_\theta} \end{bmatrix} \in \mathbb{R}^{n_\theta \times (n_y + n_\theta)}, \qquad (30)$$

(29) can then be further simplified as

$$\phi_n(T)^{-1} = \lambda_n \phi_n(T-1)^{-1} + \tilde{X}_n(T)(\tilde{X}_n(T))'.$$

Applying the matrix inversion lemma, the resulting recursive formulas to update ϕ_n are

$$\mathcal{R}_n(T) = \lambda_n I_{(n_y+n_\theta)} + (\tilde{X}_n(T))' \phi_n(T-1) \tilde{X}_n(T), \qquad (31)$$
$$K_n(T) = \phi_n(T-1)\tilde{X}_n(T)(\mathcal{R}_n(T))^{-1}, \qquad (32)$$
$$\phi_n(T) = \lambda_n^{-1}\left(I_{n_\theta} - K_n(T)(\tilde{X}_n(T))'\right)\phi_n(T-1). \qquad (33)$$

Note that the gain K_n and matrix ϕ_n are updated as in standard RLS (see [17]), with the exceptions of the increased dimension of the identity matrix in (31) and the substitution of the regressor with $\tilde{X}_n$. Only when $\lambda_n = 1$ the regressor X_n and $\tilde{X}_n$ are equal. Moreover, observe that (31)-(33) are independent from k and, consequently, $\{\mathcal{R}_n, K_n, \phi_n\}_{n=1}^N$ can be updated once fer step t.

Consider again (25). Adding and subtracting

$$\lambda_n \phi_n(T)\left[\rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right]$$

to (25), the solution of (15) corresponds to

$$\begin{aligned}
\hat{\theta}_n^{(k+1)}(T) = \phi_n(T)\Big[&\lambda_n\left(\mathcal{Y}_n(T-1) - \delta_n(T-1) + \rho\hat{\theta}^g(T-1)\right) + \\
&+ X_n(T)y_n(T) - \left(\delta_n^{(k)} - \lambda_n\delta_n(T-1)\right) + \rho\left(\hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1)\right)\Big] = \\
&= \hat{\theta}_n^{RLS}(T) + \hat{\theta}_n^{ADMM,(k+1)}(T), \qquad (34)
\end{aligned}$$

with

$$\hat{\theta}_n^{RLS}(T) = \phi_n(T)\left\{\lambda_n\left(\mathcal{Y}_n(T-1) + \rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right) + \right.$$
$$\left. + X_n(T)y_n(T)\right\}, \tag{35}$$
$$\hat{\theta}_n^{ADMM,(k+1)}(T) = \phi_n(T)\left[\rho\Delta_{g,\lambda_n}^{(k+1)}(T) - \Delta_{\lambda_n}^{(k+1)}(T)\right], \tag{36}$$

and

$$\Delta_{g,\lambda_n}^{k+1}(T) = \hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1), \tag{37}$$
$$\Delta_{\lambda_n}^{(k+1)}(T) = \delta_n^{(k)} - \lambda_n\delta_n(T-1). \tag{38}$$

Observe that (36) is independent from the past data-pairs $\{y_n(t), X_n(t)\}_{t=1}^T$, while (35) depends on $\mathcal{Y}_n(T-1)$. Aiming at obtaining recursive formulas to update $\hat{\theta}_n$, the dependence of (35) should be eliminated.

Consider (35). Exploiting (33) and (28), $\hat{\theta}_n^{RLS}(T)$ is given by

$$\begin{aligned}\hat{\theta}_n^{RLS}(T) &= \phi_n(T-1)\left\{\left(\mathcal{Y}_n(T-1) + \rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right)\right\} + \\ &- K_n(T)(\tilde{X}_n(T)')\phi_n(T-1)\left\{\left(\mathcal{Y}_n(T-1) + \rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right)\right\} + \\ &+ \phi_n(T)X_n(T)y_n(T) = \\ &= \hat{\theta}_n(T-1) - K_n(T)(\tilde{X}_n(T))'\hat{\theta}_n(T-1) + \phi_n(T)X_n(T)y_n(T).\end{aligned} \tag{39}$$

For (39) to be dependent on the extended regressor only, we define the extended measurement vector

$$\tilde{y}_n(T) = \begin{bmatrix}(y_n(T))' & 0_{1\times n_g}\end{bmatrix}'.$$

The introduction of $\tilde{y}_n$ yields (39) can be modified as

$$\hat{\theta}_n^{RLS}(T) = \hat{\theta}_n(T-1) - K_n(T)(\tilde{X}_n(T))'\hat{\theta}_n(T-1) + \phi_n(T)\tilde{X}_n(T)\tilde{y}_n(T).$$

Notice that the equality $\phi_n(T)\tilde{X}_n(T) = K_n(T)$ holds and it can be proven as follows

$$\begin{aligned}&\phi_n(T)\tilde{X}_n(T) = \lambda_n^{-1}\left(I_{n_\theta} - K_n(T)(\tilde{X}_n(T))'\right)\phi_n(T-1)\tilde{X}_n(T) = \\ &= \lambda_n^{-1}\left(I_{n_\theta} - \phi_n(T-1)\tilde{X}_n(T)(\mathcal{R}_n(T))^{-1}(\tilde{X}_n(T))'\right)\phi_n(T-1)\tilde{X}_n(T) = \\ &= \phi_n(T-1)\tilde{X}_n(T)\left(\lambda_n^{-1}I_{n_\theta} - \lambda_n^{-1}(\mathcal{R}_n(T))^{-1}(\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T)\right) = \\ &= \phi_n(T-1)\tilde{X}_n(T)\left(\lambda_n^{-1}I_{n_\theta} + \right. \\ &\left.-\lambda_n^{-1}(\lambda_n I_{(n_y+n_\theta)} + (\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T))^{-1}(\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T)\right) = \\ &= \phi_n(T-1)\tilde{X}_n(T)\left(\lambda_n^{-1}I_{n_\theta} - \lambda_n^{-1}(I_{(n_y+n_\theta)} + \lambda_n^{-1}(\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T))^{-1}\cdot\right. \\ &\left.\cdot(\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T)\lambda_n^{-1}\right) = \\ &= \phi_n(T-1)\tilde{X}_n(T)\left(\lambda_n I_{n_\theta} + (\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T)\right)^{-1} = K_n(T),\end{aligned}$$

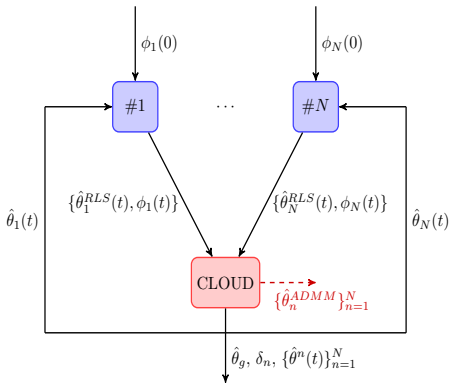


Figure 3: ADMM-RLS. Schematic of the information exchanges between the agents and the "cloud"when using a N2C2N communication scheme.

where the matrix inversion lemma and (32)-(33) are used.
It can thus be proven that $\hat{\theta}_n^{RLS}$ can be updated as

$$\hat{\theta}_n^{RLS}(T) = \hat{\theta}_n(T-1) + K_n(T)(\tilde{y}_n(T) - \tilde{X}_n(T)'\hat{\theta}_n(T-1)). \tag{40}$$

While the update for $\hat{\theta}_n^{ADMM}$ (36) depends on both the values of the Lagrange multipliers and the global estimates, $\hat{\theta}_n^{RLS}$ (40) is computed on the basis of the previous local estimate and the current measurements. Consequently, $\hat{\theta}_n^{RLS}$ is updated recursively.

Under the hypothesis that both $\hat{\theta}^g$ and δ_n are stored on the "cloud", it does seems legitimate to update $\hat{\theta}^g$ and δ_n on the "cloud", along with $\hat{\theta}_n^{ADMM}$. Instead, the partial estimates $\hat{\theta}_n^{RLS}$, $n = 1, \ldots, N$, can be updated by the local processors. Thanks to this choice, the proposed method, summarized in Algorithm 3 and Figure 3, allows to obtain estimates both at the (i) agent and (ii) "cloud" level.
Observe that, thanks to the independence of (40) from k, $\hat{\theta}_n^{RLS}$ can be updated once per step t. The local updates are thus regulated by a local clock and not by the one controlling the ADMM iterations on the "cloud".
Looking at (31)-(33) and (40), it can be noticed that $\hat{\theta}_n^{RLS}$ is updated through standard RLS, with the exceptions that, at step $t \in \{1, \ldots, T\}$, the update depends on the previous local estimate $\hat{\theta}_n(t-1)$ instead of depending on $\hat{\theta}_n^{RLS}(t-1)$ and that the output/regressor pair $\{y_n(t), X_n(t)\}$ is replaced with $\{\tilde{y}_n(t), \tilde{X}_n(t)\}$. As a consequence, the proposed method can be easily integrated with pre-existing RLS estimators already available locally.

Remark 3 *Algorithm 1 requires the initialization of the local and global estimates. If some data are available to be processed in a batch mode, $\hat{\theta}_n(0)$ can be chosen as the best linear model, i.e.*

$$\hat{\theta}_n(0) = \underset{\theta_n}{\operatorname{argmin}} \sum_{t=1}^{\tau} \|y_n(t) - X_n(t)'\theta\|_2^2$$

and $\hat{\theta}^g(0)$ can be computed as the mean of $\{P\hat{\theta}_n(0)\}_{n=1}^N$. Moreover, the matrices ϕ_n, $n = 1, \ldots, N$, can be initialized as $\phi_n(0) = \gamma I_{n_\theta}$, with $\gamma > 0$. ■

Algorithm 3 ADMM-RLS for full consensus (N2C2N)

Input: Sequence of observations $\{X_n(t), y_n(t)\}_{t=1}^T$, initial matrices $\phi_n(0) \in \mathbb{R}^{n_\theta \times n_\theta}$, initial local estimates $\hat{\theta}_n(0)$, initial dual variables $\delta_{n,\mathrm{o}}$, $n = 1, \ldots, N$, initial global estimate $\hat{\theta}_{\mathrm{o}}^g$, parameter $\rho \in \mathbb{R}^+$.

1. **for** $t = 1, \ldots, T$ **do**

   <u>Local</u>

   1.1. **for** $n = 1, \ldots, N$ **do**
   1.1.1. **compute** $\tilde{X}_n(t)$ as in (30);
   1.1.2. **compute** $K_n(t)$ and $\phi_n(t)$ with (32) - (33);
   1.1.3. **compute** $\hat{\theta}_n^{RLS}(t)$ with (40);
   1.2. **end for**;

   <u>Global</u>

   1.1. **do**
   1.1.1. **compute** $\hat{\theta}_n^{ADMM,(k+1)}(t)$ with (36), $n = 1, \ldots, N$;
   1.1.2. **compute** $\hat{\theta}^{g,(k+1)}(t)$ with (16);
   1.1.3. **compute** $\delta_n^{(k+1)}$ with (17), $n = 1, \ldots, N$;
   1.2. **until** a stopping criteria is satisfied (e.g. maximum number of iterations attained);

2. **end**.

Output: Estimated global parameters $\{\hat{\theta}^g(t)\}_{t=1}^T$, estimated local parameters $\{\hat{\theta}_n(t)\}_{t=1}^T$, $n = 1, \ldots, N$.

Remark 4 *The chosen implementation requires $\hat{\theta}_n^{RLS}$ and ϕ_n to be transmitted from the local processors to the "cloud" at each step, while the "cloud"has to communicate $\hat{\theta}_n$ to all the agents. As a consequence, the proposed approach is based on N2C2N transmissions.* ■

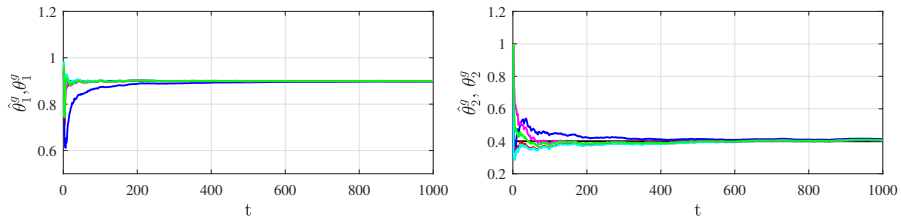
4.3 Example 1. Static parameters

Suppose that N data-generating systems are described by the following models

$$y_n(t) = 0.9y_n(t-1) + 0.4u_n(t-1) + e_n(t), \tag{41}$$

where $y_n(t) \in \mathbb{R}$, $X_n(t) = \left[\, y_n(t-1) \;\; u_n(t-1) \,\right]'$, u_n is known and is generated in this example as a sequence of i.i.d. elements uniformly distributed in the interval $[\,2 \;\; 3\,]$ and $e_n \sim \mathcal{N}(0, R_n)$ is a white noise sequence, with $\{R_n \in \mathbb{N}\}_{n=1}^N$ randomly chosen in the interval $[\,1 \;\; 30\,]$. Evaluating the effect of the noise on the output y_n through the Signal-to-Noise Ratio SNR_n, i.e.

$$\mathrm{SNR}_n = 10 \log \frac{\sum_{t=1}^T \left(y_n(t) - e_n(t)\right)^2}{\sum_{t=1}^T e_n(t)^2} \; dB \tag{42}$$



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) θ_2^g vs $\hat{\theta}_2^g$

Figure 4: Example 1. True vs estimated parameters. Black : true, red : C-RLS, blue : S-RLS, cyan : SW-RLS, magenta : M-RLS, green : MW-RLS.

the chosen covariance matrices yield $\text{SNR}_n \in [7.8\ 20.8]$ dB, $n = 1, \ldots, N$. Note that (41) can be equivalently written as

$$y_n(t) = (X_n(t))'\theta^g + e_n(t) \text{ with } \theta^g = [\,0.9\ 0.4\,]'$$

and the regressor $X_n(t)$ is defined as in (22), i.e. $X_n = [\,y_n(t-1)\ u_n(t-1)\,]$.

Observe that the deterministic input sequences $\{u_n(t)\}_{t=1}^T$ are all different. However, they are all generated accordingly to the same distribution, as it seems reasonable to assume that systems described by the same model are characterized by similar inputs.

Initializing ϕ_n as $\phi_n(0) = 0.1 I_{n_\theta}$, while $\hat{\theta}_n(0)$ and $\hat{\theta}_0^g$ are sampled from the distributions $\mathcal{N}(\hat{\theta}^g, 2I_{n_\theta})$ and $\mathcal{N}(\hat{\theta}^g, I_{n_\theta})$, respectively, and $\{\lambda_n = \Lambda\}_{n=1}^N$, with $\Lambda = 1$, we first evaluate the performance of the greedy approaches. The actual parameter θ^g and the estimate obtained with the different greedy approaches are reported in Figure 4.
Despite the slight difference performances in the first 300 steps, which seems to be legitimate, the estimates obtained with SW-RLS, M-RLS and MW-RLS are similar. Moreover, $\hat{\theta}^g$ obtained with the different methods are comparable with respect with the estimate computed with C-RLS.
In particular, the similarities between the estimates obtained with M-RLS, MW-RLS and C-RLS prove that, in the considered case, the choice of the "mixed" strategy allows to enhance the accuracy of $\hat{\theta}^g$. Comparing the estimates obtained with S-RLS and SW-RLS, observe that the convergence of the estimate to the actual value of θ^g tends to be faster if $\hat{\theta}^g$ is computed as in (24).

Setting $\rho = 0.1$, the performance of the ADMM-RLS are assessed for different values of N and T. Moreover, the retrieved estimates are compared to the ones obtained with C-RLS and the greedy approaches.
The accuracy of the estimate $\hat{\theta}^g$ is assessed through the Root Mean Square Error (RMSE), i.e.

$$\text{RMSE}_i^g = \sqrt{\frac{\sum_{t=1}^T \left(\theta_i^g - \hat{\theta}_i^g(t)\right)^2}{T}},\ i = 1, \ldots, n_g. \tag{43}$$

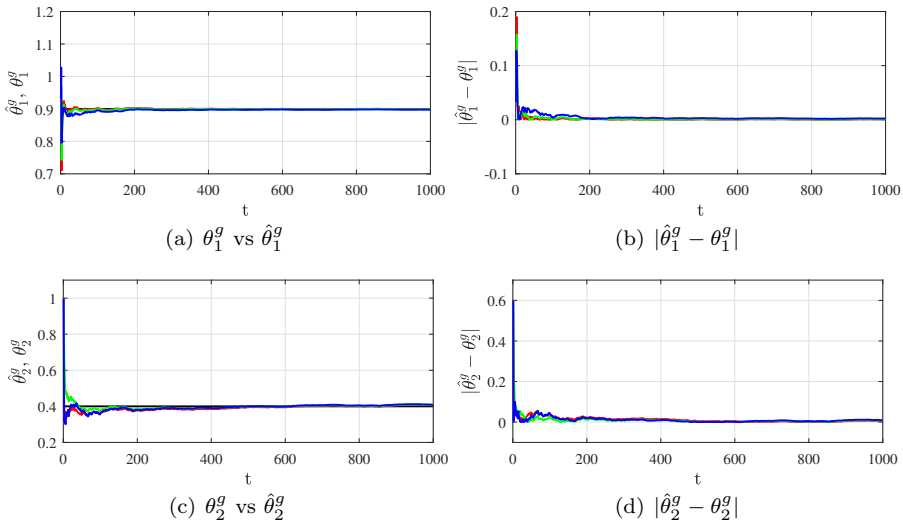
As expected (see Table 1), the accuracy of the estimates tends to increase if

Table 1: ADMM-RLS: $\|\text{RMSE}^g\|_2$

N \ T	**10**	**10^2**	**10^3**	**10^4**
2	1.07	0.33	0.16	0.10
10	0.55	0.22	0.09	0.03
10^2	0.39	0.11	0.03	0.01

Table 2: $\|\text{RMSE}^g\|_2$: C-RLS and greedy methods vs ADMM-RLS

	Method					
	C-RLS	S-RLS	SW-RLS	M-RLS	MW-RLS	ADMM-RLS
$\|\text{RMSE}^g\|_2$	0.03	0.05	0.03	0.04	0.03	0.03



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) $|\hat{\theta}_1^g - \theta_1^g|$

(c) θ_2^g vs $\hat{\theta}_2^g$ (d) $|\hat{\theta}_2^g - \theta_2^g|$

Figure 5: Example 1. Model parameters. Black : true, red : C-RLS, green : MW-RLS, blue : ADMM-RLS.

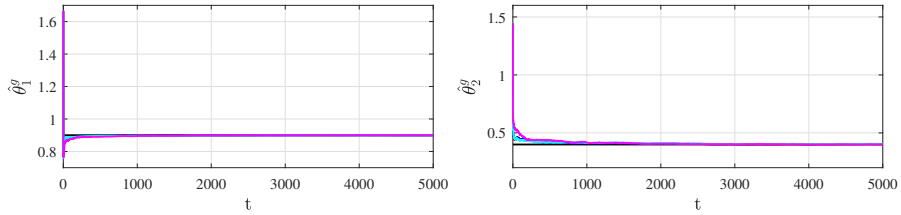
the number of local processors N and the estimation horizon T increase. In the case $N = 100$ and $T = 1000$, the estimates obtained with both C-RLS and the SW-RLS and MW-RLS have comparable accuracy. See Table 2.
The estimates obtained with ADMM-RLS, C-RLS and MW-RLS are further compared in Figure 5 and, as expected the three estimates are barely distinguishable. Thus the proposed ADMM-RLS algorithm, which uses local estimates and the cloud, is able to obtain good accuracy versus the fully centralized approach. Moreover, ADMM-RLS allows to retrieve estimates as accurate as the ones obtained with the MW-RLS, i.e. the greedy approach associated with the least RMSE.

Table 3: Example 1. $\|\text{RMSE}^g\|_2$ vs N_{ni}

	N_{ni}			
	1	10	20	50
$\|\text{RMSE}^g\|_2$	0.02	0.02	0.02	0.03

Table 4: Example 1. $\|\text{RMSE}^g\|_2$: 20% of non-informative agents

	Method					
	C-RLS	S-RLS	SW-RLS	M-RLS	MW-RLS	ADMM-RLS
$\|\text{RMSE}^g\|_2$	0.02	0.03	0.02	0.07	0.03	0.02



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) θ_2^g vs $\hat{\theta}_2^g$

Figure 6: Example 1. Model parameters vs N_{ni}. Black : true, red : $N_{ni} = 1$, blue : $N_{ni} = 10$, cyan : $N_{ni} = 20$, magenta : $N_{ni} = 50$.

4.3.1 Non-informative agents

Using the previously introduced initial setting and parameters, lets assume that some of the available data sources are non-informative, i.e. some systems are not excited enough to be able to retrieve locally an accurate estimate of all the unknown parameters [17]. Null input sequences and white noise sequences characterized by $R_n = 10^{-8}$ are used to simulate the behavior of the $N_{ni} \leq N$ non-informative agents.

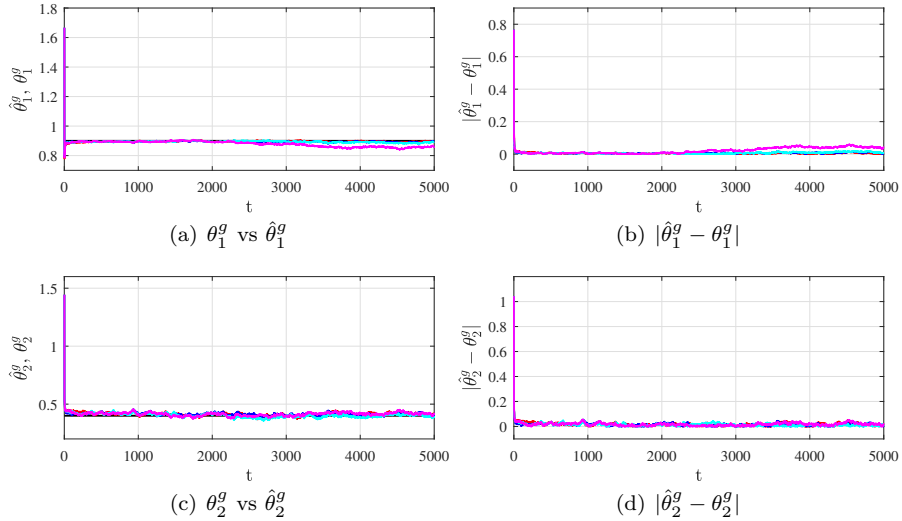
Consider the case $N = 100$ and $T = 5000$. The performance of ADMM-RLS are studied under the hypothesis that an increasing number N_{ni} of systems is non-informative. Looking at the RMSEs in Table 3 and the estimates reported in Figure 6, it can be noticed that the quality of the estimate starts to decrease only when half of the available systems are non-informative. In case of $N_{ni} = 20$, the estimates obtained with ADMM-RLS are then compared with the ones computed with C-RLS and the greedy approaches. As it can be noticed from the RMSEs reported in Table 4, in presence of non-informative agents SW-RLS tends to perform better than the other greedy approaches and the accuracy of the estimates obtained with C-RLS, SW-RLS and ADMM-RLS are comparable.

4.3.2 Agents failure

Consider again $N = 100$ and $T = 5000$ and suppose that, due to a change in the behavior of N_f local agents the parameters of their models suddenly assume different values with respect to $[\,0.9\ 0.4\,]$. We study the performance of ADMM-RLS under the hypothesis that the change in the value of the parameters happens at an unknown instant t_n, randomly chosen in the interval $[1875, 3750]$ samples,

Table 5: Example 1. ADMM-RLS: $\|\text{RMSE}^g\|_2$ vs N_f

	N_f			
	1	10	20	50
$\|\text{RMSE}^g\|_2$	0.03	0.03	0.03	0.04



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) $|\hat{\theta}_1^g - \theta_1^g|$

(c) θ_2^g vs $\hat{\theta}_2^g$ (d) $|\hat{\theta}_2^g - \theta_2^g|$

Figure 7: Example 1. Model parameters vs N_f. Black : true, red : $N_f = 1$, blue : $N_f = 10$, cyan : $N_f = 20$, magenta : $N_f = 50$.

and simulating the change in the local parameters using $\theta_{n,1}$ sampled from the distribution $\mathcal{U}_{[\,0.2\ 0.21\,]}$ and $\theta_{n,2}$ sampled from $\mathcal{U}_{[\,1.4\ 1.43\,]}$ after t_n.
Observe that it might be beneficial to use a non-unitary forgetting factor, due to the change in the local parameters. Consequently, λ_n, $n = 1, \ldots, N$, is set to 0.99 for all the N agents.
The performance of ADMM-RLS are initially assessed considering an increasing number of systems subject to failure. See Table 5 and Figure 7. Observe that the failure of the agents seems not to influence the accuracy of the obtained estimates if $N_f \neq 50$. The use of ADMM-RLS thus allows to compute accurate global estimates even when some of the agent experience a failure.

4.4 Example 2. Time-varying parameters

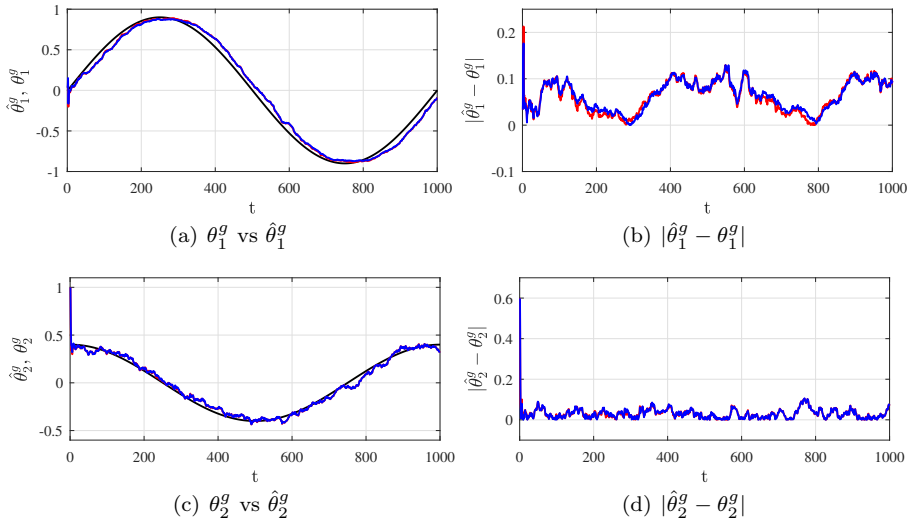
The presence of the forgetting factor in the cost functions f_n (see (20)) allows to estimate time-varying parameters, as it enables to weight differently past and currently collected data.
Suppose that the behavior of N systems is described by the ARX model

$$y_n(t+1) = \theta_1^g(t) y_n(t-1) + \theta_2^g(t) u_n(t-1) + e_n(t) \tag{44}$$

where $\theta_1^g = 0.9 \sin(x)$ and $\theta_2^g = 0.4 \cos(x)$, with $x \in [0, 2\pi]$, and $u_n \sim \mathcal{U}_{[\,2\ 3\,]}$. The white noise sequences $e_n \sim \mathcal{N}(0, R_n)$, $n = 1, \ldots, N$, have covariances R_n randomly selected in the interval $[\,1\ 30\,]$ yielding to $SNR_n \in [\,2.4\ 6.5\,]$ dB.
Considering an estimation horizon $T = 1000$, imposing ϕ_n as $\phi_n(0) = 0.1 I_{n_\theta}$, while $\hat{\theta}_n(0)$ and $\hat{\theta}_{\text{o}}^g$ are sampled from the distributions $\mathcal{N}(\hat{\theta}^g, 2I_{n_\theta})$ and $\mathcal{N}(\hat{\theta}^g, I_{n_\theta})$,

Table 6: Example 2. $\|\text{RMSE}^g\|_2$ vs Method

	Method					
	C-RLS	S-RLS	SW-RLS	M-RLS	MW-RLS	ADMM-RLS
$\|\text{RMSE}^g\|_2$	0.08	0.10	0.08	0.09	0.08	0.08



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) $|\hat{\theta}_1^g - \theta_1^g|$

(c) θ_2^g vs $\hat{\theta}_2^g$ (d) $|\hat{\theta}_2^g - \theta_2^g|$

Figure 8: Example 2. True vs estimated model parameters. Black : true, red : C-RLS, blue : ADMM-RLS.

respectively, $\rho = 0.1$ and setting $\{\lambda_n = \Lambda\}_{n=1}^N$, with $\Lambda = 0.95$, the performances of ADMM-RLS are compared with the ones of C-RLS and the four greedy approaches. See Table 6. As for the case where time-invariant parameters have to be estimated (see Example 1), SW-RLS and MW-RLS tend to perform slightly better than the other greedy approaches. Note that the accuracy of the estimates C-RLS, SW-RLS and MW-RLS is comparable.

Figure 8 reports the actual global parameters and the estimates obtained with C-RLS and ADMM-RLS, along with the estimation errors. As already observed, the accuracy of the estimates computed with C-RLS and ADMM-RLS is comparable.

5 Collaborative estimation for partial consensus

Consider the more general hypothesis that there exist a parameter vector $\theta^g \in \mathbb{R}^{n_g}$, with $n_g \leq n_\theta$ such that:

$$P\theta_n = \theta^g \quad \forall n \in \{1, \dots, N\}, \tag{45}$$

where $P \in \mathbb{R}^{n_g \times n_\theta}$ is a matrix assumed to be known a priori. The problem that we want to solve is then given by

$$\begin{aligned} \min_{\{\theta_n\}_{n=1}^N} \quad & \sum_{n=1}^N f_n(\theta_n) \\ \text{s.t.} \quad & P\theta_n = \theta^g, \quad n = 1, \dots, N, \end{aligned} \tag{46}$$

with f_n defined as in (20). Note that (46) corresponds to (18) with the consensus constraint modified as

$$F(\theta_n) = \theta^g \rightarrow P\theta_n = \theta^g.$$

The considered consensus constraint allows to enforce consensus over a linear combination of the components of θ_n. Note that, through proper choices of P, different settings can be considered, e.g. if $P = I_{n_\theta}$ then $\theta_n = \theta^g$ and thus (46) is equal to (12). We can also enforce consensus only over some components of θ_n, so that some of the unknowns are assumed to be *global* while others are supposed to assume a different value for each agent.
As we are interested in obtaining an estimate for both $\{\theta_n\}_{n=1}^N$ and θ^g, note that (46) cannot be solved resorting to a strategy similar to C-RLS (see Appendix A). In particular, even if properly modified, a method as C-RLS would allow to compute an estimate for the global parameter only.

The ADMM iterations to solve problem (46) are given by

$$\hat{\theta}_n^{(k+1)}(T) = \underset{\theta_n}{\operatorname{argmin}}\ \mathcal{L}(\theta_n, \hat{\theta}^{g,(k)}, \delta_n^{(k)}), \tag{47}$$

$$\hat{\theta}^{g,(k+1)} = \underset{\theta^g}{\operatorname{argmin}}\ \mathcal{L}(\{\hat{\theta}_n^{(k+1)}(T)\}_{n=1}^N, \theta^g, \{\delta_n^{(k)}\}_{n=1}^N), \tag{48}$$

$$\delta_n^{(k+1)} = \delta_n^{(k)} + \rho(P\hat{\theta}_n^{(k+1)}(T) - \hat{\theta}^{g,(k+1)}), \tag{49}$$

with $k \in \mathbb{N}$ indicating the ADMM iteration, $\rho \in \mathbb{R}^+$ being a tunable parameter, $\delta_n \in \mathbb{R}^{n_g}$ representing the Lagrange multiplier and the augmented Lagrangian $\mathcal{L}$ given by

$$\mathcal{L} = \sum_{n=1}^{N} \left\{ f_n(\theta_n) + \delta_n'(P\theta_n - \theta^g) + \frac{\rho}{2} \left\| P\theta_n - \theta^g \right\|_2^2 \right\}. \tag{50}$$

Note that the dependence on T is explicitly indicated only for the local estimates $\hat{\theta}_n$, as they are the only quantities directly affected by the measurement and the regressor at T.
Consider the update of the estimate $\hat{\theta}^g$. The closed form solution for (48) is

$$\hat{\theta}^{g,(k+1)} = \frac{1}{N} \sum_{n=1}^{N} \left(P\hat{\theta}_n^{(k+1)}(T) + \frac{1}{\rho}\delta_n^{(k)} \right). \tag{51}$$

The estimate of the global parameter is thus updated through the combination of the mean of $\{\delta_n\}_{n=1}^N$ and the mean of $\{P\hat{\theta}_n^{(k+1)}(T)\}_{n=1}^N$. As expected, (51) resembles (16), where the local estimates are replaced by a linear combination of their components.
Consider the update for the estimate of the local parameters. The close form solution for (47) is given by:

$$\hat{\theta}_n^{(k+1)}(T) = \phi_n(T) \left\{ \mathcal{Y}_n(T) + P'(\rho\hat{\theta}^{g,(k)} - \delta_n^{(k)}) \right\}, \tag{52}$$

$$\mathcal{Y}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau) y_n(\tau), \quad t = 1, \ldots, T, \tag{53}$$

$$\phi_n(t) = \left(\left[\sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau) X_n(\tau)' \right] + \rho P'P \right)^{-1}, \quad t = 1, \ldots, T. \tag{54}$$

As also in this case we are interested in obtaining recursive formulas for the local updates, consider $\hat{\theta}_n(T-1)$, defined as

$$\hat{\theta}_n(T-1) = \phi_n(T-1)\left(\mathcal{Y}_n(T-1) + P'(\rho\hat{\theta}^g(T-1) - \delta_n(T-1))\right), \quad (55)$$

where $\phi_n(T-1)$ is equal to (54), and $\hat{\theta}^g(T-1)$ and $\delta_n(T-1)$ are the global estimate and the Lagrange multiplier obtained at $T-1$, respectively.
Observe that the following equalities hold

$$\begin{aligned}\phi_n(T) &= \left(\mathcal{X}_n(T) + \rho P'P\right)^{-1} = \\ &= \left(\lambda_n\mathcal{X}_n(T-1) + X_n(T)X_n(T)' + \rho P'P\right)^{-1} = \\ &= \left(\lambda_n\left(\mathcal{X}_n(T-1) + \rho P'P\right) + X_n(T)X_n(T)' + \rho(1-\lambda_n)P'P\right)^{-1} = \\ &= \left(\lambda_n\phi_n(T-1)^{-1} + X_n(T)X_n(T)' + \rho(1-\lambda_n)P'P\right)^{-1},\end{aligned}$$

with

$$\mathcal{X}_n(t) = \sum_{\tau=1}^{t}\lambda_n^{t-\tau}X_n(\tau)(X_n(\tau))', \quad t = 1,\ldots,T.$$

Introducing the extended regressor

$$\tilde{X}_n(T) = \begin{bmatrix} X_n(T) & \sqrt{\rho(1-\lambda_n)}P'\end{bmatrix} \in \mathbb{R}^{n_\theta\times(n_y+n_g)} \quad (56)$$

and applying the matrix inversion lemma, it can be proven that ϕ_n can be updated as

$$\mathcal{R}_n(T) = \lambda_n I_{(n_y+n_g)} + (\tilde{X}_n(T))'\phi_n(T-1)\tilde{X}_n(T), \quad (57)$$

$$K_n(T) = \phi_n(T-1)\tilde{X}_n(T)\left(\mathcal{R}_n(T)\right)^{-1}, \quad (58)$$

$$\phi_n(T) = \lambda_n^{-1}(I_{n_\theta} - K_n(T)(\tilde{X}_n(T))')\phi_n(T-1). \quad (59)$$

Note that (57)-(59) are similar to (31)-(33), with differences due to the new definition of the extended regressor.

Consider again (52). Adding and subtracting

$$\lambda_n\phi_n(T)P'\left(\rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right)$$

to (52), $\hat{\theta}_n^{(k+1)}$ can be computed as

$$\begin{aligned}\hat{\theta}_n^{(k+1)}(T) &= \phi_n(T)\Big[\lambda_n\left(\mathcal{Y}_n(T-1) + P'(\rho\hat{\theta}^g(T-1) - \delta_n(T-1)\right)) + \\ &+X_n(T)y_n(T) - P'\left(\delta_n^{(k)} - \lambda_n\delta_n(T-1)\right) + P'\rho\left(\hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1)\right)\Big] = \\ &= \hat{\theta}_n^{RLS}(T) + \hat{\theta}_n^{ADMM,(k+1)}(T). \quad (60)\end{aligned}$$

In particular,

$$\begin{aligned}\hat{\theta}_n^{RLS}(T) = \phi_n(T)\lambda_n\left\{\mathcal{Y}_n(T-1) + \rho P'\hat{\theta}(T-1) - P'\delta_n(T-1)\right\} + \\ + \phi_n(T)\tilde{X}_n(T)y_n(T), \quad (61)\end{aligned}$$

and

$$\hat{\theta}_n^{ADMM,(k+1)}(T) = \phi_n(T)P' \left(\rho\Delta_{g,\lambda_n}^{(k+1)}(T) - \Delta_{\lambda_n}^{(k+1)}\right), \tag{62}$$

with

$$\Delta_{g,\lambda_n}^{k+1}(T) = \hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1),$$
$$\Delta_{\lambda_n}^{(k+1)}(T) = \delta_n^{(k)} - \lambda_n\delta_n(T-1).$$

Observe that, as for (16) and (51), (62) differs from (36) because of the presence of P.

Note that, accounting for the definition of $\phi_n(T-1)$, exploiting the equality $K_n(T) = \phi_n(T)\tilde{X}_n(T)$ (see Section 4 for the proof) and introducing the extended measurement vector

$$\tilde{y}_n(T) = \begin{bmatrix} y_n(T)' & O_{1\times n_g} \end{bmatrix}',$$

the formula to update $\hat{\theta}_n^{RLS}$ in (62) can be further simplified as

$$\begin{aligned}
\hat{\theta}_n^{RLS}(T) &= \phi_n(T-1)\left\{\left(\mathcal{Y}_n(T-1) + P'(\rho\hat{\theta}^g(T-1) - \delta_n(T-1))\right)\right\} + \\
&- K_n(T)(\tilde{X}_n(T)')\phi_n(T-1)\left\{\left(\mathcal{Y}_n(T-1) + P'(\rho\hat{\theta}^g(T-1) - \delta_n(T-1))\right)\right\} + \\
&+ \phi_n(T)X_n(T)y_n(T) = \\
&= \hat{\theta}_n(T-1) - K_n(T)(\tilde{X}_n(T))'\hat{\theta}_n(T-1) + \phi_n(T)\tilde{X}_n(T)\tilde{y}_n(T) = \\
&= \hat{\theta}_n(T-1) + K_n(T)(\tilde{y}_n(T) - (\tilde{X}_n(T))'\hat{\theta}_n(T-1)).
\end{aligned} \tag{63}$$

As the method tailored to attain full consensus (see Section 4), note that both $\hat{\theta}^g$ and δ_n should be updated on the "cloud". As a consequence, also $\hat{\theta}_n^{ADMM}$ should be updated on the "cloud", due to its dependence on both $\hat{\theta}^g$ and δ_n. On the other hand, $\hat{\theta}_n^{RLS}$ can be updated by the local processors. As for the case considered in Section 4, note that (63) is independent from k and, consequently, the synchronization between the local clock and the one on the "cloud"is not required.

The approach is outlined in Algorithm 4 and the transmissions characterizing each iteration is still the one reported in the scheme in Figure 3. As a consequence, the observations made in Section 4 with respect to the information exchange between the nodes and the "cloud" hold also in this case.

5.1 Example 3

Assume to collect data for $T = 1000$ from a set of $N = 100$ dynamical systems modelled as

$$y_n(t) = \theta_1^g y_n(t-1) + \theta_{n,2}y_n(t-2) + \theta_2^g u_n(t-1) + e_n(t), \tag{64}$$

where $\theta^g = \begin{bmatrix} 0.2 & 0.8 \end{bmatrix}'$ and $\theta_{n,2}$ is sampled from a normal distribution $\mathcal{N}(0.4, 0.0025)$, so that it is different for the N systems. The white noise sequence $e_n \sim \mathcal{N}(0, R_n)$, where, for the "informative' systems, $R_n \in [1\ 20]$ yields $SNR \in$

Algorithm 4 ADMM-RLS for partial consensus (N2C2N)

Input: Sequence of observations $\{X_n(t), y_n(t)\}_{t=1}^{T}$, initial matrices $\phi_n(0) \in \mathbb{R}^{n_\theta \times n_\theta}$, initial local estimates $\hat{\theta}_n(0)$, initial dual variables $\delta_{n,\text{o}}$, forgetting factors λ_n, $n = 1, \ldots, N$, initial global estimate $\hat{\theta}_{\text{o}}^{g}$, parameter $\rho \in \mathbb{R}^{+}$.

1. **for** $t = 1, \ldots, T$ **do**

   Local

   1.1. **for** $n = 1, \ldots, N$ **do**
   1.1.1. **compute** $\tilde{X}_n(t)$ with (56);
   1.1.2. **compute** $K_n(t)$ and $\phi_n(t)$ with (58) - (59);
   1.1.3. **compute** $\hat{\theta}_n^{RLS}(t)$ with (63);
   1.2. **end for**;

   Global

   1.1. **do**
   1.1.1. **compute** $\hat{\theta}_n^{ADMM,(k+1)}(t)$ with (62), $n = 1, \ldots, N$;
   1.1.2. **compute** $\hat{\theta}_n^{(k+1)}(t)$ with (60), $n = 1, \ldots, N$;
   1.1.3. **compute** $\hat{\theta}^{g,(k+1)}$ with (51);
   1.1.4. **compute** $\delta_n^{(k+1)}$ with (49), $n = 1, \ldots, N$;
   1.2. **until** a stopping criteria is satisfied (e.g. maximum number of iterations attained);

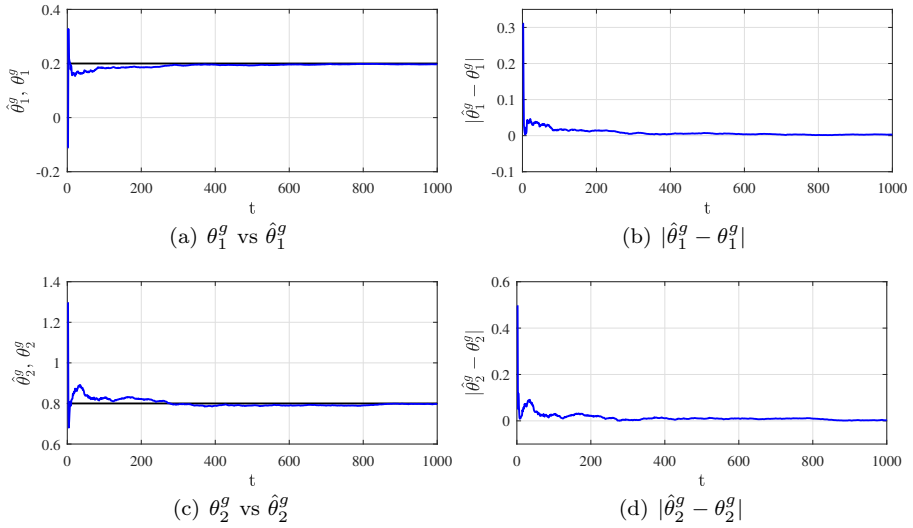
2. **end**.

Output: Estimated global parameters $\{\hat{\theta}^g(t)\}_{t=1}^{T}$, estimated local parameters $\{\hat{\theta}_n(t)\}_{t=1}^{T}$, $n = 1, \ldots, N$.

$[3.1, 14.6]$ dB (see (42)).
Initializing ϕ_n as $\phi_n(0) = 0.1 I_{n_\theta}$, while $\hat{\theta}_n(0)$ and $\hat{\theta}_{\text{o}}^{g}$ are sampled from the distributions $\mathcal{N}(\hat{\theta}^g, 2I_{n_\theta})$ and $\mathcal{N}(\hat{\theta}^g, I_{n_\theta})$, respectively, $\{\lambda_n = \Lambda\}_{n=1}^{N}$, with $\Lambda = 1$, and $\rho = 0.1$, the performance of the proposed approach are evaluated. Figure 9 shows $\hat{\theta}^g$ obtained with ADMM-RLS, along with the estimation error. Observe that the estimates tends to converge to the actual value of the global parameters. To further assess the performances of ADMM-RLS, θ_n, $\hat{\theta}_n$ and $\hat{\theta}_n^{RLS}$ obtained for the 5th system, i.e. $n = 5$, are compared in Figure 10. It can thus be seen that the difference between $\hat{\theta}_n^{RLS}$ and $\hat{\theta}_n$ is mainly noticeable at the beginning of the estimation horizon, but then $\hat{\theta}_n^{RLS}$ and $\hat{\theta}_n$ are barely distinguishable. Note that $\text{SNR}_5 = 8.9$ dB.

5.1.1 Non-informative agents

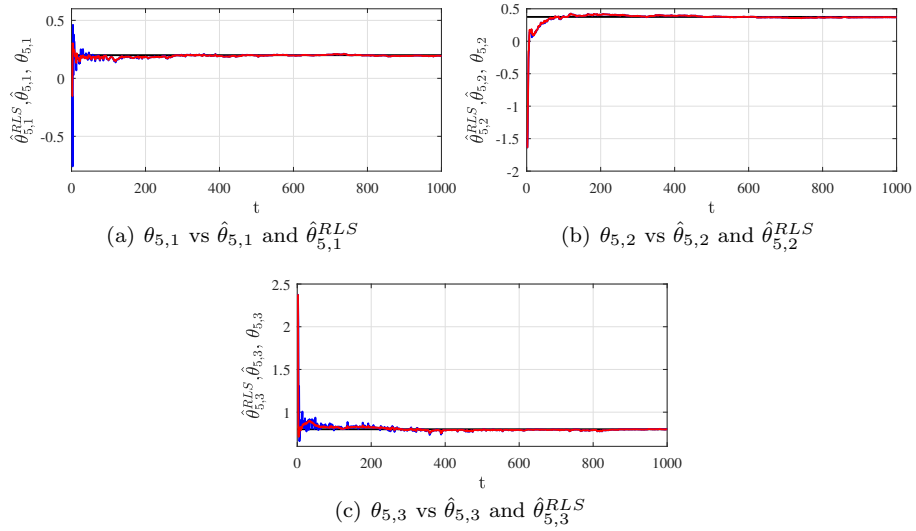
Suppose that among the $N = 100$ systems described by the model in (64), $N_{ni} = 20$ randomly chosen agents are non-informative, i.e. their input sequences u_n are null and $R_n = 10^{-8}$.
As it can be observed from the estimates reported in Figure 11, $\{\hat{\theta}_i^g\}_{i=1}^{2}$ con-



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) $|\hat{\theta}_1^g - \theta_1^g|$

(c) θ_2^g vs $\hat{\theta}_2^g$ (d) $|\hat{\theta}_2^g - \theta_2^g|$

Figure 9: Example 3. True vs estimated global parameters. Black : true, blue : ADMM-RLS.



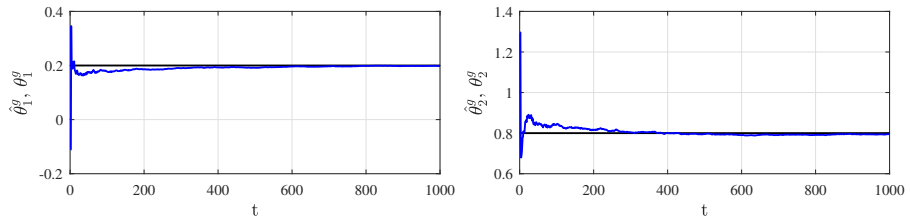
(a) $\theta_{5,1}$ vs $\hat{\theta}_{5,1}$ and $\hat{\theta}_{5,1}^{RLS}$ (b) $\theta_{5,2}$ vs $\hat{\theta}_{5,2}$ and $\hat{\theta}_{5,2}^{RLS}$

(c) $\theta_{5,3}$ vs $\hat{\theta}_{5,3}$ and $\hat{\theta}_{5,3}^{RLS}$

Figure 10: Example 3. Local parameter $\theta_{n,2}$, $n = 5$. Black : true, blue : $\hat{\theta}_5$, red: $\hat{\theta}_5^{RLS}$.

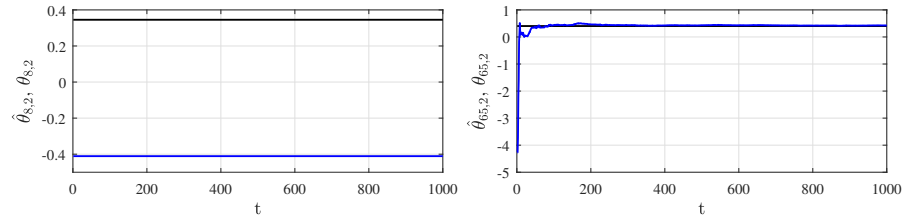
verge to the actual values of the global parameters even if 20% of the systems provide non-informative data.

The local estimates $\hat{\theta}_{n,2}$ for the 8th and 65th system ($SNR_{65} \approx 6$ dB) are reported in Figure 12. As, the 8th system is among the ones with a non exciting input, $\hat{\theta}_{8,2} = \hat{\theta}_{8,2}(0)$ over the estimation horizon. Instead, $\hat{\theta}_{65,2}$ tends to converge to the actual value of $\theta_{65,2}$. Even if the purely local parameter is not retrieved from the data, using the proposed collaborative approach $\theta_{8,1}$ and



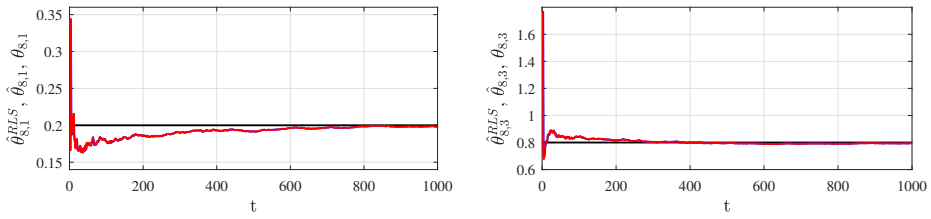
(a) θ_1^g vs $\hat{\theta}_1^g$ (b) θ_2^g vs $\hat{\theta}_2^g$

Figure 11: Example 3. True vs estimated global parameters. Black : true, blue : ADMM-RLS.



(a) $\theta_{8,2}$ vs $\hat{\theta}_{8,2}$ (b) $\theta_{65,2}$ vs $\hat{\theta}_{65,2}$

Figure 12: Example 3. Local parameters $\theta_{n,2}$, $n = 8, 65$. Black : true, blue : ADMM-RLS.



(a) $\theta_{8,1}$ vs $\hat{\theta}_{8,1}$ and $\hat{\theta}_{8,1}^{RLS}$ (b) $\theta_{8,3}$ vs $\hat{\theta}_{8,3}$ and $\hat{\theta}_{8,3}^{RLS}$

Figure 13: Example 3. Local parameters $\theta_{8,i}$, $i = 1, 3$. Black : true, blue : $\hat{\theta}_{8,i}^{RLS}$, red: $\hat{\theta}_{8,i}$.

$\theta_{8,3}$ are accurately estimated (see Figure 13). We can thus conclude that the proposed estimation method "forces" the estimates of the global components of θ_n to follow $\hat{\theta}^g$, which is estimated automatically discarding the contributions from the systems that lacked excitation.

6 Constrained Collaborative estimation for partial consensus

Suppose that the value of the local parameter θ_n is constrained to a set $\mathcal{C}_n$ and that this hypothesis holds for all the agents $n \in \{1, \ldots, N\}$. With the objective of reaching partial consensus among the agents, the problem to be solved can

thus be formulated as

$$\begin{aligned} \text{minimize} \quad & \sum_{n=1}^{N} f_n(\theta_n) \\ \text{s.t.} \quad & P\theta_n = \theta, \quad n = 1, \ldots, N, \\ & \theta_n \in \mathcal{C}_n, \quad n = 1, \ldots, N. \end{aligned} \tag{65}$$

Observe that (65) corresponds to (18) if the nonlinear consensus constraint is replaced with (45).
To use ADMM to solve (65), the problem has to be modified as

$$\begin{aligned} \text{minimize} \quad & \sum_{n=1}^{N} \{f_n(\theta_n) + g_n(z_n)\} \\ \text{s.t.} \quad & P\theta_n = \theta^g \quad n = 1, \ldots, N \\ & \theta_n = z_n, \quad n = 1, \ldots, N \end{aligned} \tag{66}$$

where $\{g_n\}_{n=1}^N$ are the indicator functions of the sets $\{\mathcal{C}_n\}_{n=1}^N$ (defined as in (7)) and $\{z_n \in \mathbb{R}^{n_\theta}\}_{n=1}^N$ are auxiliary variables. Observe that (66) can be solved with ADMM. Given the augmented Lagrangian associated with (66), i.e.

$$\begin{aligned} \mathcal{L} = \sum_{n=1}^{N} \{ & f_n(\theta_n) + g_n(z_n) + \delta'_{n,1}(\theta_n - z_n) + \delta'_{n,2}(P\theta_n - \theta^g) + \\ & + \frac{\rho_1}{2}\|\theta_n - z_n\|_2^2 + \frac{\rho_2}{2}\|P\theta_n - \theta^g\|_2^2\}, \end{aligned} \tag{67}$$

the iterations that have to be performed to solve the addressed problem with ADMM are

$$\hat{\theta}_n^{(k+1)}(T) = \underset{\theta_n}{\operatorname{argmin}}\ \mathcal{L}(\theta_n, \hat{\theta}^{g,(k)}, z_n^{(k)}, \delta_n^{(k)}), \tag{68}$$

$$z_n^{(k+1)} = \underset{z_n}{\operatorname{argmin}}\ \mathcal{L}(\hat{\theta}_{n,(k+1)}(T), \hat{\theta}^{g,(k)}, z_n, \delta_n^{(k)}), \tag{69}$$

$$\hat{\theta}^{g,(k+1)} = \underset{\theta^g}{\operatorname{argmin}}\ \mathcal{L}(\{\hat{\theta}_n^{(k+1)}\}_{n=1}^N, \theta^g, \{z_n^{(k+1)}, \delta_n^{(k)}\}_{n=1}^N), \tag{70}$$

$$\delta_{n,1}^{(k+1)} = \delta_{n,1}^{(k)} + \rho_1(\hat{\theta}_n^{(k+1)}(T) - z^{(k+1)}), \tag{71}$$

$$\delta_{n,2}^{(k+1)} = \delta_{n,2}^{(k)} + \rho_2(P\hat{\theta}_n^{(k+1)}(T) - \hat{\theta}^{g,(k+1)}). \tag{72}$$

Note that two sets of Lagrangian multipliers, $\{\delta_{n,1}\}_{n=1}^N$ and $\{\delta_{n,2}\}_{n=1}^N$, have been introduced. While $\delta_{n,1} \in \mathbb{R}^{n_g}$ is associated with the partial consensus constraint, $\delta_{n,2} \in \mathbb{R}^{n_\theta}$ is related to the constraint $\theta_n \in \mathcal{C}_n$, $n = 1, \ldots, N$.

Solving (69)-(70), the resulting updates for the auxiliary variables and the global estimates are

$$z_n^{(k+1)} = \mathcal{P}_{\mathcal{C}_n}\left(\hat{\theta}_n^{(k+1)}(T) + \frac{1}{\rho_1}\delta_{n,1}^{(k)}\right), \quad n = 1, \ldots, N, \tag{73}$$

$$\hat{\theta}^{g,(k+1)} = \frac{1}{N}\sum_{n=1}^{N}\left(P\hat{\theta}_n^{(k+1)}(T) + \frac{1}{\rho_2}\delta_{n,2}^{(k)}\right). \tag{74}$$

Observe that z-update is performed projecting onto the set $\mathcal{C}_n$ a combination of the updated local estimate and $\delta_{n,1}^{(k)}$, while $\hat{\theta}^{g,(k+1)}$ is computed as in Section 5, with δ_n replaced by $\delta_{n,2}$.

Consider the close form solution of (68), which is given by

$$\hat{\theta}_n^{(k+1)}(T) = \phi_n(T)\left\{\mathcal{Y}_n(T) - \delta_{n,1}^{(k)} - P'\delta_{n,2}^{(k)} + \rho_1 z_n^{(k)} + \rho_2 P'\hat{\theta}^{g,(k)}\right\}, \tag{75}$$

$$\mathcal{Y}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau) y_n(\tau), \tag{76}$$

$$\phi_n(t) = \left(\left[\sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau) X_n(\tau)'\right] + \rho_1 I_{n_\theta} + \rho_2 P'P\right)^{-1}. \tag{77}$$

Aiming at finding recursive formulas to update the estimates of the local parameters, we introduce the nth local estimate obtained at $T-1$, i.e.

$$\begin{aligned}\hat{\theta}_n(T-1) = \phi_n(T-1)\{&\mathcal{Y}_n(T-1) - \delta_{n,1}(T-1) - P'\delta_{n,2}(T-1) + \\ &+\rho_1 z_n(T-1) + \rho_2 P'\hat{\theta}^g(T-1)\Big\}\end{aligned} \tag{78}$$

with $\delta_{n,1}(T-1)$, $\delta_{n,2}(T-1)$, $z_n(T-1)$ and $\hat{\theta}^g(T-1)$ being the Lagrange multipliers and the global estimate obtained at $T-1$, respectively.

To obtain recursive formulas to compute $\hat{\theta}_n^{(k+1)}$, we start proving that $\phi_n(T)$ can be computed as a function of $\phi_n(T-1)$. in particular, introducing

$$\mathcal{X}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n(\tau)(X_n(\tau))',$$

note that

$$\begin{aligned}&\phi_n(T)^{-1} = \mathcal{X}_n(T) + \rho_1 I_{n_\theta} + \rho_2 P'P = \\ &= \lambda_n \mathcal{X}_n(T-1) + X_n(T)X_n(T)' + \rho_1 I_{n_\theta} + \rho_2 P'P = \\ &= \lambda_n\left[\mathcal{X}_n(T-1) + \rho_1 I_{n_\theta} + \rho_2 P'P\right] + X_n(T)X_n(T)' + (1-\lambda_n)\rho_1 + (1-\lambda_n)\rho_2 P'P = \\ &= \lambda_n \phi_n(T-1)^{-1} + X_n(T)X_n(T)' + (1-\lambda_n)\rho_1 + (1-\lambda_n)\rho_2 P'P.\end{aligned}$$

Defining the extended regressor as

$$\tilde{X}_n(T) = \begin{bmatrix} X_n(T) & \sqrt{(1-\lambda_n)\rho_1} I_{n_\theta} & \sqrt{(1-\lambda_n)\rho_2} P' \end{bmatrix} \in \mathbb{R}^{n_\theta \times (n_y + n_\theta + n_g)}, \tag{79}$$

and applying the matrix inversion lemma, it can be easily proven that $\phi_n(T)$ can then be computed as:

$$\mathcal{R}_n(T) = \lambda_n I_{(n_y+n_\theta+n_g)} + \tilde{X}_n(T)'\phi_n(T)\tilde{X}_n(T), \tag{80}$$

$$K_n(T) = \phi_n(T-1)\tilde{X}_n(T)(\mathcal{R}_n(T))^{-1}, \tag{81}$$

$$\phi_n(T) = \lambda_n^{-1}(I_{n_\theta} - K_n(T)\tilde{X}_n(T)')\phi_n(T-1). \tag{82}$$

The same observations relative to the update of ϕ_n made in Section 5 holds also in the considered case.

Consider (75). Adding and subtracting

$$\lambda_n \left[-\delta_{n,1}(T-1) - P'\delta_{n,2}(T-1) + \rho_1 z_n(T-1) + \rho_2 P'\hat{\theta}^g(T-1) \right]$$

to (75) and considering the definition of $\phi_n(T-1)$ (see (77)), the formula to update $\hat{\theta}_n$ can be further simplified as

$$\begin{aligned}
&\hat{\theta}_n^{(k+1)}(T) = \phi_n(T)\{\lambda_n \big(\mathcal{Y}_n(T-1) - \delta_{n,1}(T-1) - P'\delta_{n,2}(T-1) + \\
&+\rho_1 z_n(T-1) + \rho_2 P'\hat{\theta}^g(T-1)\big) + X_n(T)y_n(T) + \rho_1(z_n^{(k)} - \lambda_n z_n(T-1)) + \\
&+ \rho_2 P'(\hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1)) - (\delta_{n,1}^{(k)} - \lambda_n\delta_{n,1}(T-1)) + \\
&- P'(\delta_{n,2}^{(k)} - \lambda_n\delta_{n,2}(T-1))\} = \\
&= \hat{\theta}_n(T-1) - K_n(T)\tilde{X}_n(T)\hat{\theta}_n(T-1) + \phi_n(T)\{X_n(T)y_n(T) + \\
&+ \rho_1(z_n^{(k)} - \lambda_n z_n(T-1)) + \rho_2 P'(\hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1))\} + \\
&- (\delta_{n,1}^{(k)} - \lambda_n\delta_{n,1}(T-1)) - P'(\delta_{n,2}^{(k)} - \lambda_n\delta_{n,2}(T-1))\} = \\
&= \hat{\theta}_n^{RLS}(T) + \hat{\theta}_n^{ADMM,(k+1)}(T).
\end{aligned} \tag{83}$$

In particular,

$$\begin{aligned}
\hat{\theta}_n^{RLS} = \phi_n(T)\lambda_n \big(&\mathcal{Y}_n(T-1) - \delta_{n,1}(T-1) - P'\delta_{n,2}(T-1) + \rho_1 z_n(T-1) + \\
&+\rho_2 P'\hat{\theta}^g(T-1)\big) + \phi_n(T)X_n(T)y_n(T),
\end{aligned} \tag{84}$$

while

$$\hat{\theta}_n^{ADMM,(k+1)}(T) = \phi_n(T)\left[\rho_1\Delta_{z,\lambda_n}^{(k+1)}(T) + \rho_2 P'\Delta_{g,\lambda_n}^{(k+1)}(T) - \Delta_{1,\lambda_n}^{(k+1)} - P'\Delta_{2,\lambda_n}^{(k+1)}\right]. \tag{85}$$

with

$$\begin{aligned}
\Delta_{z,\lambda_n}^{(k+1)}(T) &= z_n^{(k)} - \lambda_n z_n(T-1), \\
\Delta_{g,\lambda_n}^{(k+1)}(T) &= \hat{\theta}^{g,(k)} - \lambda_n\hat{\theta}^g(T-1), \\
\Delta_{1,\lambda_n}^{(k+1)} &= \delta_{n,1}^{(k)} - \lambda_n\delta_{n,1}(T-1), \\
\Delta_{2,\lambda_n}^{(k+1)} &= \delta_{n,2}^{(k)} - \lambda_n\delta_{n,2}(T-1).
\end{aligned}$$

Note that (85) differs from (62) because of the introduction of the additional terms Δ_{z,λ_n} and Δ_{1,λ_n}.
Similarly to what is presented in Section 5, thanks to (82) the formula to update $\hat{\theta}_n^{RLS}$ can be further reduced as

$$\begin{aligned}
\hat{\theta}_n^{RLS} &= \hat{\theta}_n(T-1) - K_n(T)(\tilde{X}_n(T))'\hat{\theta}_n(T-1) + \phi_n(T)X_n(T)y_n(T) = \\
&= \hat{\theta}_n(T-1) - K_n(T)(\tilde{X}_n(T))'\hat{\theta}_n(T-1) + \phi_n(T)\tilde{X}_n(T)\tilde{y}_n(T),
\end{aligned}$$

with the extended measurement vector $\tilde{y}_n(T)$ is defined as

$$\tilde{y}_n(T) = \begin{bmatrix} y_n(T)' & O_{1\times n_\theta} & O_{1\times n_{n_g}} \end{bmatrix}'.$$

Exploiting the equality $K_n(T) = \phi_n(T)\tilde{X}_n(T)$ (the proof can be found in (4)), it can thus be proven that

$$\hat{\theta}_n^{RLS} = hat\theta_n(T-1) + K_n(T)(\tilde{y}_n(T) - (\tilde{X}_n(T))'\hat{\theta}_n(T-1)). \tag{86}$$

It is worth remarking that $\hat{\theta}_n^{RLS}$ can be updated (i) locally, (ii) recursively and (iii) once per step t.

Remark 5 *The proposed method, summarized in Algorithm 5 and in Figure 3, requires the agents to transmit $\{\hat{\theta}_n^{RLS}, \phi_n\}$ to the "cloud", while the "cloud" has to communicate $\hat{\theta}_n$ to each node once it has been computed. As a consequence, a N2C2N transmission scheme is required.* ■

Algorithm 5 ADMM-RLS algorithm for constrained consensus

Input: Sequence of observations $\{X_n(t), y_n(t)\}_{t=1}^T$, initial matrices $\phi_n(0) \in \mathbb{R}^{n_\theta \times n_\theta}$, initial local estimates $\hat{\theta}_n(0)$, initial dual variables $\delta_{n,1}^{\text{o}}$ and $\delta_{n,2}^{\text{o}}$, initial auxiliary variables $\hat{z}_{n,\text{o}}$, forgetting factors λ_n, $n = 1, \ldots, N$, initial global estimate $\hat{\theta}_{\text{o}}^g$, parameters $\rho_1, \rho_2 \in \mathbb{R}^+$.

1. **for** $t = 1, \ldots, T$ **do**

   Local

   1.1. **for** $n = 1, \ldots, N$ **do**
   1.1.1. **compute** $\tilde{X}_n(t)$ with (79);
   1.1.2. **compute** $K_n(t)$ and $\phi_n(t)$ with (81) - (82);
   1.1.3. **compute** $\hat{\theta}_n^{RLS}(t)$ with (86);
   1.2. **end for**;

   Global

   1.1. **do**
   1.1.1. **compute** $\hat{\theta}_n^{ADMM,(k+1)}(t)$ with (85), $n = 1, \ldots, N$;
   1.1.2. **compute** $\hat{\theta}_n^{(k+1)}(t)$ with (83), $n = 1, \ldots, N$;
   1.1.3. **compute** $z_n^{(k+1)}(t)$ with (73), $n = 1, \ldots, N$;
   1.1.4. **compute** $\hat{\theta}^{g,(k+1)}$ with (74);
   1.1.5. **compute** $\delta_{n,1}^{(k+1)}$ with (71), $n = 1, \ldots, N$;
   1.1.6. **compute** $\delta_{n,2}^{(k+1)}$ with (72), $n = 1, \ldots, N$;
   1.2. **until** a stopping criteria is satisfied (e.g. maximum number of iterations attained);

2. **end**.

Output: Estimated global parameters $\{\hat{\theta}^g(t)\}_{t=1}^T$, estimated local parameters $\{\hat{\theta}_n(t)\}_{t=1}^T$, $n = 1, \ldots, N$.

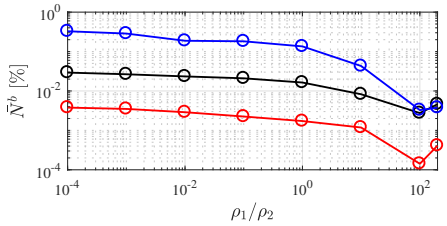


Figure 14: Example 4. $\bar{N}^b$ vs ρ_1/ρ_2: black = $\bar{N}_1^b$, red = $\bar{N}_2^b$, blue = $\bar{N}_3^b$.

6.1 Example 4

Suppose that the data are gathered from $N = 100$ systems, described by (64) and collected over an estimation horizon $T = 5000$. Moreover, assume that the a priori information constraints parameter estimates to the following ranges:

$$\begin{aligned} \ell_{n,1} \leq \hat{\theta}_{n,1} \leq up_{n,1} \quad \ell_{n,2} \leq \hat{\theta}_{n,2} \leq up_{n,2} \\ \ell_{n,3} \leq \hat{\theta}_{n,3} \leq up_{n,3}. \end{aligned} \tag{87}$$

Observe that the parameters $\rho_1, \rho_2 \in \mathbb{R}^+$ have to be tuned. To assess how the choice of these two parameters affects the satisfaction of (87), consider the number of steps the local estimates violate the constraints over the estimation horizon T, $\{N_i^b\}_{i=1}^3$. Assuming that "negligible" violations of the constraints are allowed, (87) are supposed to be violated if the estimated parameters fall outside the interval $\mathcal{B}_n = [\,\ell_n - 10^{-4} \;\; u_n + 10^{-4}\,]$. Considering the set of constraints

$$\mathcal{S}_2 = \{\ell_n = [\,0.19 \;\; \theta_{n,2} - 0.1 \;\; 0.79\,], up_n = [\,0.21 \;\; \theta_{n,2} + 0.1 \;\; 0.81\,]\},$$

Figure 14 shows the average percentage of violations over the N agents obtained fixing $\rho_2 = 0.1$ and choosing

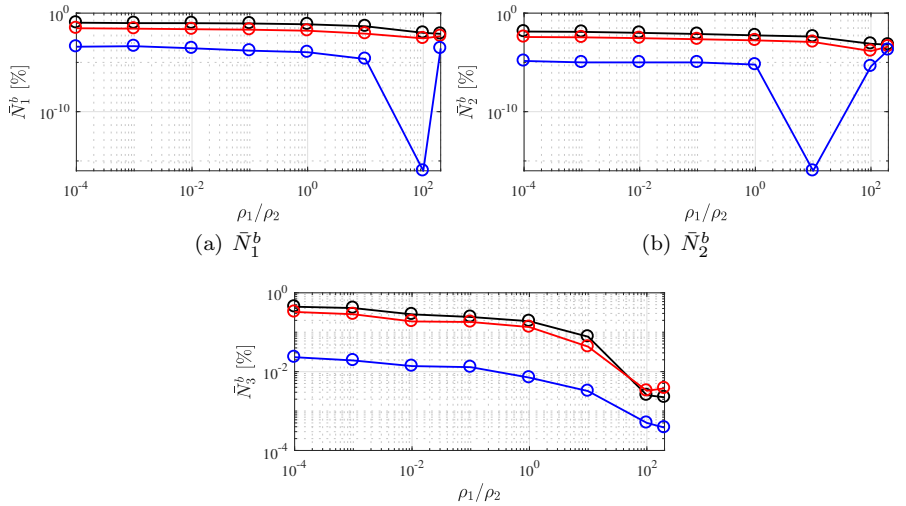
$$\rho_1 = \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}.10^{-1}, 1, 10, 20\}.$$

Observe that if ρ_1 dominates over ρ_2 the number of violations tends to decrease, as in the augmented Lagrangian (87) are weighted more than the consensus constraint. However, if $\rho_1/\rho_2 > 100$, $\{\bar{N}_i^b\}_{i=1}^3$ tend to slightly increase. It is thus important to trade-off between the weights attributed to (87) and the consensus constraint. To evaluate how the stiffness of the constraints affects the choice of the parameters, $\{N_i^b\}_{i=1}^3$ are computed considering three different sets of box constraints

$$\begin{aligned} \mathcal{S}_1 &= \{\ell_n = [\,0.195 \;\; \theta_{n,2} - 0.05 \;\; 0.795\,], up_n = [\,0.205 \;\; \theta_{n,2} + 0.05 \;\; 0.805\,]\}, \\ \mathcal{S}_2 &= \{\ell_n = [\,0.19 \;\; \theta_{n,2} - 0.1 \;\; 0.79\,], up_n = [\,0.21 \;\; \theta_{n,2} + 0.1 \;\; 0.81\,]\}, \\ \mathcal{S}_3 &= \{\ell_n = [\,0.15 \;\; \theta_{n,2} - 0.5 \;\; 0.75\,], up_n = [\,0.25 \;\; \theta_{n,2} + 0.5 \;\; 0.85\,]\}. \end{aligned}$$

The resulting $\{\bar{N}_i^b\}_{i=1}^3$ are reported in Figure 15. Note that also in this case the higher the ratio ρ_1/ρ_2 is, the smaller $\{\bar{N}_i^b\}_{i=1}^3$ are. However, also in this case, the constraint violations tend to increase for $\rho_1/\rho_2 > 100$.

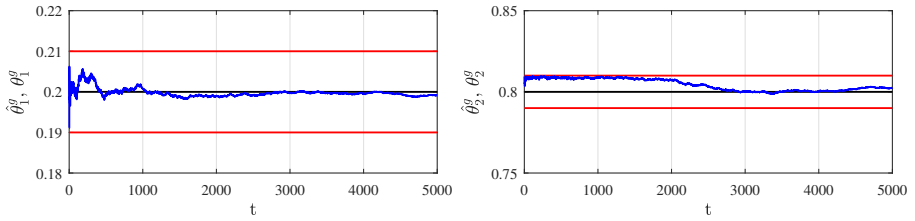
Focusing on the assessment of ADMM-RLS performances when the set of constraints is $\mathcal{S}_2$, Figure 16 shows the global estimates obtained using the same



(a) $\bar{N}_1^b$ (b) $\bar{N}_2^b$

(c) $\bar{N}_3^b$

Figure 15: Example 4. Average percentage of constraint violations $\bar{N}_i^b$ %, $i = 1, 2, 3$, vs ρ_1/ρ_2. Black : $\mathcal{S}_1$, red : $\mathcal{S}_2$, blue : $\mathcal{S}_3$.



(a) θ_1^g vs $\hat{\theta}_1^g$ (b) θ_2^g vs $\hat{\theta}_2^g$

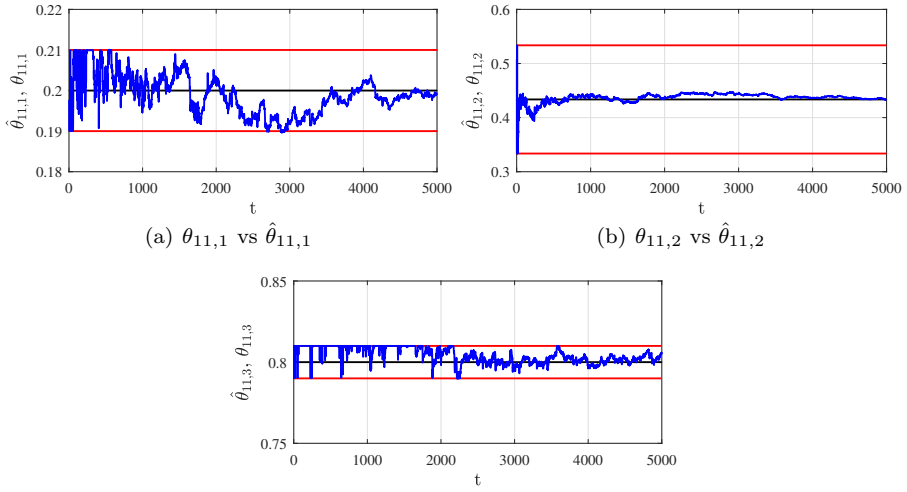
Figure 16: Example 4.Global model parameters: black = true, blue = ADMM-RLS, red = upper and lower bounds.

initial conditions and forgetting factors as in Section 6, with $\rho_1 = 10$ and $\rho_2 = 0.1$. Note that the global estimates satisfy (87), showing that the constraints on the global estimate are automatically enforced imposing $\theta_n \in \mathcal{C}_n$. As it concerns the RMSEs for $\hat{\theta}^g$ (43), they are equal to:

$$RMSE_1^g = 0.001 \text{ and } RMSE_2^g = 0.006,$$

and their relatively small values can be related to the introduction of the additional constraints, that allow to limit the resulting estimation error.

Figure 17 show the estimate $\hat{\theta}_n$ for $n = 11$, with $SNR_{11} = 10.6$ dB. Note that the estimated parameters tend to satisfy the constraints. In Figure 18 $\hat{\theta}_n$ and $\hat{\theta}_n^{RLS}$, with $n = 11$, are compared. As it can be noticed, while $\hat{\theta}_{11}$ satisfied the imposed constraints on its values, the effect of using $\hat{\theta}_{11}$ to update $\hat{\theta}_{11}^{RLS}$ (see (86)) is not strong enough to enfoce also the estimates computed locally to satisfy the contraints. To further assess the performance of the proposed

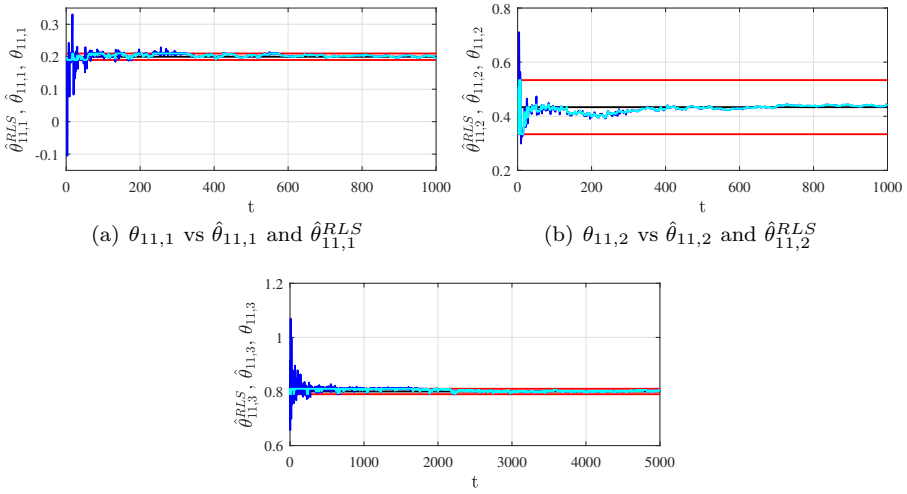


(a) $\theta_{11,1}$ vs $\hat{\theta}_{11,1}$

(b) $\theta_{11,2}$ vs $\hat{\theta}_{11,2}$

(c) $\theta_{11,3}$ vs $\hat{\theta}_{11,3}$

Figure 17: Local parameter θ_n, $n = 11$. Black : true, blue : ADMM-RLS, red : upper and lower bounds



(a) $\theta_{11,1}$ vs $\hat{\theta}_{11,1}$ and $\hat{\theta}_{11,1}^{RLS}$

(b) $\theta_{11,2}$ vs $\hat{\theta}_{11,2}$ and $\hat{\theta}_{11,2}^{RLS}$

(c) $\theta_{11,3}$ vs $\hat{\theta}_{11,3}$ and $\hat{\theta}_{11,3}^{RLS}$

Figure 18: Local parameter θ_n, $n = 11$, for $t \in [1\ 1000]$. Black : true, blue : $\hat{\theta}_{11}^{RLS}$, cyan : $\hat{\theta}_{11}$, red : upper and lower bounds

approach, the RMSE for the local estimates

$$\text{RMSE}_{n,i} = \sqrt{\frac{\sum_{t=1}^{T} \left(\theta_{n,i} - \hat{\theta}_{n,i}(t)\right)^2}{T}}. \tag{88}$$

is also considered. $RMSE_{n,2}$ obtained for each of the N systems is reported in Figure 19 and, as it can be noticed, $RMSE_{n,2}$ is relatively small. As for the global parameters' estimates, this result can be related to the introduction of

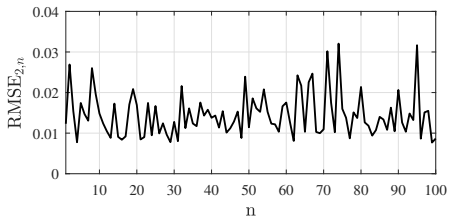


Figure 19: $RMSE_2$ for each agent n, $n = 1, \ldots, N$.

the additional constraints.

7 Concluding Remarks and Future Work

In this report a method for collaborative least-squares parameter estimation is presented based on output measurements from multiple systems which can perform local computations and are also connected to a centralized resource in the "cloud". The approach includes two stages: (i) a *local* step, where estimates of the unkown parameters are obtained using the locally available data, and (ii) a *global* stage, performed on the cloud, where the local estimates are fused.
Future research will address extentions of the method to the nonlinear and multi-class consensus cases. Moreover, an alternative solution of the problem will be studied so to replace the transmission policy required now, i.e. N2C2N, with a Node-to-Cloud (N2C) communication scheme. This change should allow to alleviate problems associated with the communication latency between the cloud and the nodes. Moreover, it should enable to obtain local estimators that run independently from the data transmitted by the cloud, and not requiring synchronous processing by the nodes and "cloud". Other, solutions to further reduce the trasmission complexity and to obtain an asynchronous scheme with the same characteristics as the one presented in this report will be investigated.

A Centralized RLS

Consider problem (12), with the cost functions given by

$$f_n(\theta_n) = \frac{1}{2} \sum_{t=1}^{T} \|y_n(t) - (X_n(t))'\theta_n\|_2^2.$$

The addressed problem can be solved in a fully centralized fashion, if at each step t all the agents transmit the collected data pairs $\{y_n(t), X_n(t)\}$, $n = 1, \ldots, N$, to the "cloud". This allows the creation of the lumped measurement vector and regressor, given by

$$\begin{aligned} \check{y}(t) &= \begin{bmatrix} y_1(t)' & \ldots & y_N(t)' \end{bmatrix}' \in \mathbb{R}^{N \cdot n_y \times 1}, \\ \check{X}(t) &= \begin{bmatrix} X_1(t)' & \ldots & X_N(t)' \end{bmatrix}' \in \mathbb{R}^{n_\theta \times n_y \cdot N}. \end{aligned} \tag{89}$$

Through the introduction of the lumped vectors, (12) with f_n as in (20) is equivalent to

$$\min_{\theta^g} \frac{1}{2} \sum_{t=1}^{T} \left\| \check{y}(t) - (\check{X}(t))' \theta^g \right\|_2^2 . \tag{90}$$

The estimate for the unknown parameters $\hat{\theta}^g$ can thus be retrieved applying standard RLS (see [17]), i.e. performing at each step t the following iterations

$$\mathcal{K}(t) = \phi(t-1)\check{X}(t) \left(I_{\mathscr{D}} + (\check{X}(t))' \phi(t-1) \check{X}(t) \right)^{-1}, \tag{91}$$

$$\phi(t) = \left(I_{n_\theta} - \mathcal{K}(t)(\check{X}(t))' \right) \phi(t-1), \tag{92}$$

$$\hat{\theta}^g(t) = \hat{\theta}^g(t-1) + \mathcal{K}(t) \left(\check{y}(t) - (\check{X}(t))' \hat{\theta}^g(t-1) \right), \tag{93}$$

with $\mathscr{D} = N \cdot n_y \times 1$.

References

[1] R. Arablouei, K. DoğanÃ§ay, S. Werner, and Y. F. Huang. Adaptive distributed estimation based on recursive least-squares and partial diffusion. *IEEE Transactions on Signal Processing*, 62(14):3510–3522, July 2014.

[2] M. Benning, F. Knoll, C. Schönlieb, and T. Valkonen. Preconditioned admm with nonlinear operator constraint, 2015. https://arxiv.org/abs/1511.00425.

[3] F. Boem, Y. Xu, C. Fischione, and T. Parisini. A distributed estimation method for sensor networks based on pareto optimization. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 775–781, Dec 2012.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, January 2011.

[5] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed. Diffusion recursive least-squares for distributed estimation over adaptive networks. *IEEE Transactions on Signal Processing*, 56(5):1865–1877, May 2008.

[6] F. S. Cattivelli and A. H. Sayed. Diffusion lms strategies for distributed estimation. *IEEE Transactions on Signal Processing*, 58(3):1035–1048, March 2010.

[7] T. H. Chang, M. Hong, and X. Wang. Multi-agent distributed optimization via inexact consensus admm. *IEEE Transactions on Signal Processing*, 63(2):482–497, Jan 2015.

[8] T. H. Chang, A. Nedić, and A. Scaglione. Distributed constrained optimization by consensus-based primal-dual perturbation method. *IEEE Transactions on Automatic Control*, 59(6):1524–1538, June 2014.

[9] Pedro A. Forero, Alfonso Cano, and Georgios B. Giannakis. Consensus-based distributed support vector machines. *The Journal of Machine Learning Research*, 11:1663–1707, Aug 2010.

[10] F. Garin and L. Schenato. *A Survey on Distributed Estimation and Control Applications Using Linear Consensus Algorithms*, pages 75–107. Springer London, London, 2010.

[11] E. Ghadimi, M. Johansson, and I. Shames. Accelerated gradient methods for networked optimization. In *Proceedings of the 2011 American Control Conference*, pages 1668–1673, June 2011.

[12] M.N. Howell, J.P. Whaite, P. Amatyakul, Y.K. Chin, M.A. Salman, C.H. Yen, and M.T. Riefe. Brake pad prognosis system, Apr 2010. US Patent 7,694,555.

[13] Z. Li, I. Kolmanovsky, E. Atkins, J. Lu, D. P. Filev, and J. Michelini. Road risk modeling and cloud-aided safety-based route planning. *IEEE Transactions on Cybernetics*, 46(11):2473–2483, Nov 2016.

[14] Z. Li, I. Kolmanovsky, E. M. Atkins, J. Lu, D. P. Filev, and Y. Bai. Road disturbance estimation and cloud-aided comfort-based route planning. *IEEE Transactions on Cybernetics*, PP(99):1–13, 2017.

[15] Q. Ling, W. Shi, G. Wu, and A. Ribeiro. Dlm: Decentralized linearized alternating direction method of multipliers. *IEEE Transactions on Signal Processing*, 63(15):4051–4064, Aug 2015.

[16] Z. Liu, Y. Liu, and C. Li. Distributed sparse recursive least-squares over networks. *IEEE Transactions on Signal Processing*, 62(6):1386–1395, March 2014.

[17] L. Ljung. *System identification: theory for the user*. Prentice-Hall Englewood Cliffs, NJ, 1999.

[18] C. G. Lopes and A. H. Sayed. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing*, 55(8):4064–4077, Aug 2007.

[19] C. G. Lopes and A. H. Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, July 2008.

[20] G. Mateos, I. D. Schizas, and G. B. Giannakis. Distributed recursive least-squares for consensus-based in-network adaptive estimation. *IEEE Transactions on Signal Processing*, 57(11):4583–4588, Nov 2009.

[21] Gonzalo Mateos and Georgios B. Giannakis. Distributed recursive least-squares: Stability and performance analysis. *CoRR*, abs/1109.4627, 2011.

[22] Peter M. Mell and Timothy Grance. Sp 800-145. the nist definition of cloud computing. Technical report, Gaithersburg, MD, United States, 2011.

[23] Joao F. C. Mota, Joao M. F. Xavier, Pedro M. Q. Aguiar, and Markus Puschel. D-admm: A communication-efficient distributed algorithm for separable optimization. *Trans. Sig. Proc.*, 61(10):2718–2723, May 2013.

[24] R. Olfati-Saber. Distributed kalman filtering for sensor networks. In *2007 46th IEEE Conference on Decision and Control*, pages 5492–5498, Dec 2007.

[25] E. Ozatay, S. Onori, J. Wollaeger, U. Ozguner, G. Rizzoni, D. Filev, J. Michelini, and S. Di Cairano. Cloud-based velocity profile optimization for everyday driving: A dynamic-programming-based solution. *IEEE Transactions on Intelligent Transportation Systems*, 15(6):2491–2505, Dec 2014.

[26] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science, 2 edition, 2012.

[27] S. S. Ram, A. Nedić, and V. V. Veeravalli. Stochastic incremental gradient descent for estimation in sensor networks. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, pages 582–586, Nov 2007.

[28] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli. A new class of distributed optimization algorithms: application to regression of distributed data. *Optimization Methods and Software*, 27(1):71–88, 2012.

[29] A. H. Sayed and C. G. Lopes. Distributed recursive least-squares strategies over adaptive networks. In *2006 Fortieth Asilomar Conference on Signals, Systems and Computers*, pages 233–237, Oct 2006.

[30] I. D. Schizas, G. Mateos, and G. B. Giannakis. Consensus-based distributed recursive least-squares estimation using ad hoc wireless sensor networks. In *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, pages 386–390, Nov 2007.

[31] I. D. Schizas, G. Mateos, and G. B. Giannakis. Distributed lms for consensus-based in-network adaptive processing. *IEEE Transactions on Signal Processing*, 57(6):2365–2382, June 2009.

[32] E. Taheri, O. Gusikhin, and I. Kolmanovsky. Failure prognostics for in-tank fuel pumps of the returnless fuel systems. In *Dynamic Systems and Control Conference*, Oct 2016.

[33] S. Y. Tu and A. H. Sayed. Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks. *IEEE Transactions on Signal Processing*, 60(12):6217–6234, Dec 2012.

[34] E. Wei and A. Ozdaglar. Distributed alternating direction method of multipliers. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 5445–5450, Dec 2012.

[35] L. Xiao, S. Boyd, and S. Lai. A space-time diffusion scheme for peer-to-peer least-squares estimation. In *2006 5th International Conference on Information Processing in Sensor Networks*, pages 168–176, April 2006.

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

[37] C. y. Chong, S. Mori, and K. c. Chang. Adaptive distributed estimation. In *26th IEEE Conference on Decision and Control*, volume 26, pages 2233–2238, Dec 1987.

[38] Ruiliang Zhang and James T. Kwok. Asynchronous distributed admm for consensus optimization. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1701–II–1709. JMLR.org, 2014.