2021 60th IEEE Conference on Decision and Control (CDC)
December 13-15, 2021. Austin, Texas

An ADMM-based approach for multi-class recursive parameter estimation

Valentina Breschi[1], Alberto Bemporad[2] and Ilya Kolmanovsky[3]

2021 60th IEEE Conference on Decision and Control (CDC) | 978-1-6654-3659-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CDC45484.2021.9683108

***Abstract*— Due to the increasing popularity of cloud-based architectures, it is of paramount importance to understand how to benefit from shared information for solving collaborative estimation problems and exploit the additional computational resources available. Meanwhile, it is crucial to devise solutions that allow connected devices to retain private data and to carry out the desired tasks on their own, when disconnected from the cloud.**

In this paper, we present a cloud-aided iterative solution for multi-class parameter estimation for a set of mass-produced devices. The method exploits the similarity between systems operating under comparable conditions and their connection to the cloud, while allowing devices to retain and process raw data privately. The effectiveness of the strategy is assessed on a numerical example, showing its potential.

I. Introduction

Because of the widespread percolation of wireless communication of sensor data, mass-produced devices are increasingly connected with each other. At the same time, they are now also coupled with central repositories, thanks to the growing popularity of cloud-based platforms [10]. This escalation of connectivity gives users and manufacturers access to large datasets, that can be used to enhance the performance of these systems. Specifically, mass-produced devices are designed and calibrated to be nominally equal and their physical similarity can be exploited together with shared information to tackle problems ranging from fault detection [1] to parameter estimation [9]. When tackling this last class of problems, research has mainly focused on distributed solutions, see *e.g.,* [11], [12], [13], [14]. These techniques rely on communication between neighbor nodes only, so as to accommodate the low computational power characterizing Wireless Sensor Networks (WSNs). However, this feature makes these approaches unsuited whenever communication between devices is inhibited because of privacy/security reasons. At the same time, it might cause a drop in performance when consensus among devices is sought.

In these scenarios, cloud-aided solutions become increasingly appealing [6], [7] by allowing consumer devices to eventually retain private data, while giving them access to shared information on-demand. The connection to the cloud further guarantees the availability of dedicated resources with customizable computational power and memory, that can be easily accessed and released. In this light, a cloud-aided consensus-based approach has been recently presented in [3], [4]. By relying on the Alternating Direction Method of Multipliers (ADMM) [2], this strategy is designed to take advantage of the similarities between devices and their connection to the cloud. Nonetheless, it relies on the assumption that they all share some parameters, which may not be satisfied in practice. As a motivating example, assume that we aim at estimating a set of parameters needed for the prognostics of automotive components, *e.g.,* fuel pumps [15]. Suppose that we have access to data collected over a fleet of similar vehicles and we are inferring component aging as a function of its cumulative workload. As the component degradation rate may depend on multiple causes, such as environmental conditions and user behaviors, a common aging model may not be able to accurately describe all devices, making solutions based on full consensus on the wear-rates possibly ineffective when scheduling condition-based maintenance actions.

A. Contribution

To overcome this limitation, we propose a cloud-aided solution for constrained collaborative least-squares estimation over multiple classes. Similarly to the approaches in [3], [4], the proposed scheme is centralized, requiring devices to broadcast surrogates of their raw data only, while benefiting from recursive least-squares (RLS) estimators eventually available locally. Nonetheless, the approach presented here is tailored to enforce intraclass consensus on the cloud, thus allowing one to distinguish between devices operating under different regimes, yet exploiting the connection to the cloud and the nominal similarity between devices. Indeed, similar devices subject to comparable working conditions are likley to age similarly, making it useful to still exploit the similarity in retrieving the aging model.

The paper is organized as follows. Our setting and goal are introduced in Section II, while the multi-class estimation problem is formalized in Section III. The proposed solution is presented in Section IV and its performance is discussed in Section V.

Notation: $\mathbb{N}$, $\mathbb{R}^+$ and $\mathbb{R}^n$ are the sets of natural and real positive numbers (excluding zero), and real vectors of dimension n, respectively. Given a vector $a \in \mathbb{R}^n$, a_i is its i-th element, $\|a\|_2$ its Euclidean norm. The cardinality of a set $\mathcal{A}$ is denoted as $|\mathcal{A}|$, while $\Pi_{\mathcal{A}}(a)$ is the Euclidean projection of the vector a onto $\mathcal{A}$. Given a matrix $A \in \mathbb{R}^{n\times m}$, A' is its transpose. The $n \times n$ identity matrix is denoted as I_n, while the zero vector of dimension n is $\mathbf{0}_n$. For every pair

[1] Dipartimento di Elettronica, Informatica e Bioingegneria (DEIB), Politecnico di Milano, Via Ponzio 34/5, Milano, Italy. Corresponding author: valentina.breschi@polimi.it

[2] IMT School for Advanced Studies Lucca, Piazza San Francesco 19, Lucca, Italy.

[3] Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA.

$a, b \in \mathbb{N}$, the indicator function of the event $\{a=b\}$ is

$$\mathbf{1}_{[a=b]} = \begin{cases} 1 & \text{if } a = b, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Finally, $\mathcal{N}(\mu, \sigma^2)$ is the Gaussian distribution with mean μ and standard deviation σ.

II. Setting and goal

Assume that we collect data from a set of N mass-produced devices, here referred to nodes. Suppose that we can access a matrix $X_n(t) \in \mathbb{R}^{n_y \times n_\theta}$ of features and the corresponding output $y_n(t) \in \mathbb{R}^{n_y}$ at each time instant $t \in \mathbb{N}$, for each $n \in \mathcal{I}_N = \{1, \dots, N\}$. Assume that the devices can exchange information with the cloud, while retaining raw data locally for security/privacy reasons or due to communication bandwidth limitations. Since mass-produced devices are designed to be nominally the same, let them be characterized by structurally identical linear regressor/output relationships

$$y_n(t) = X_n(t)\theta_n^{\mathrm{o}}(t) + e_n(t), \quad n = 1, \dots, N, \quad (2)$$

where $e_n(t) \in \mathbb{R}^{n_y}$ is a zero-mean white noise independent of $X_n(t)$, $\theta_n^{\mathrm{o}}(t) \in \Theta_n \subseteq \mathbb{R}^{n_\theta}$ is a local vector of *unknown* parameters, subject to *sudden* changes in their values according to the operating condition of the system, and Θ_n is assumed to be closed and convex.

Despite the resemblance of their dynamics, assume that these systems do not always share the same parameters as they are subject to different conditions of use. Meanwhile, suppose that the range of possible working conditions can be clustered into $M \ll N$ macro-level classes according to their relative similarity, and that each device can be *uniquely* associated to one of these groups at each instant t. Accordingly, let us assume that there exists a set of *constant* M class-dependent parameters $\vartheta_m \in \mathbb{R}^{n_g}$, with $n_g \leq n_\theta$ and $m = 1, \dots, M$, that are common to all nodes within each macro-level group and that are linked to the local parameters via the following equality:

$$P\theta_n^{\mathrm{o}}(t) = \vartheta_{s_n(t)}, \quad (3)$$

where $P \in \mathbb{R}^{n_g \times n_\theta}$ is a *known* matrix dictating the intraclass similarities, while $s_n(t) \in \{1, \dots, M\}$ is an integer variable dictating the group the n-th device belongs to at time t, *i.e.,*

$$s_n(t) = m \iff n \in \mathcal{C}_m(t), \quad m \in \{1, \dots, M\}, \quad (4)$$

with $\mathcal{C}_m(t)$ being the time-varying set comprising the devices associated with the m-th cluster at time t.

Our goal is to use the available data to iteratively estimate the local parameter vector $\theta_n^{\mathrm{o}}(t)$ in a memory-effective and computationally efficient way, while exploiting the similarities between devices belonging to the same class and eventually using priors on the set Θ_n.

III. Problem formulation

The one-to-one correspondence dictated by (4) implies that

$$\sum_{m=1}^{M} \mathbf{1}_{[s_n(t)=m]} = 1, \quad \forall t, \ \forall n \in \mathcal{I}_N. \quad (5)$$

Accordingly, the cloud-aided estimation problem to be solved at each time instant t can be formally cast as the following multi-model constrained least-squares problem:

$$\min_{\substack{\{\theta_n, \mathcal{S}_n^t\}_{n=1}^N \\ \{\vartheta_m\}_{m=1}^M}} \ \frac{1}{2} \sum_{n=1}^{N} \sum_{\tau=1}^{t} \lambda_n^{t-\tau} \|y_n(\tau) - X_n(\tau)\theta_n\|_2^2 \quad (6a)$$

$$\text{s.t.} \quad \theta_n \in \Theta_n, \ \forall n \in \mathcal{I}_N, \quad (6b)$$

$$\sum_{m=1}^{M} \mathbf{1}_{[s_n(\tau)=m]} (P\theta_n - \vartheta_m) = 0, \ \forall n \in \mathcal{I}_N, \quad (6c)$$

where $\mathcal{S}_n^t = \{s_n(\tau)\}_{\tau=1}^t$, the constraint in (6c) holds for all instants $\tau \in \mathcal{I}_t = \{1, \dots, t\}$, so as to enforce intraclass consensus, and θ_n denotes the estimate of the local parameters based on the data collected up to time t. The forgetting factor $\lambda_n \in (0, 1]$ used in the local cost

$$f_n(\theta_n; t) = \frac{1}{2} \sum_{\tau=1}^{t} \lambda_n^{t-\tau} \|y_n(\tau) - X_n(\tau)\theta_n\|_2^2, \quad (7)$$

allows us to eventually trust recent data more [8].

Towards tackling problem (6) by performing simple recursive operations on board of each device, we rewrite it as

$$\min_{\substack{\{\theta_n, z_n, \mathcal{S}_n^t\}_{n=1}^N \\ \{\vartheta_m\}_{m=1}^M}} \ \sum_{n=1}^{N} [f_n(\theta_n; t) + g_n(z_n)] \quad (8a)$$

$$\text{s.t.} \quad \theta_n = z_n, \ \forall n \in \mathcal{I}_N, \quad (8b)$$

$$\sum_{m=1}^{M} \mathbf{1}_{[s_n(\tau)=m]} (P\theta_n - \vartheta_m) = 0, \ \forall n \in \mathcal{I}_N, \quad (8c)$$

where $f_n : \mathbb{R}^{n_\theta} \to \mathbb{R}$ is defined as in (7), and each optimization variable θ_n in (6) has been split into two parts via the introduction of the auxiliary variables z_n, for $n \in \mathcal{I}_N$. Along with the augmentation of the cost with

$$g_n(z_n) = \begin{cases} 0 \text{ if } z_n \in \Theta_n, \\ +\infty \text{ otherwise,} \end{cases} \quad n = 1, \dots, N, \quad (8d)$$

the reformulation makes the objective of problem in (8) separable over this splitting, paving the way for a solution decoupling the fitting problem from the constraints in (6b).

IV. A recursive approach for collaborative multi-class estimation

Solving (8) entails the estimation of (i) the local parameter θ_n and the auxiliary variable z_n for each node $n \in \mathcal{I}_N$, (ii) the class-dependent global variables ϑ_m, with $m = 1, \dots, M$, and (iii) the sequence $\mathcal{S}_n^t = \{s_n(\tau)\}_{\tau=1}^t$ of integers dictating the active mode at each step up to t. To retrieve all these unknowns and limit the computational burden on the devices, we propose to alternate between an ADMM-based scheme to solve (8) with respect to $\{\theta_n, z_n\}_{n=1}$ and $\{\vartheta_m\}_{m=1}^M$, for a fixed sequence $\mathcal{S}_n^t$, and a greedy approach used to find the latest class to which each device belongs to, for fixed local and global parameters.

Denote by $F_n(\theta_n, z_n; t) = f_n(\theta_n; t) + g_n(z_n)$ the local loss function, for all $n \in \mathcal{I}_N$. Let $r_{n,1}$ and $r_{n,2}(\tau)$ be the residuals

for the equality constraints in (8b) and (8c) respectively, *i.e.,*

$$r_{n,1} = \theta_n - z_n, \tag{9a}$$

$$r_{n,2}(\tau) = \sum_{m=1}^{M} \mathbf{1}_{[s_n(\tau)=m]}(P\theta_n - \vartheta_m), \quad \forall \tau \in \mathcal{I}_t, \tag{9b}$$

for $n = 1, \dots, N$.

Both stages of our approach rely on the augmented Lagrangian associated with (8), which is given by

$$\mathcal{L}^t = \sum_{n=1}^{N} \mathcal{L}_n \left(\theta_n, z_n, \delta_{n,1}, \vartheta, \mathcal{S}_n^t, \Delta_{n,2}^t\right), \tag{10a}$$

where $\Delta_{n,2}^t = \{\delta_{n,2}(\tau)\}_{\tau=1}^t$ and $\mathcal{L}_n$ is

$$\begin{aligned}\mathcal{L}_n =& F_n(\theta_n, z_n; t) + \delta_{n,1}' r_{n,1} + \frac{\rho_1}{2}\|r_{n,1}\|_2^2 \\ &+ \sum_{\tau=1}^{t} \left[\delta_{n,2}'(\tau) r_{n,2}(\tau) + \frac{\rho_2}{2}\|r_{n,2}(\tau)\|_2^2\right],\end{aligned} \tag{10b}$$

the intraclass consensus residual in (9b) has been recast as

$$r_{n,2}(\tau) = P\theta_n - \vartheta_{s_n(\tau)}, \quad \forall \tau \in \mathcal{I}_t, \tag{11}$$

and $\delta_{n,1} \in \mathbb{R}^{n_\theta}$ and $\{\delta_{n,2}(\tau) \in \mathbb{R}^{n_g}\}_{\tau=1}^t$ are the Lagrange multipliers associated to (8c) and (8b), respectively. Clearly, the augmented Lagrangian in (10) is separable across the devices, allowing us to process individual (eventually private) data on board of each system or in resources specifically allocated on the cloud, as detailed next.

A. Parameter estimation

At time t, suppose that the sequence $\mathcal{S}_n^t$ is fixed for all $n \in \mathcal{I}_N$. Denote by $\vartheta = \{\vartheta_m\}_{m=1}^M$ the unknown class-dependent parameters. For each node $n \in \mathcal{I}_N$ and each class $m \in \{1, \dots, M\}$, the estimates $\{\hat{\theta}_n(t)\}_{n=1}^N$ and $\hat{\vartheta}$ of the local and the class-dependent parameters can be retrieved by carrying out the following ADMM steps:

$$\hat{\theta}_n^{(i+1)}(t) = \underset{\theta_n}{\operatorname{argmin}}\, \mathcal{L}_n(\theta_n, z_n^{(i)}, \delta_{n,1}^{(i)}, \vartheta^{(i)}, \mathcal{S}_n^t, \Delta_{n,2}^{t,(i)}), \tag{12a}$$

$$z_n^{(i+1)} = \underset{z_n}{\operatorname{argmin}}\, \mathcal{L}_n(\hat{\theta}_n^{(i+1)}(t), z_n, \delta_{n,1}^{(i)}, \vartheta^{(i)}, \mathcal{S}_n^t, \Delta_{n,2}^{t,(i)}), \tag{12b}$$

$$\hat{\vartheta}^{(i+1)} = \underset{\vartheta}{\operatorname{argmin}} \sum_n \mathcal{L}_n(\hat{\theta}_n^{(i+1)}(t), z_n^{(i+1)}, \delta_{n,1}^{(i)}, \vartheta, \mathcal{S}_n^t, \Delta_{n,2}^{t,(i)}), \tag{12c}$$

$$\delta_{n,1}^{(i+1)} = \delta_{n,1}^{(i)} + \rho_1(\hat{\theta}_n^{(i+1)}(t) - z_n^{(i+1)}), \tag{12d}$$

$$\delta_{n,2}^{(i+1)}(\tau) = \delta_{n,2}^{(i)}(\tau) + \rho_2(P\hat{\theta}_n^{(i+1)}(t) - \vartheta_{s_n(\tau)}^{(i+1)}), \ \forall \tau \in \mathcal{I}_t, \tag{12e}$$

where $i \in \mathbb{N}$ indicates the current iteration, and the minimization problems in (12a)-(12b) must be solved for all devices $n \in \mathcal{I}_N$, along with the updates in (12d)-(12e). These iterations have to be carried out until a stopping criterion is met, *e.g.,* a maximum number i_{max} of iterations is exceeded.

1) Auxiliary variables update: Let us focus on the update in (12b) and denote by $r_{n,1}^{(i+1)} = \hat{\theta}_n^{(i+1)}(t) - z_n$the residual in (9a) for the updated estimate $\hat{\theta}_n^{(i+1)}(t)$. Due to the features of (10b), the n-th auxiliary variable is updated as

$$z_n^{(i+1)} = \underset{z_n}{\operatorname{argmin}}\ g_n(z_n) + (\delta_{n,1}^{(i)})' r_{n,1}^{(i+1)} + \frac{\rho_1}{2}\|r_{n,1}^{(i+1)}\|_2^2. \tag{13}$$

By manipulating the cost in (13) it can be proven [2] that the optimal auxiliary variable corresponds to the projection of $\hat{\theta}_n^{(i+1)}(t) + \frac{1}{\rho_1}\delta_{n,1}^{(i)}$ onto the closed convex set Θ_n. Therefore, Step (12b) translates into

$$z_n^{(i+1)} = \Pi_{\Omega_n}\left(\hat{\theta}_n^{(i+1)}(t) + \frac{1}{\rho_1}\delta_{n,1}^{(i)}\right), \quad \forall n \in \mathcal{I}_N. \tag{14}$$

2) Class-dependent parameters update: Let $r_{n,2}^{(i+1)}(\tau)$ be the intraclass consensus residual in (11) computed with the updated local parameters, *i.e.,* $r_{n,2}^{(i+1)}(\tau) = P\hat{\theta}_n^{(i+1)} - \vartheta_{s_n(\tau)}$, for all $\tau \in \mathcal{I}_t$. Based on the definition of the augmented Lagrangian in (10), the parameters $\vartheta^{(i+1)} = \{\vartheta_m^{(i+1)}\}_{m=1}^M$ are retrieved by solving the minimization problem,

$$\min_{\vartheta} \sum_{n=1}^{N}\sum_{\tau=1}^{t}\left[(\delta_{n,2}^{(i+1)}(\tau))' r_{n,2}^{(i+1)}(\tau) + \frac{\rho_2}{2}\|r_{n,2}^{i+1}(\tau)\|_2^2\right]. \tag{15}$$

Let $S_m^\tau = \{n \in \mathcal{I}_N : s_n(\tau) = m\}$ be the set of devices associated to the m-th mode at time τ according to the fixed sequence $\mathcal{S}_n^t$, for $m \in \{1, \dots, M\}$. The objective characterizing (15) can be equivalently recast as

$$\sum_{m=1}^{M}\sum_{\tau=1}^{t}\sum_{n\in S_m^\tau}\left[(\delta_{n,2}^{(i+1)}(\tau))' r_{n,2}^{(i+1)}(\tau) + \frac{\rho_2}{2}\|r_{n,2}^{(i+1)}(\tau)\|_2^2\right], \tag{16}$$

showing that problem (15) can be split across the classes. Therefore, the estimate of each class-dependent parameter at time t, *i.e.,* $\{\hat{\vartheta}_m(t)\}_{m=1}^M$, can be obtained as

$$\min_{\vartheta_m} \sum_{\tau=1}^{t}\sum_{n\in S_m^\tau}\left[(\delta_{n,2}^{(i+1)}(\tau))' r_{n,2}^{(i+1)}(\tau) + \frac{\rho_2}{2}\|r_{n,2}^{(i+1)}(\tau)\|_2^2\right]. \tag{17}$$

The cost in (17) can be further split across time, thus allowing a recursive expression to update the class-dependent parameters in closed-form:

$$\hat{\vartheta}_m^{(i+1)}(\tau) = \frac{1}{|S_m^\tau|}\sum_{n\in S_m^\tau}\left[P\hat{\theta}_n^{(i+1)}(t) + \frac{\delta_{n,2}^{(i)}(\tau)}{\rho_2}\right], \ \forall \tau \in \mathcal{I}_t. \tag{18}$$

3) Local parameter update: The structure of the loss function in (7) allows us to find the analytical solution to the minimization problem in (12a). The latter can be compactly cast as follows:

$$\hat{\theta}_n^{(i+1)}(t) = \phi_n(t)\left(\mathcal{Y}_n(t) + \xi_n^{(i)}(t)\right), \tag{19}$$

with

$$\mathcal{Y}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n'(\tau) y_n(\tau), \tag{20a}$$

$$\xi_n^{(i)}(t) = \rho_1 z_n^{(i)} - \delta_{n,1}^{(i)} + \sum_{\tau=1}^{t} P'\left[\rho_2 \vartheta_{s_n(\tau)}^{(i)} - \delta_{n,2}^{(i)}(\tau)\right], \tag{20b}$$

$$\phi_n(t) = (\mathcal{X}_n(t) + t\rho_2 P'P + \rho_1 I_{n_\theta})^{-1} \tag{20c}$$

and

$$\mathcal{X}_n(t) = \sum_{\tau=1}^{t} \lambda_n^{t-\tau} X_n'(t) X_n(t). \tag{20d}$$

Based on these definitions, the local estimate in (19) can be recast as the sum of two terms, namely

$$\hat{\theta}_n^{rls}(t) = \phi_n(t)\mathcal{Y}_n(t), \tag{21a}$$

$$\hat{\theta}_n^{admm,(i+1)}(t) = \phi_n(t)\xi_n^{(i)}(t), \tag{21b}$$

where the partial estimate in (21a) depends on the data available locally, while the one in (21b) is iteration-dependent and it can be readily computed by relying on the outcomes of the previous ADMM run.

Let us focus on the partial estimate $\hat{\theta}_n^{rls}(t)$ in (21a) and consider again $\mathcal{Y}_n(t)$ in (20a) and $\mathcal{X}_n(t)$ in (20d). The latter quantities can be updated as:

$$\mathcal{Y}_n(t) = X_n'(t)y_n(t) + \lambda_n \mathcal{Y}_n(t-1), \tag{22a}$$

$$\mathcal{X}_n(t) = X_n'(t)X_n(t) + \lambda_n \mathcal{X}_n(t-1). \tag{22b}$$

By leveraging the properties in (22) and by introducing

$$\hat{\theta}_n^{rls}(t-1) = \phi_n(t-1)\mathcal{Y}_n(t-1), \tag{23a}$$

$$\phi_n(t-1) = (\mathcal{X}_n(t-1) + (t-1)\rho_2 P'P + \rho_1 I_{n_\theta})^{-1} \tag{23b}$$

manipulations similar to the one performed in [3] allow us to obtain a *recursive* update for the estimate in (21a), *i.e.,*

$$\hat{\theta}_n^{rls}(t) = \hat{\theta}_n^{rls}(t-1) + K_n(t)\varepsilon_n(t|t-1), \tag{24a}$$

where

$$\varepsilon_n(t|t-1) = \tilde{y}_n(t) - \tilde{X}_n(t)\hat{\theta}_n^{rls}(t-1), \tag{24b}$$

$$K_n(t) = \phi_n(t-1)\tilde{X}_n'(t)(\lambda_n I_{n_{\tilde{X}}} + \tilde{X}_n(t)\phi_n(t-1)\tilde{X}_n'(t))^{-1} \tag{24c}$$

$$\phi_n(t) = \lambda_n^{-1}(I_{n_\theta} - K_n(t)\tilde{X}_n(t))\phi_n(t-1), \tag{24d}$$

and the extended regressor $\tilde{X}_n(t) \in \mathbb{R}^{n_{\tilde{X}} \times n_\theta}$, with $n_{\tilde{X}} = n_y + n_\theta + n_g$, and output $\tilde{y}_n(t)$ are respectively given by

$$\tilde{X}_n(t) = \begin{bmatrix} X_n(t) \\ \sqrt{[t-(t-1)\lambda_n]\rho_2}P \\ \sqrt{(1-\lambda_n)\rho_1}I_{n_\theta} \end{bmatrix}, \quad \tilde{y}_n(t) = \begin{bmatrix} y_n(t) \\ \mathbf{0}_{n_g} \\ \mathbf{0}_{n_\theta} \end{bmatrix}. \tag{24e}$$

The separation induced by the use of the ADMM-based scheme leads to update formulas that resemble the ones of standard recursive least squares, modified to handle the forgetting factor and the constraint-dependent terms in (20c).

B. Class estimation

Assume now that the parameters $\{\theta_n\}_{n=1}^N$ and $\{\vartheta_m\}_{m=1}^M$ are fixed, along with the Lagrange multipliers $\delta_{n,1}$ and $\{\delta_{n,2}(\tau)\}_{\tau=1}^t$, for $n \in \mathcal{I}_N$. By considering once again the augmented Lagrangian in (10), the class can be estimated over time by solving the following problem,

$$\min_{\{\mathcal{S}_n^t\}_{n\in\mathcal{I}_N}} \sum_{n=1}^N \sum_{\tau=1}^t \Big[\delta_{n,2}'(\tau)r_{n,2}(\tau) + \frac{\rho_2}{2}\|r_{n,2}(\tau)\|_2^2\Big], \tag{25}$$

where the residual $r_{n,2}(\tau)$ is defined as in (11) and we have neglected all terms of $\mathcal{L}^t$ in (10) that are independent of the class. Note that, whenever ρ_2 is sufficiently large to let the quadratic term dominate over the linear one, the problem in (25) shares the same cost function as K-means [5] with centroids dictated by the class-dependent parameters.

The objective function in (25) can be split across the nodes and time, allowing us to retrieve the class of each device iteratively as follows:

$$\hat{s}_n(\tau) = \arg\min_{s_n(\tau)} \delta_{n,2}'(\tau)r_{n,2}(\tau) + \frac{\rho_2}{2}\|r_{n,2}(\tau)\|_2^2, \tag{26}$$

for all $n \in \mathcal{I}_N$ and instants $\tau \in \mathcal{I}_t$. Here we propose to greedily evaluate the cost in (26) for all possible clusters $m \in \{1, \dots, M\}$, and then pick the class corresponding to the minimal loss.

C. Communication scheme

The simple, recursive operations in (24) solely rely on data available at the device level. Moreover, they require few variables to be stored on board of the nodes, *i.e.,* $\hat{\theta}_n^{rls}(t-1)$ and $\phi_n(t-1)$. It is thus natural to perform these updates on board of each device, prior to the iterations of the actual ADMM scheme. By computing a partial estimate of the local parameters, each node can privately retain raw data while only sharing their surrogates, *i.e.,* $\hat{\theta}_n^{rls}(t)$ and $\phi_n(t)$, with the cloud. Moreover, if the connection with the cloud is lost, an estimate of the local parameters would be available at the device level. All remaining operations depend on the ADMM run counter and they have to be performed by all devices. Thus, they can be performed either by the central processing unit or by exploiting resources allocated to each device on the cloud, including the update of $\{s_n(t)\}_{n\in\mathcal{I}_N}$. To benefit from the estimated class-dependent parameters retrieved on the cloud, we introduce the additional local estimate

$$\tilde{\theta}_n(t) = \hat{\theta}_n^{rls}(t) + P'(\hat{\vartheta}_{s_n(t-1)}(t-1) - P\hat{\theta}_n^{rls}(t-1)), \tag{27}$$

correcting the estimate available locally with the previous parameter (broadcast to the node by the cloud).

D. Practical hints

The ADMM-based scheme in (12) requires updating the estimates of the class-dependent parameters by exploiting information collected up to time t, as shown in (17). Analogously, the Lagrangian multipliers associated to each intraclass similarity constraint must be updated as in (12e) over the whole time horizon. These operations are performed on the cloud, but they make the scheme not recursive. Since we aim at devising an iterative approach, we propose to approximate the steps in (12c) and (12e) by exploiting their separability over time. Therefore, at time t, we *approximately* compute the estimates $\{\hat{\vartheta}_m(t)\}_{m=1}^M$ as in (17) and update $\{\delta_{n,2}(t)\}_{n=1}^N$ only, while fixing $\{\delta_{n,2}(\tau)\}_{\tau=1}^{t-1}$ to their values obtained at previous time instants.

In Algorithm 1 we summarize a possible implementation of the method at time t, where the parameters at time t are estimated by fixing $s_n(t) = \hat{s}_n(t-1)$. Note that the proposed implementation involves the introduction of an additional penalty $\rho_3 > 0$, which replaces ρ_2 when the mode is updated as in (26). This change enables us to enforce the quadratic term in (26) to dominate over the linear one, while not jeopardizing constraint satisfaction. Algorithm 1 further requires the initialization of the class-dependent estimates, the auxiliary variables and the Lagrange multipliers. Since $\{\hat{\theta}_n^{rls}(t)\}_{n\in\mathcal{I}_N}$ are updated prior to the ADMM runs, they can be exploited to initialize these parameters. In particular, $z_n^{(0)}$ can be chosen as the projection of $\hat{\theta}_n^{rls}(t)$ onto Θ_n, for all $n \in \mathcal{I}_N$, while $\hat{\vartheta}_m^{(0)}$ can be initialized as the sample mean of $\{P\hat{\theta}_n^{rls}\}_{n:\hat{s}_n(t-1)=m}$ for all $m = 1, \dots, M$.

Algorithm 1 ADMM-RLS for multi-class estimation

Local inputs:$\{X_n(t), y_n(t)\}$, $\{\hat{\theta}_n^{rls}(t-1), \phi_n(t-1)\}$, $\hat{\vartheta}_{\hat{s}_n(t-1)}(t-1)$; $\lambda_n \in (0,1]$; $\rho_1, \rho_2, \rho_3 \in \mathbb{R}^+$.
Cloud inputs: $\hat{\vartheta}^{(0)}(t)$, $\{\delta_{n,i}^{(0)}\}_{i=1}^2$, $z_n^{(0)}$, $n \in \mathcal{I}_N$; ρ_1, ρ_2, ρ_3.

Node-level computations

1. **each node** $n \in \mathcal{I}_N$ **does**
1.1. **update** $\hat{\theta}_n^{rls}(t)$ and $\phi_n(t)$ as in (24);
1.2. **discard** $\{\hat{\theta}_n^{rls}(t-1), \phi_n(t-1)\}$;
1.3. **memorize** $\{\hat{\theta}_n^{rls}(t), \phi_n(t)\}$;
1.4. **compute** $\tilde{\theta}_n(t)$ as in (27);
1.5. **transmit** $\hat{\theta}_n^{rls}(t)$ and $\phi_n(t)$ **to** the **cloud**;

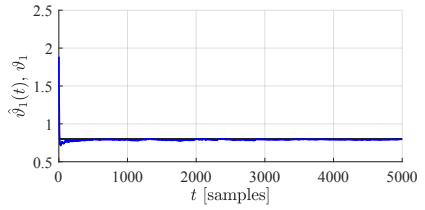
Cloud-level computations

1. **iterate for** $i = 1, \dots$
1.1. **for all** $n \in \mathcal{I}_N$
1.1.1. **compute** $\hat{\theta}_n^{admm,(i+1)}(t)$ as in (21b);
1.1.2. **set** $\hat{\theta}_n^{(i+1)}(t) \leftarrow \hat{\theta}_n^{rls}(t) + \hat{\theta}_n^{admm,(i+1)}(t)$;
1.1.3. **compute** $z_n^{(i+1)}$ as in (14);
1.2. **for** $m = 1, \dots, M$
1.2.1. **update** $\hat{\vartheta}_m^{(i+1)}(t)$ as in (18);
1.3. **for all** $n \in \mathcal{I}_N$
1.3.1. **compute** $\delta_{n,1}^{(i+1)}$ and $\delta_{n,2}^{(i+1)}(t)$ as in (12d) and (12e);
2. **until** a stopping criterion is satisfied;
3. **for all** $n \in \mathcal{I}_N$
3.1. **update** $\hat{s}_n(t)$ as in , with $\rho_2 \leftarrow \rho_3$;
4. **transmit** $\hat{\vartheta}_{\hat{s}_n(t)}(t)$ **to** the n-th node, $n \in \mathcal{I}_N$.

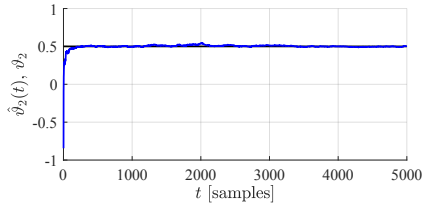
Local outputs: $\phi_n(t)$, $\hat{\theta}_n^{rls}(t)$ and $\tilde{\theta}_n(t)$.
Cloud outputs: $\{\hat{\theta}_n(t), \hat{s}_n(t)\}_{n=1}^N$; $\{\hat{\vartheta}_m(t)\}_{m=1}^M$.

TABLE I
PARAMETERS AND INITIAL CONDITIONS FOR ALGORITHM 1

$\phi_n(0)$	λ_n	ρ_1	ρ_2	ρ_3	$\delta_{n,1}^{(0)}$	$\delta_{n,2}^{(0)}$
$0.1I_2$	0.995	0.1	10^{-5}	1	$10^{-8}I_5$	$10^{-8}I_5$



(a) ϑ_1 vs $\hat{\vartheta}_1(t)$

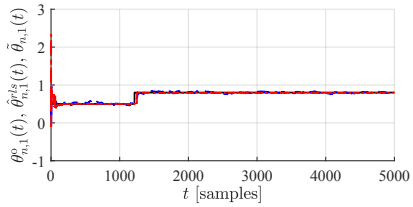


(b) ϑ_2 vs $\hat{\vartheta}_2(t)$

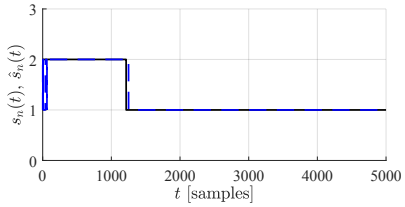
Fig. 1. Class-dependent estimates (blue) *vs* true values (black) over time.

V. NUMERICAL EXAMPLE

Consider $N = 100$ dynamical systems described by the following ARX model:



(a) $\theta_{n,1}^o$ (black), $\hat{\theta}_n^{rls}(t)$ (dashed blue), $\tilde{\theta}_n(t)$ (red)



(b) $s_n(t)$ (dashed black) and $\hat{s}_n(t)$ (blue)

Fig. 2. Node #3: true *vs* estimated local parameters and modes over time.

$$y_n(t) = \theta_{n,1}^o y_n(t-1) + \theta_{n,2}^o u_n(t-1) + e_n(t), \quad (28a)$$

where $\theta_{n,2}^o \sim \mathcal{N}(0.4, 4\cdot 10^{-4})$ is purely local, while the value of $\theta_{n,1}^o$ is dictated by the following class-based rule:

$$\theta_{n,2}^o = \vartheta_{s_n(t)} = \begin{cases} 0.8 \text{ if } s_n(t) = 1, \\ 0.5 \text{ if } s_n(t) = 2. \end{cases} \quad (28b)$$

According to the dynamics in (28a), the feature vector in (24) is constructed as $X_n(t) = \begin{bmatrix} y_n(t-1) & u_n(t-1) \end{bmatrix}'$. Throughout an estimation horizon of $T = 5000$, the class of each node changes only once at a randomly chosen instant within $[100, T)$. All local input sequences comprise a set of uniformly distributed inputs in the interval $[2, 3]$ generated at random, while $e_n \sim \mathcal{N}(0, R_n)$ is characterized by a device-dependent variance $R_n \in [1, 4]$. The effect of noise is quantified through the Signal-to-Noise Ratio (SNR), yielding $\text{SNR}_n \in [4.5, 14.3]$ dB for all $n \in \mathcal{I}_N$.

At each instant t, a new instance of Algorithm 1 is run by using the parameters reported in Table I, and stopped once $i_{max} = 20$ iterations have been performed. We also rely on the following priors:

$$0.45 \le \vartheta_i \le 0.85, \quad i = 1, 2, \quad (29a)$$
$$0.3 \le \theta_{n,2} \le 0.5, \quad \forall n \in \mathcal{I}_N. \quad (29b)$$

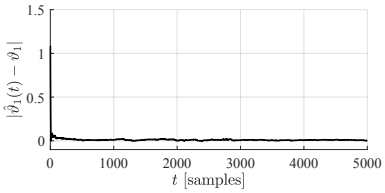
The initial class $\hat{s}_n(0)$ is chosen at random. According to its value, we randomly initialize the local estimates as $\hat{\theta}_n(0) \sim \mathcal{N}(\begin{bmatrix} \vartheta_{\hat{s}_n(0)} & \theta_{n,2}^o \end{bmatrix}, I_2)$, while the remaining initial estimates are selected according to Section IV-D.

Figure 1 shows that, after an initial transient, both estimates converge to the actual values that can be taken by $\theta_{n,1}^o$ according to (28b), despite the approximations discussed in Section IV-D. This result could only be achieved when the local estimates and the reconstructed operating conditions of the devices are sufficiently accurate throughout the horizon. The comparison between the purely data-driven and the corrected estimated in Figure 2 highlights that $\{\tilde{\theta}_n(t)\}_{n=1}$ are less subject to oscillations due to the noisy nature of the data. Therefore, the information broadcast back from the cloud helps improving the estimates obtained by processing local data only. Meanwhile, a delay in the estimation of the actual active mode might lead to a corresponding slower

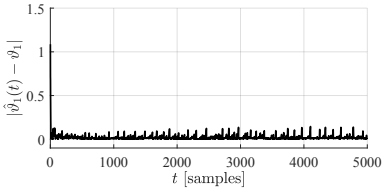
TABLE II

SENSITIVITY ANALYSIS: AVERAGE $L^{true}_{n,\%}$ (30) *vs* ρ_3.

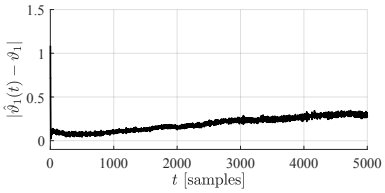
	$\rho_3 = 10^{-2}$	$\rho_3 = 10^{-1}$	$\rho_3 \geq 1$
mean($L^{true}_{n,\%}$) [%]	8.9	5.7	5.5



(a) $|\hat{\vartheta}_1(t) - \vartheta_1|$ for $\rho_2 = 10^{-5}$



(b) $|\hat{\vartheta}_1(t) - \vartheta_1|$ for $\rho_2 = 10^{-3}$



(c) $|\hat{\vartheta}_1(t) - \vartheta_1|$ for $\rho_2 = 10^{-1}$

Fig. 3. Sensitivity analysis: absolute estimation error *vs* ρ_2. Increasing ρ_2 leads to a deterioration of the estimator performance. The same behavior characterizes the absolute error on ϑ_2.

TABLE III

SENSITIVITY ANALYSIS: AVERAGE $L^{true}_{n,\%}$ (30) *vs* ρ_2

	$\rho_2 = 10^{-5}$	$\rho_2 = 10^{-3}$	$\rho_2 = 10^{-1}$
mean($L^{true}_{n,\%}$) [%]	5.5	8.7	25.5

correction, which does not impact the quality of the retrieved class-dependent parameters. The quality of the reconstructed mode sequence is instead measured as:

$$L^{true}_{n,\%} = \left(\frac{1}{T} \sum_{t=1}^{T} \mathbf{1}_{[s_n(t) \neq \hat{s}_n(t)]} \right) \cdot 100, \qquad (30)$$

which is computed using the true sequences as ground-truth. The average number of misclassified steps $L^{true}_{n,\%}$ over the nodes is equal to 5 %. This indicates that the retrieved mode sequences are accurate overall, with initial errors related to the random initializations and delays due to slow transitions of the local estimates towards the true parameter.

a) Sensitivity analysis: The performance of the proposed approach is now assessed for different choices of $\rho_1, \rho_2, \rho_3 \in \mathbb{R}^+$, by changing one parameter at a time and keeping the others as in Table I.

As ρ_1 is associated with the value constraints in (6b), we look at the average number of constraint violations. By selecting ρ_1 between 10^{-5} and 10^{-1} we have always obtained that the range constraints in (29) are violated around 3% of the times. Since ρ_3 directly influences the quality of the reconstructed class sequences, the performance of Algorithm 1 for different values of this penalty is evaluated via the average $L^{true}_{n,\%}$ reported in Table II. It can be seen that the higher ρ_3 is, the smaller is the number of spurious switches in the class estimate. As ρ_2 weights violations of the intraclass consensus constraints, different choices are expected to primarily affect the quality of the global estimates and, indirectly, the accuracy of the reconstructed class sequences. The absolute error on the global parameters for different ρ_2 are reported in Figure 3. We can conclude that the larger ρ_2 is, the less the global parameters of different classes are distinguishable, as the consensus constraint is enforced too strongly. As expected, this has a negative effect on the quality of the estimated classes (see Table III).

VI. CONCLUSIONS

We presented an ADMM-based recursive approach to estimate a set of parameters in a collaborative fashion. The method is tailored to scenarios in which data are acquired from N mass-produced devices, that can be clustered into M macro-level groups according to their operating regime.

In the future, the convergence of the approach will be formally studied and the effect of asynchronous and unideal communications will be explored in depth.

REFERENCES

[1] F. Boem, R. M.G. Ferrari, C. Keliris, T. Parisini, and M.M. Polycarpou. A distributed networked approach for fault detection of large-scale systems. *IEEE Transactions on Automatic Control*, 62(1):18–33, 2017.

[2] 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, 2011.

[3] V. Breschi, A. Bemporad, and I.V. Kolmanovsky. Cooperative constrained parameter estimation by ADMM-RLS. *Automatica*, 121:109175, 2020.

[4] V. Breschi, I. Kolmanovsky, and A. Bemporad. Cloud-aided collaborative estimation by ADMM-RLS algorithms for connected vehicle prognostics. In *Proceedings of the American Control Conference (ACC)*, 2018.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2009.

[6] 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, 2016.

[7] 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.

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

[9] 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, 2009.

[10] P.M. Mell and T. Grance. Sp 800-145. the NIST definition of cloud computing. Technical report, Gaithersburg, MD, United States, 2011.

[11] R. Nassif, S. Vlaski, and A.H. Sayed. Adaptation and learning over networks under subspace constraints – Part I: Stability analysis, 2019.

[12] 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, 2007.

[13] A.K. Sahu, D. Jakovetić, and S. Kar. $\mathcal{CIRFE}$: A distributed random fields estimator. *IEEE Transactions on Signal Processing*, 66(18):4980–4995, 2018.

[14] 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, 2009.

[15] 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*, 2016.