Distributed Variance Consensus with Application to Personalized Learning

Diego Deplano * **Nicola Bastianello** ** **Mauro Franceschelli** *
Karl H. Johansson **

* *DIEE, University of Cagliari, 09123 Cagliari, Italy (e-mails:* {diego.deplano,mauro.franceschelli}@unica.it*).*
** *School of Electrical Engineering and Computer Science, and Digital Futures, KTH Royal Institute of Technology, Stockholm, Sweden (e-mails:* {nicolba,kallej}@kth.se*)*

Abstract: This paper addresses the problem of computing the sample variance of datasets scattered across a network of interconnected agents. A general procedure is outlined to allow the agents to reach consensus on the variance of their local data, which involves two cascaded (dynamic) average consensus protocols. Our implementation of the procedure exploits the distributed ADMM, yielding a distributed protocol that does not involve the sharing of any local, private data nor any coordination of a central authority; the algorithm is proved to be convergent with linear rate and null steady-state error. The proposed distributed variance estimation scheme is then leveraged to tune personalization in "*personalized learning*" where agents aim at training a local model tailored to their own data, while still benefiting from the cooperation with other agents to enhance the models' generalization power. The degree to which an agent tailors its local model depends on the diversity of the local datasets, and we propose to use the variance to tune personalization. Numerical simulations test the proposed approach in a classification task of handwritten digits, drawn from the EMNIST dataset, showing the better performance of variance-tuned personalization over non-personalized training.

1. INTRODUCTION

In the context of distributed systems, the pursuit of consensus has emerged as a focal point of research, driven by the imperative to infer global information from local, private data. The goal then is the development of algorithms that enable the agents to converge upon a shared understanding, or consensus, relying solely on peer-to-peer interactions. Different functions of the local data have been explored as objectives for consensus: the *average* (Moreau, 2005; Freeman et al., 2006; Zhu and Martínez, 2010; Kia et al., 2019; Bastianello and Carli, 2022), the *maximum/minimum* (Abdelrahim et al., 2017; Deplano et al., 2021, 2022, 2023a; Lippi et al., 2023) and the *median* (Sanai Dashti et al., 2019; Vasiljevic et al., 2020; Deplano et al., 2023b).

However, aggregate metrics such as the average and median offer only a partial depiction of the distribution of local data. The maximum/minimum offers an improvement by establishing an upper bound on the distance between the local data and the average, but still falls short. In this paper, we are therefore interested in a more insightful metric: *variance*, which quantifies the degree of diversity across agents' data. Joint knowledge of average and variance has indeed proven useful in different applications, from measuring the thermal comfort in buildings (Niazi et al., 2020a,b), to monitoring tasks in traffic networks (Kawahara et al., 2013; Guo et al., 2015), and to validating performance of dynamic tracking protocols (Olfati-Saber and Shamma, 2005; Speranzon et al., 2008).

The **first contribution** of this paper consists in a fully distributed procedure to jointly estimate the average and variance of the local, private data. The procedure, which consists in cascading two (dynamic) average consensus protocols, even though quite intuitive, appears to be currently absent in the literature, to the best of our knowledge. Our implementation resorts to the Distributed Operator Theoretical (DOT) ADMM presented by us in (Bastianello et al., 2025), which has been shown to have superior performance with respect to other state-of-the-art protocols (Deplano et al., 2023b). The main features of the proposed algorithm are: linear convergence rate, null steady-state error, and robustness initial conditions.

After establishing the proposed joint mean and variance estimation algorithm, we discuss its usefulness in the emerging field of *personalized learning* (T. Dinh et al., 2020; Hanzely et al., 2023). A well-known concern in cooperative learning is the heterogeneity of the data collected by each agent, and the goal of personalization is mitigating this issue. The idea is to allow the agents to train local models tailored to the agents' data, instead of a global consensus model. This paradigm shift ensures that a local model has good predictive performance on local data, while still generalizing well, owing to the collaboration with other agents (Hanzely et al., 2023). In more practical terms, personalization is often accomplished by regularizing local loss functions with a term that depends on the other agents' models. The weight of the personalizing regularization thus determines the degree to which an agent tailors its local model (T. Dinh et al., 2020).

© 2025 the authors. This work has been accepted to IFAC for publication under a Creative Commons Licence CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041

Therefore, the **second contribution** of this paper is the application of the proposed mean/variance estimation algorithm to calibrate the personalization weights of each agent. As a byproduct of our interest for personalized learning, another contribution is the formulation of a new fully distributed, personalized set-up, which differs from the vast literature on federated set-ups that rely on a central coordinator, and the outline of a solution strategy that leverages an alternating minimization technique (Attouch et al., 2010).. Unlike our approach, in which each agent may locally decide the level of desired personalization, other distributed set-ups manage personalization by calibrating mixing weights between pairs of nodes, thus encouraging clustering of nodes with similar local models (Zantedeschi et al., 2020; Li et al., 2022).

Summarizing, the paper offers the following contributions:

- We propose a novel distributed variance estimation algorithm for peer-to-peer networks, with linear, exact convergence and robustness to re-initialization.
- We formalize a fully distributed set-up for personalized learning, discussing how to calibrate personalization through the variance across datasets.
- We showcase the performance of the estimation algorithm and its use in personalized learning with promising numerical results.

Paper's structure: Section 2 presents the novel distributed algorithm for joint mean and variance estimation, along with its convergence analysis and numerical simulations. Section 3 first formalizes the personalized learning problem in fully distributed systems, then outlines the use of mean and variance estimation for calibrating personalization, and finally discusses promising numerical results for a classification problem with the EMNIST dataset. Section 4 concludes the paper by outlining future lines of research.

Preliminaries on networks and graphs: We consider networks modeled by graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, n\}$ with $n \in \mathbb{N}$ is the set of *nodes*, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of *edges* connecting the nodes. A graph is said to be *undirected* if for any edge $(i, j) \in \mathcal{E}$ there is also $(j, i) \in \mathcal{E}$. An undirected graph $\mathcal{G}$ is said to be *connected* if there exists a sequence of adjacent edges in $\mathcal{E}$ between any pair of nodes $i, j \in \mathcal{V}$. The set of neighbors of the i-th node is denoted by $\mathcal{N}_i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$ and we denote by $\eta_i = |\mathcal{N}_i|$ the number of neighbors of node i.

Algorithm 1 Distributed joint mean/variance estimation

Initialization: Each agent $i \in \mathcal{V}$ arbitrarily initializes the auxiliary variables $\{y_{ij}(0), z_{ij}(0)\}_{j \in \mathcal{N}_i}$, choose the relaxation constant $\alpha \in (0, 1)$, and the penalty $\rho > 0$.

for $k = 1, 2, \dots$ **each active agent** $i \in \mathcal{V}$ applies the local updates

$$\mu_i(k) = \frac{u_i + \sum_{j \in \mathcal{N}_i} y_{ij}(k-1)}{1 + \rho \eta_i} \tag{1a}$$

$$\sigma_i^2(k) = \frac{(u_i - \mu_i(k))^2 + \sum_{j \in \mathcal{N}_i} z_{ij}(k-1)}{1 + \rho \eta_i} \tag{2a}$$

for each agent $j \in \mathcal{N}_i$ transmits the packets

$$p_{i \to j}(k) = 2\rho \mu_i(k) - y_{ij}(k-1) \tag{1b}$$

$$q_{i \to j}(k) = 2\rho \sigma_i^2(k) - z_{ij}(k-1) \tag{2b}$$

for each neighbor $j \in \mathcal{N}_i$ updates the auxiliary variables

$$y_{ij}(k) = (1 - \alpha) y_{ij}(k-1) + \alpha p_{j \to i}(k) \tag{1c}$$

$$z_{ij}(k) = (1 - \alpha) z_{ij}(k-1) + \alpha q_{j \to i}(k) \tag{2c}$$

2. DISTRIBUTED MEAN/VARIANCE ESTIMATION

Consider a network of n agents interacting according to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and having access to some local data $u_i \in \mathbb{R}$. The *consensus problem* consists in the design of proper local interaction rules to enable the estimation of a function of the local data $u_i \in \mathbb{R}$ stacked into $\boldsymbol{u} \in \mathbb{R}^n$. We are interested in the sample *variance* σ^2 and "average" μ of all data formally defined next:

$$\sigma^2 = \mathrm{var}(\boldsymbol{u}) := \frac{1}{n} \sum_{i=1}^{n} \big(u_i - \mu\big)^2, \quad \mu = \mathrm{avg}(\boldsymbol{u}) := \frac{1}{n} \sum_{i=1}^{n} u_i.$$

For simplicity, we consider here the base formulation of the problem for scalar data, but both the formulation of the problem and the algorithm proposed in the next section naturally extend to vector data $u_i \in \mathbb{R}^p$ by considering parallel consensus problems for each data entry.

2.1 Proposed solution via time-varying optimization

Let $k \in \mathbb{N}$ represent discrete instant of times at which the agents communicate and update their state. The proposed distributed protocol is implemented in Algorithm 1, which consists of two cascaded operations as in Fig. 1:

a) Each agent $i \in \mathcal{V}$ estimates the average of the constant data $u_i \in \mathbb{R}$ by running an average consensus protocol and stores its estimation in $\mu_i(k)$;
b) Each agent $i \in \mathcal{V}$ tracks the average of the time-varying data $(u_i - \mu_i(k))^2 \in \mathbb{R}$ – that is the variance of data u_i – by running a dynamic average consensus protocol and stores its estimation in $\sigma^2(k)$.

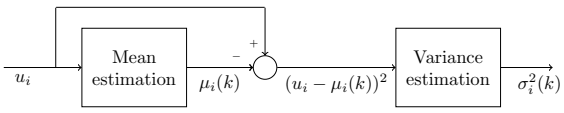


Fig. 1. Flowchart representing Algorithm 1.

Letting $s_i(k) \in \mathbb{R}$ be the signal of interest, each of the above consensus problems can be cast into a (time-varying) distributed optimization problem of the kind:

$$\begin{aligned} \boldsymbol{x}^\star(k) = \underset{x_1, \dots, x_n}{\mathrm{argmin}} \ & \sum_{i=1}^{n} \frac{1}{2} |x_i - s_i(k)|^2, \\ \text{s.t.} \quad & x_i = x_j \quad \forall (i, j) \in \mathcal{E}, \end{aligned} \tag{3}$$

where for the operation a) it must be used $s_i(k) = u_i$, and for operation b) it must be used $s_i(k) = (u_i - \mu_i(k))^2$. We have the following basic result.

Proposition 1. Consider a network of n agents interacting through to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and consider the optimization problem in eq. (3). If $\mathcal{G}$ is connected, the solution is $\boldsymbol{x}^\star(k) = \mu_k \mathbf{1}$ where $\mu_k = \mathrm{avg}(s_1(k), \dots, s_n(k))$, and $\mathbf{1}$ is the vector of ones.

Proof. The proof is trivial and can be carried out following the steps in (Deplano et al., 2023b, Proposition 1).

© 2025 the authors. Licensed under CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041

In Algorithm 1 the two consensus procedures for estimating the average and the variance are carried out by resorting to the DOT-ADMM Algorithm (Bastianello et al., 2021, 2025). More precisely, eqs. (1a)-(1b)-(1c) and (2a)-(2b)-(2c) are the explicit updates of the DOT-ADMM applied to the distributed optimization problem in the form of eq. (3) when $s_i(k) = u_i$ and $s_i(k) = (u_i - \mu_i(k))^2$, respectively (cfr. (Deplano et al., 2023b, Lemma 1) and (Bastianello et al., 2021, Section III-B)).

Remark 2. Even though the strategy outlined in Section 2.1 and depicted in Figure 1 could be implemented by using any dynamic average consensus algorithms, our choice to use DOT-ADMM lies in its superior performance and robustness features. We refer the interested reader to (Deplano et al., 2023b) for a comparison of the DOT-ADMM performance against other state-of-the-art dynamic average consensus protocols.

The following theorem proves the convergence of the proposed algorithm, which occurs with a linear rate and a null steady-state error.

Theorem 3. Consider a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ executing Algorithm 1 with local data $u_i \in \mathbb{R}$. If the graph $\mathcal{G}$ is connected, the root mean squared errors $e_\mu(k)$, $e_\sigma(k)$ on the estimation of the mean and variance, respectively, converge to zero with a linear rate.

Proof of Theorem 3: Let $\mathbf{1}$ and $\mathbf{0}$ denote the vectors of ones and zeros, respectively. Denote $\boldsymbol{\mu} = [\mu_i]_{i\in\mathcal{V}}$, $\boldsymbol{\sigma} = [\sigma_i]_{i\in\mathcal{V}}$ $\boldsymbol{y} = [y_{ij}]_{i\in\mathcal{V},j\in\mathcal{N}_i}$, $\boldsymbol{z} = [z_{ij}]_{i\in\mathcal{V},j\in\mathcal{N}_i}$. Moreover, define $A = \text{blk diag}\{\mathbf{1}_{\eta_i}\}_{i=1}^n$, $D = \text{blk diag}\{(\rho\eta_i)^{-1}\}_{i=1}^n$, and P as the permutation matrix swapping the ij-th and ji-th components of the $\boldsymbol{y}$ (or $\boldsymbol{z}$) vector. Then Algorithm 1 can be written in the compact form (Bastianello et al., 2025):

$$\boldsymbol{\mu}(k) = D^{-1}(\boldsymbol{u} + A^\top \boldsymbol{y}(k)) \tag{4a}$$
$$\boldsymbol{y}(k) = (1-\alpha)\boldsymbol{y}(k-1) - \alpha P\boldsymbol{y}(k-1) + 2\alpha\rho PA\boldsymbol{\mu}(k) \tag{4b}$$
$$\boldsymbol{\sigma}^2(k) = D^{-1}\left((\boldsymbol{u} - \boldsymbol{\mu}(k))^{\odot 2} + A^\top \boldsymbol{z}(k)\right) \tag{4c}$$
$$\boldsymbol{z}(k) = (1-\alpha)\boldsymbol{z}(k-1) - \alpha P\boldsymbol{z}(k-1) + 2\alpha\rho PA\boldsymbol{\sigma}^2(k) \tag{4d}$$

where $^{\odot}$ is the Hadamard (element-wise) exponent. The first two equations (4a), (4b) are used to compute the mean of the values in $\boldsymbol{u}$, while (4c), (4d) are used to compute the variance. Notice that $\boldsymbol{\mu}(k)$ feeds into (4c), thus creating a chain of two systems (cf. Fig. 1). Since $\boldsymbol{u}$ is constant, by (Bastianello et al., 2025) we know there exist $\zeta \in (0,1)$ and $C > 0$ such that:

$$d(\boldsymbol{y}(k)) \leq \zeta d(\boldsymbol{y}(k-1)), \text{ and } e_\mu(k) \leq C d(\boldsymbol{y}(k)),$$

where d is the distance function from the set of fixed points $\{\boldsymbol{y} \mid (I+P)\boldsymbol{y} = 2\rho\,\text{avg}(\boldsymbol{u})\mathbf{1}\}$. This implies that the error on the mean $e_\mu(k)$ converges to zero, i.e.,

$$\lim_{k\to\infty} e_\mu(k) = 0 \quad \Leftrightarrow \quad \lim_{k\to\infty} \mu_i(k) = \text{avg}(\boldsymbol{u}), \tag{5}$$

and it does so with a linear rate, completing the first part of the proof. The variance estimator tracks the time-varying quantity $\boldsymbol{v}(k) = (\boldsymbol{u} - \boldsymbol{\mu}(k))^{\odot 2}$ due to its dependence on the running estimate of the mean. Similarly, by (Deplano et al., 2023b)

$$d_k(\boldsymbol{z}(k)) \leq \zeta d_{k-1}(\boldsymbol{z}(k-1)) + C'\varepsilon(k) \tag{6a}$$
$$\left\|\boldsymbol{\sigma}^2(k) - \text{avg}(\boldsymbol{v}(k))\mathbf{1}\right\| \leq C d_k(\boldsymbol{z}(k)) \tag{6b}$$

for some $C', C > 0$, where d_k is the distance function from the set of time-varying fixed points $\{\boldsymbol{y} \mid (I+P)\boldsymbol{y} = 2\rho\,\text{avg}(\boldsymbol{v}(k))\mathbf{1}\}$ and where

$$\varepsilon(k) = \left(\text{avg}(\boldsymbol{v}(k)) - \text{avg}(\boldsymbol{v}(k-1))\right)^2.$$

which is such that

$$\lim_{k\to\infty} \varepsilon(k) = \left(\sum_{i=1}^n (u_i - \text{avg}(\boldsymbol{u}))^2 - (u_i - \text{avg}(\boldsymbol{u}))^2\right)^2 = 0.$$

Iterating (6a) yields

$$d_k(\boldsymbol{z}(k)) \leq \zeta^k d_0(\boldsymbol{z}(0)) + C' \sum_{h=0}^{k-1} \zeta^{k-h}\varepsilon(k).$$

Since $\varepsilon(k)$ decays, then by (Sundhar Ram et al., 2010, Lemma 3.1(a)) it holds

$$\lim_{k\to\infty} \sum_{h=0}^{k-1} \zeta^{k-h}\varepsilon(k) = 0,$$

Thus, from eq. (6) follows $\lim_{k\to\infty} d_k(\boldsymbol{z}(k)) = 0$ and also

$$\begin{aligned}\lim_{k\to\infty} e_\sigma(k) &= \lim_{k\to\infty} \left\|\boldsymbol{\sigma}^2(k) - \text{var}(\boldsymbol{u})\mathbf{1}\right\| \\ &= \lim_{k\to\infty} \left\|\boldsymbol{\sigma}^2(k) - \text{avg}((\boldsymbol{u} - \boldsymbol{\mu}(k))^{\odot 2})\mathbf{1}\right\| = 0.\end{aligned}$$

Together with eq. (5), this completes of the proof. □

2.2 Numerical simulations

In this section we evaluate the performance of Algorithm 1 for distributed mean and variance estimation. The simulations were implemented using `tvopt` (Bastianello, 2021).

We start by displaying in Figure 2(left) the error trajectories of Algorithm 1 for both the mean and the variance, that is $\|\boldsymbol{\mu}(k) - \text{avg}(\boldsymbol{u})\mathbf{1}\|$ and $\left\|\sigma^2(k) - \text{var}(\boldsymbol{u})\mathbf{1}\right\|$. The results are derived by applying the algorithm to a random geometric graph with $n = 25$ nodes and mean degree 12. The data are randomly drawn according to $u_i \sim \mathcal{N}(0, 10)$, and we set the parameters $\rho = 1$, $\alpha = 0.5$. As we can see, both the mean and the variance are correctly computed by Algorithm 1, except for errors due to the numerical precision. We remark that the variance estimation seems to be more sensitive to rounding errors, an aspect we will explore in future research.

We conclude by also testing Algorithm 1 to track the mean and variance of time-varying data $\{\boldsymbol{u}(k)\}_{k\in\mathbb{N}}$, on a random graph. The data changes every 100 iterations and are drawn as $u_i \sim \mathcal{N}(0, 10)$. The tracking errors depicted in Figure 2(right) show that both the mean and variance estimates converge to the true mean and variance while the data remain unchanged, thus showing robustness to re-initialization.

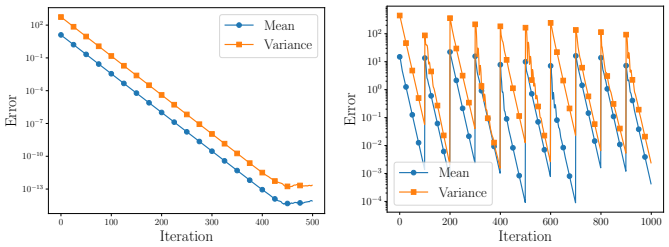


Fig. 2. Error trajectories of Algorithm 1 on a random geometric graph with $n = 25$: (left) constant data u_i; (right) time-varying data $u_i(k)$.

© 2025 the authors. Licensed under CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041

3. APPLICATION TO PERSONALIZED LEARNING

In distributed learning, each agent $i \in \mathcal{V}$ holds a local data set $\{\boldsymbol{a}_{i,h}, \boldsymbol{b}_{i,h}\}_{h=1}^{m_i}$ of dimension $m_i \in \mathbb{N}$ and aims at estimating the parameters of the common regression model g such that the sum of local partial costs f_i is minimized:

$$f_i(\boldsymbol{x}) = \sum_{h=1}^{m_i} g(\boldsymbol{x}, \boldsymbol{a}_{i,h}, \boldsymbol{b}_{i,h}).$$

In other words, the problem is that of making the agents cooperatively train a model $\boldsymbol{x}^\star_{\text{DIS}}$ common for everyone by exploiting all data across the datasets, hopefully obtaining a better model than the local ones:

$$\boldsymbol{x}^\star_{\text{DIS}} = \operatorname*{argmin}_{\boldsymbol{x}} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}).$$

This problem can be equivalently formulated as a distributed optimization problem as follows:

$$\min_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_n} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{x}_i) \tag{7a}$$

$$\text{s.t.} \quad \boldsymbol{x}_i = \boldsymbol{x}_j \text{ if } (i,j) \in \mathcal{E} \tag{7b}$$

In many applications there may be an interest in providing *personalization*: the model trained by each agent is tailored according to the specific data that it stores. One method of implementing personalization is by regularizing the costs. In particular, we choose to reformulate the problem as

$$(\boldsymbol{x}^\star_{\text{PER}}, \boldsymbol{y}^\star_{1,\text{PER}}, \cdots, \boldsymbol{y}^\star_{n,\text{PER}}) = \operatorname*{argmin}_{\boldsymbol{x},\boldsymbol{y}_1,\dots,\boldsymbol{y}_n} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{y}_i) + \frac{\lambda_i}{2}\|\boldsymbol{y}_i - \boldsymbol{x}\|^2,$$

whose equivalent distributed formulation is given by

$$\min_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_n,\boldsymbol{y}_1,\dots,\boldsymbol{y}_n} \sum_{i\in\mathcal{V}} f_i(\boldsymbol{y}_i) + \frac{\lambda_i}{2}\|\boldsymbol{y}_i - \boldsymbol{x}_i\|^2, \tag{8a}$$

$$\text{s.t.} \qquad \boldsymbol{x}_i = \boldsymbol{x}_j \text{ if } (i,j) \in \mathcal{E}. \tag{8b}$$

This novel problem formulation uses the regularization terms to keep locally trained models $\boldsymbol{y}^\star_{i,\text{PER}}$ close to the global model $\boldsymbol{x}^\star_{\text{PER}}$, which is, in general, different from the global model $\boldsymbol{x}^\star_{\text{DIS}}$ obtained without personalization. To the best of our knowledge, personalization techniques have been studied exclusively in the federated set-up, while problem (8) can be solved in a fully distributed fashion. For instance, the concept of personalization we employ is similar to that of (T. Dinh et al., 2020), but differs in two crucial points: 1) the personalized problem in (T. Dinh et al., 2020) is formulated as a bi-level optimization problem, while our formulation is a direct extension of the consensus optimization problem (7); 2) we allow for *uncoordinated* personalization weights, i.e., λ_i are heterogeneous, while in (T. Dinh et al., 2020) it is assumed that they are all equal to a common value of $\lambda = \lambda_i$. Choosing uncoordinated weights allows the agents to tune their desired degree of personalization, indeed:

- *Complete personalization*: if $\lambda_i = 0$, then agent i only relies on its own data to train its local model.
- *No personalization*: if $\lambda_i = \infty$, then agent i fully collaborates in order to train a common model with other agents. If all agents set $\lambda_i = \infty$, problem (8) reduces to problem (7).
- *Partial personalization*: if $\lambda_i \in (0, \infty)$, then agent i partially collaborates with other agents to train both a common model and a personalized model.

3.1 Solving the personalized problem

The solution of (8) is not an easy task, as the cost function depends on both the interconnected variables $\boldsymbol{x}$ and $\boldsymbol{y}$, but the consensus constraints only apply to $\boldsymbol{y}$. Therefore, standard distributed optimization algorithms cannot be directly applied. In the following we discuss a possible approach to the solution of (8), while a more in-depth analysis is left to future work. The idea is to apply the divide-and-conquer heuristic of *alternating minimization* (Attouch et al., 2010). That is, we alternate between a minimization in terms of $\boldsymbol{x}$ with a fixed value of $\boldsymbol{y}$, and vice versa. In particular, letting

$$h(\boldsymbol{x}, \boldsymbol{y}) := \sum_{i\in\mathcal{V}} f_i(\boldsymbol{y}_i) + \frac{\lambda_i}{2}\|\boldsymbol{y}_i - \boldsymbol{x}_i\|^2,$$

we apply

$$\boldsymbol{y}(k+1) = \operatorname*{argmin}_{\boldsymbol{y}} h(\boldsymbol{x}(k), \boldsymbol{y}) \tag{9a}$$

$$\boldsymbol{x}(k+1) = \operatorname*{argmin}_{\boldsymbol{x}} h(\boldsymbol{x}, \boldsymbol{y}(k+1)) \quad \text{s.t. } \boldsymbol{x}_i = \boldsymbol{x}_j \text{ if } (i,j) \in \mathcal{E}. \tag{9b}$$

Since $h(\boldsymbol{x}, \boldsymbol{y})$ is separable in $\boldsymbol{y}$, then the minimization (9a) can be carried out independently by each agent, namely,

$$\boldsymbol{y}_i(k+1) = \operatorname*{argmin}_{\boldsymbol{y}_i} f_i(\boldsymbol{y}_i) + \frac{\lambda_i}{2}\|\boldsymbol{y}_i - \boldsymbol{x}_i(k)\|^2. \tag{10}$$

Problem (10) may have a closed form solution, as in the case of the quadratic costs in Section 3. But if this is not the case, the agents need to approximate its solution by the use of *e.g.* gradient descent. Given the solution to eq. (9a), the term $f_i(\boldsymbol{y}_i(k+1))$ in $h(\boldsymbol{x}, \boldsymbol{y}_i(k+1))$ becomes constant, thus the update in eq. (9b) reduces to

$$\{\boldsymbol{x}_i(k+1)\}_{i\in\mathcal{V}} = \operatorname*{argmin}_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_n} \sum_{i\in\mathcal{V}} \frac{\lambda_i}{2}\|\boldsymbol{y}_i(k+1) - \boldsymbol{x}_i\|^2 \quad \text{s.t. } \boldsymbol{x}_i = \boldsymbol{x}_j \text{ if } (i,j) \in \mathcal{E}. \tag{11}$$

In what follows, we choose to solve the above problem by resorting DOT-ADMM (Bastianello et al., 2025), and leave to future work the exploration of alternatives such as gradient tracking.

3.2 Choice of personalized weights via variance estimation

The strategy we propose for selecting appropriate personalized weights can be outlined in three main steps:

1) *Local training*: Each agent independently trains its own model $\boldsymbol{y}^\star_{i,\text{LOC}}$ without considering the models of other nodes:

$$\boldsymbol{y}^\star_{i,\text{LOC}} = \operatorname*{argmin}_{\boldsymbol{y}} f_i(\boldsymbol{y}); \tag{12}$$

2) *Model distribution analysis*: The agents collaborate to assess the similarity between their local models, thus allowing them to gauge the degree of alignment or discrepancy among their respective models. In particular, the agents cooperatively compute the (component-wise) average $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$ of the models by executing Algorithm 1.

3) *Personalized learning*: Finally, nodes integrate insights from other models to determine a personalized model that encapsulates the combined knowledge and nuances identified across the network. Each agent sets λ_i based

© 2025 the authors. Licensed under CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041

Table 1. Accuracy comparison across all test sets for different levels of personalization.

	Param.	Model	Min	Mean	Max
Global model without personalization		$\boldsymbol{x}^\star_{\text{DIS}}$	**98.1%**	98.1%	98.1%
Global model with personalization	$\forall w \geq 0$	$\boldsymbol{x}^\star_{\text{PER}}$	**98.1%**	98.1%	98.1%
Local model with personalization	$w = 10$	$\boldsymbol{y}^\star_{i,\text{PER}}$	97.6%	98.1%	98.6%
Local model with personalization	$w = 1$	$\boldsymbol{y}^\star_{i,\text{PER}}$	96.2%	**98.3%**	**99.5%**
Local model with personalization	$w = 0.1$	$\boldsymbol{y}^\star_{i,\text{PER}}$	94.3%	98.1%	**100%**
Local model without cooperation		$\boldsymbol{y}^\star_{i,\text{LOC}}$	59.5%	85.0%	94.3%

on the distance of its own local model from the mean, saturated by the standard deviation:

$$\lambda_i = \frac{w}{\text{avg}(\max\{\mathbf{0}, \left|\boldsymbol{y}^\star_{i,\text{LOC}} - \boldsymbol{\mu}\right| - \boldsymbol{\sigma}\})}, \tag{13}$$

where $w \geq 0$ is an arbitrary tuning parameter.

The larger the distance of $\boldsymbol{y}^\star_{i,\text{LOC}}$ from the average $\boldsymbol{\mu}$, the smaller the regularization weight $\lambda_i \to 0$, which means that more personalization is applied. If instead, the distance is lesser than the standard deviation, no personalization is applied, i.e., $\lambda_i = \infty$. In the latter case, agent i can directly impose a local constraint $\boldsymbol{y}_i = \boldsymbol{x}_i$ instead of $\lambda_i = \infty$, i.e., its personalized model $\boldsymbol{y}^\star_{i,\text{PER}}$ coincides with the global model $\boldsymbol{x}^\star_{\text{PER}}$.

3.3 Numerical results on the EMNIST dataset

We provide a numerical simulation by considering the well-known EMNIST database, an extension (E) of the modified (M) version of the National Institute of Standards and Technology (NIST) database, containing digitalized black-and-white pictures (28×28 pixels) of handwritten digits and uppercase/lowercase handwritten letters. The EMNIST dataset presents the data in different separate data hierarchies, among which is the "*by author*" hierarchy, containing the segmented character classes organized by writer.

Our simulation takes into consideration a classification task between two digits that could be confused depending on the specific handwriting, which are the number 3 and the number 8. We take into consideration only those writers for which there are at least 15 labelled entries in the training set (i.e., $m_i \geq 15$) and at least 5 entries in the test set, yielding to a total number of $n = 40$ writers. Finally, we consider the classic linear regression model as a common regressor. For each writer $i \in \{i, \dots, n\}$ and picture $h \in \{1, \dots, m_i\}$, the vector $a_{i,h} \in [0,1]^p$ (where $p = 28 \cdot 28 = 784$) is the vector describing the BW level of each pixel, row by row, of the corresponding picture, while $b_{i,h} \in \{-1, +1\}$ is a scalar denoting that the picture is either a 3 (when equal to -1) or an 8 (when equal to $+1$). Let $A_i \in \mathbb{R}^{m_i \times p}$ and $b_i \in \mathbb{R}^{m_i}$ be the matrix and vector constructed by concatenating all features $a_{i,h}$ and labels $b_{i,h}$, then the local cost reads as $f_i(\boldsymbol{y}) = \|A_i \boldsymbol{y} - b_i\|^2$.

Let us now discuss the accuracy of the models on the test sets for all cases of personalization detailed in Table 1.

Complete personalization: When the writers train their own model $\boldsymbol{y}^\star_i$ based on their local dataset only as in eq. (12), they obtain a model that is quite performing on their own test set but is defective when tasked in distinguishing digits written by others. Indeed, about half of the writers' models achieves an accuracy on the joint test set lower than 85%, the average accuracy across all agents models. Moreover, the accuracy of each local model is at most 94% with worst cases as low as 59%.

No personalization: Consider now the case in which the writers fully cooperate to train a model that better fits all the datasets, i.e., they cooperate to solve problem (7) in a distributed way. As might be expected, the common model on which the writers agree upon has much better performance on the joint test set when compared to the local models, achieving an accuracy of 98.1%.

Partial personalization: We now compare these results with those obtained through a personalized learning approach, in which the writers train a personalized model by also exploiting the information received by the other authors. This is done by solving the distributed optimization problem in eq. (8) where the personalized weights (13) are chosen exploiting the mean and variance among the local trained models $\boldsymbol{y}^\star_i$ computed through Algorithm 1. When the tuning parameter is selected equal to $w = 1$, the average accuracy is 98.3%, slightly higher than the accuracy obtained without any personalization. Moreover, there are writers achieving an even higher accuracy of 99.4%, while the worst accuracy is about 96.2%, much higher than the accuracy obtained with complete personalization. It is also interesting to see that when the tuning parameter is selected to $w = 0.1$ some of the writers obtain a model with an accuracy of 100%, while the average accuracy across the writers remains equal to the standard distributed optimization without personalization.

4. DISCUSSION AND FUTURE PERSPECTIVES

We have presented a novel personalized learning approach in a fully distributed set-up, where personalization is calibrated by exploiting an original algorithm to jointly compute the average and variance across the local, private datasets. Numerical simulations indicate that agents are motivated to employ the proposed personalized cooperative learning approach for two main reasons: 1) the expected accuracy of the personalized model remains greater than or equal to the expected accuracy without personalization, while the maximum achievable accuracy increases up to 100%; 2) the agents jointly compute a global model whose accuracy is comparable with the global model derived without personalization, which remains as a valuable alternative for the agents.

This work sets the stage for several promising lines of future research. By leveraging the distributed variance estimation algorithm, agents could autonomously assess the coherence of their own datasets, determining the extent to which they should rely on information from other agents' datasets by appropriately calibrating the parameters of

© 2025 the authors. Licensed under CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041

personalized learning. Additionally, personalized learning could help in mitigating the impact of malicious behavior exhibited by some agents attempting to make the global model diverge far from the optimal solution. Furthermore, it could be interesting to consider real-world scenarios where local datasets may undergo updates over time and agents may join into or depart from the learning process.

REFERENCES

Abdelrahim, M., Hendrickx, J., and Heemels, W. (2017). Max-consensus in open multi-agent systems with gossip interactions. In *IEEE 56th Conference on Decision and Control*, 4753–4758.

Attouch, H., Bolte, J., Redont, P., and Soubeyran, A. (2010). Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the kurdyka-łojasiewicz inequality. *Mathematics of operations research*, 35(2), 438–457.

Bastianello, N. and Carli, R. (2022). ADMM for dynamic average consensus over imperfect networks. In *IFAC–PapersOnLine*, volume 55, 228–233.

Bastianello, N., Carli, R., Schenato, L., and Todescato, M. (2021). Asynchronous distributed optimization over lossy networks via relaxed ADMM: Stability and linear convergence. *IEEE Transactions on Automatic Control*, 66(6), 2620–2635.

Bastianello, N. (2021). tvopt: A python framework for time-varying optimization. In *IEEE 60th Conference on Decision and Control*, 227–232.

Bastianello, N., Deplano, D., Franceschelli, M., and Johansson, K.H. (2025). Robust online learning over networks. *IEEE Transactions on Automatic Control*, 70(2), 933–946.

Deplano, D., Franceschelli, M., and Giua, A. (2022). Dynamic max–consensus with local self–tuning. *IFAC–PapersOnLine*, 55(13), 127 – 132. doi: 10.1016/j.ifacol.2022.07.247.

Deplano, D., Franceschelli, M., and Giua, A. (2023a). Dynamic min and max consensus and size estimation of anonymous multiagent networks. *IEEE Transactions on Automatic Control*, 68(1), 202–213.

Deplano, D., Bastianello, N., Franceschelli, M., and Johansson, K.H. (2023b). A unified approach to solve the dynamic consensus on the average, maximum, and median values with linear convergence. In *IEEE 62nd Conference on Decision and Control*, 6442–6448. IEEE.

Deplano, D., Franceschelli, M., and Giua, A. (2021). Distributed tracking of graph parameters in anonymous networks with time-varying topology. In *IEEE 60th Conference on Decision and Control*, 6258–6263. IEEE.

Freeman, R.A., Yang, P., and Lynch, K.M. (2006). Stability and convergence properties of dynamic average consensus estimators. In *IEEE 45th Conference on Decision and Control*, 398–403.

Guo, J., Huang, W., and Williams, B.M. (2015). Real time traffic flow outlier detection using short-term traffic conditional variance prediction. *Transportation Research Part C: Emerging Technologies*, 50, 160–172.

Hanzely, F., Zhao, B., and Kolar, M. (2023). Personalized Federated Learning: A Unified Framework and Universal Optimization Techniques. *Transactions on Machine Learning Research*.

Kawahara, R., Takine, T., Mori, T., Kamiyama, N., and Ishibashi, K. (2013). Mean–variance relationship of the number of flows in traffic aggregation and its application to traffic management. *Computer Networks*, 57(6), 1560–1576.

Kia, S.S., Van Scoy, B., Cortes, J., Freeman, R.A., Lynch, K.M., and Martinez, S. (2019). Tutorial on dynamic average consensus: The problem, its applications, and the algorithms. *IEEE Control Systems*, 39(3), 40–72.

Li, S., Zhou, T., Tian, X., and Tao, D. (2022). Learning to collaborate in decentralized learning of personalized models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9766–9775.

Lippi, M., Furchi, A., Marino, A., and Gasparri, A. (2023). An adaptive distributed protocol for finite–time infimum or supremum dynamic consensus. *IEEE Control Systems Letters*, 7, 401–406.

Moreau, L. (2005). Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2), 169–182. doi: 10.1109/TAC.2004.841888.

Niazi, M.U.B., Canudas-de Wit, C., and Kibangou, A.Y. (2020a). State variance estimation in large-scale network systems. In *IEEE 59th Conference on Decision and Control*, 6052–6057. IEEE.

Niazi, M.U.B., Canudas-de Wit, C., and Kibangou, A.Y. (2020b). Thermal monitoring of buildings by aggregated temperature estimation. *IFAC-PapersOnLine*, 53(2), 4132–4137.

Olfati-Saber, R. and Shamma, J.S. (2005). Consensus filters for sensor networks and distributed sensor fusion. In *IEEE 44th Conference on Decision and Control*, 6698–6703. IEEE.

Sanai Dashti, Z.A.Z., Seatzu, C., and Franceschelli, M. (2019). Dynamic consensus on the median value in open multi–agent systems. In *IEEE 58th Conference on Decision and Control*, 3691–3697.

Speranzon, A., Fischione, C., Johansson, K.H., and Sangiovanni-Vincentelli, A. (2008). A distributed minimum variance estimator for sensor networks. *IEEE Journal on Selected Areas in Communications*, 26(4), 609–621.

Sundhar Ram, S., Nedić, A., and Veeravalli, V.V. (2010). Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of Optimization Theory and Applications*, 147(3), 516–545.

T. Dinh, C., Tran, N., and Nguyen, J. (2020). Personalized Federated Learning with Moreau Envelopes. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, 21394–21405. Curran Associates, Inc.

Vasiljevic, G., Petrovic, T., Arbanas, B., and Bogdan, S. (2020). Dynamic median consensus for marine multi–robot systems using acoustic communication. *IEEE Robotics and Automation Letters*, 5(4), 5299 – 5306.

Zantedeschi, V., Bellet, A., and Tommasi, M. (2020). Fully decentralized joint learning of personalized models and collaboration graphs. In *International Conference on Artificial Intelligence and Statistics*, 864–874. PMLR.

Zhu, M. and Martínez, S. (2010). Discrete–time dynamic average consensus. *Automatica*, 46(2), 322–329.

© 2025 the authors. Licensed under CC-BY-NC-ND. DOI: 10.1016/j.ifacol.2025.07.041