Proceedings of Machine Learning Research vol 283:1–13, 2025 7th Annual Conference on Learning for Dynamics and Control

Controlling Participation in Federated Learning with Feedback

Michael Cummins[1] M.CUMMINS24@IMPERIAL.AC.UK
Guner Dilsad Er[2] GDER@TUE.MPG.DE
Michael Muehlebach[2] MICHAEL.MUEHLEBACH@TUE.MPG.DE
[1]*Department of Electrical and Electronic Engineering, Imperial College London*
[2]*Max Planck Institute for Intelligent Systems, Tübingen, Germany*

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

We address the problem of client participation in federated learning, where traditional methods typically rely on a random selection of a small subset of clients for each training round. In contrast, we propose FedBack, a deterministic approach that leverages control-theoretic principles to manage client participation in ADMM-based cross-silo federated learning. FedBack models client participation as a discrete-time dynamical system and employs an integral feedback controller to adjust each client's participation rate individually, based on the client's optimization dynamics. We provide global convergence guarantees for our approach by building on the recent federated learning research. Numerical experiments on federated image classification demonstrate that FedBack achieves up to 50% improvement in communication and computational efficiency over algorithms that rely on a random selection of clients.

Keywords: Federated Learning, Distributed Optimization, Client Participation Control, Event-Triggered Communication, ADMM, Feedback Control, Communication Efficiency

1. Introduction

The rapid growth of data generation across numerous devices has created new challenges for modern machine learning. Traditional centralized training methods, which involve aggregating data from individual devices into a central server for model training, are increasingly impractical due to both privacy concerns and the heavy communication costs associated with transferring large volumes of data. Federated learning (FL), a term coined by McMahan et al. (2017), provides a decentralized solution by allowing devices to collaboratively train machine learning models without the need to share raw data. Instead, each device trains a local model on its own data and transmits only updated model parameters to a central server or directly to other devices, thus addressing privacy concerns and facilitating large-scale learning across networks of distributed devices.

While FL effectively mitigates privacy risks, it introduces significant challenges in communication efficiency. The frequent exchange of model parameters between a central server and participating devices over potentially unreliable, bandwidth-constrained networks can be both energy-intensive and costly. This challenge is further amplified in large-scale networks, where high communication demands lead to increased operational costs and latency (Li et al., 2020b). To address these issues, methods like FedAvg (McMahan et al., 2017) and FedProx (Li et al., 2020c), and many more, have explored strategies such as random sampling of training clients to reduce communication overhead, offering partial solutions to this pressing concern.

A promising approach to further reduce communication load in FL is event-triggered communication, where updates are transmitted only when significant changes occur in local model parameters (Er et al., 2024). Building on the sent-on-delta approach proposed by Miskowicz (2006), we conceptualize client participation as an event-based control system (Aström, 2008; Muehlebach and Trimpe, 2017), where clients participate only when necessary. Event-based methods, such as (Er et al., 2024; Zehtabi et al., 2022), leverage peer-to-peer communication in decentralized settings with event-triggered mechanisms. These approaches achieve communication savings by transmitting model parameters only when certain conditions are met, such as changes in model parameters exceeding a fixed threshold. Additionally, Chen et al. (2018) adopt an event-triggered framework where lagged gradients are adaptively reused to reduce communication load. Building on these ideas, our strategy integrates event-triggered communication into an Alternating Direction Method of Multipliers (ADMM) framework, providing robustness to heterogeneity in local data distributions (Er et al., 2024; Zhou and Li, 2023). Unlike existing event-based methods, which rely on static thresholds, our approach emphasizes the need for dynamically tuned thresholds that adapt to system state, network variability, and application-specific requirements, offering greater flexibility and efficiency.

Non-i.i.d. data distribution across clients creates a fundamental challenge in distributed learning by causing significant discrepancies among local datasets, which impairs the ability of individual models to generalize to a global solution. This divergence often results in slower convergence and suboptimal (global) models. Various approaches have been proposed to address this issue. Methods such as (Li et al., 2020a; Acar et al., 2021; Shi et al., 2023) introduce proximal regularization terms to local objectives, which bias local updates toward the global model and reduce client drift. Similarly, ADMM-based methods, such as FedPD (Zhang et al., 2021) and FedADMM (Zhou and Li, 2023; Wang et al., 2022; Gong et al., 2022), align local models with global objectives through structured optimization formulations, though they differ in their strategies for client participation. Alternative splitting-based methods, such as the variant of Douglas-Rachford splitting proposed by Tran Dinh et al. (2021), similarly align local and global objectives but remain constrained by random client participation. Meanwhile, FedNova (Wang et al., 2020b) addresses the inconsistency in server-side weighted aggregations, improving the alignment of local and global objectives. On another front, Karimireddy et al. (2020) introduce SCAFFOLD, which uses client control variates, serving as dual variables, to improve convergence in non-i.i.d. settings. However, SCAFFOLD comes with the trade-off of doubling communication costs, as it requires clients to exchange two variables, increasing the burden compared to other methods for the same participation rate.

In this paper, we introduce *FedBack*, a novel FL framework that addresses these limitations through an adaptive thresholding mechanism designed to actively control device participation. Rather than relying on a fixed communication threshold, *FedBack* dynamically adjusts the threshold δ based on factors such as network conditions, model accuracy requirements, and device capabilities. This tuning process allows *FedBack* to balance the trade-off between communication cost and model convergence more effectively than fixed-threshold methods, achieving communication savings without compromising model performance. By incorporating an event-based communication strategy into an ADMM framework, *FedBack* also enables joint optimization over both primal and dual variables, providing robustness against data and network heterogeneity.

Our approach is inspired by recent advances in the optimization literature that model algorithms as dynamical systems (Muehlebach and Jordan, 2020; Tong and Muehlebach, 2023; Nishihara et al., 2015; Dörfler et al., 2024; Wibisono et al., 2016; Kolev et al., 2023). The key idea is to interpret the sequence generated by an optimization algorithm as the trajectory of a dynamical system. For instance, Dörfler et al. (2024) provides a systems-theoretic perspective on algorithms, illustrating examples such as interpreting primal-dual algorithms as proportional-integral controllers. Moreover, it is common to use similar mathematical tools, such as Lyapunov functions, for understanding and designing optimization algorithms with robust performance guarantees (Rawlings et al., 2017; Nishihara et al., 2015; Lessard et al., 2016). Our contributions are as follows:

i) We propose *FedBack*, a novel cross-silo FL algorithm that dynamically controls client participation by adjusting the communication threshold δ based on network conditions.
ii) We establish global stability conditions for the feedback control law and provide global convergence guarantees for *FedBack*.
iii) Extensive numerical experiments are provided and illustrate *FedBack* achieving substantial savings in both client communication and computation when compared to baseline methods, such as FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020c), FedADMM (Zhou and Li, 2023), in communication efficiency and classification accuracy across various benchmark datasets.

By providing a mechanism for adaptive participation control in FL, *FedBack* addresses the core scaling challenges of large-scale, distributed learning (such as limited communication and computation resources) while ensuring robustness in non-i.i.d. settings. This makes *FedBack* well-suited for deployment in real-world, bandwidth-constrained environments with varying data distributions.

The article is organized as follows: Sec. 2 outlines the problem statement of learning from decentralized data via ADMM. A solution via feedback control is presented in Sec. 3 and Sec. 4 contains the theoretical analysis. Sec. 5 provides a numerical study detailing the performance of *FedBack* in two challenging benchmarks. We then conclude the paper in Sec. 6 with final remarks and a comment on future work.

2. Problem Statement

We consider the optimization problem,

$$\min_{\theta \in \mathbb{R}^d} \sum_{i=1}^{N} f_i(\theta), \tag{2.1}$$

where N is the number of clients in the distributed network and $f_i : \mathbb{R}^d \to \mathbb{R}$ is the loss function of client i with dataset D_i. Each function f_i has a Lipschitz continuous gradient with Lipschitz constant r_i.

We formulate an equivalent problem as

$$\min_{\{\theta_i\}_{1:N}, \omega} \sum_{i=1}^{N} f_i(\theta_i) \quad \text{s.t.} \quad \theta_i = \omega \;\; \forall i \in \{1, \dots, N\}, \tag{2.2}$$

and solve (2.2) with ADMM (Boyd et al., 2010) via the following dynamics,

$$\lambda_i^{k+1} = \lambda_i^k + \theta_i^k - \omega^k, \quad \theta_i^{k+1} = \operatorname*{argmin}_{\theta} f_i(\theta) + \frac{\rho}{2} |\theta - \omega^k + \lambda_i^{k+1}|^2, \tag{2.3}$$

$$\omega^{k+1} = \frac{1}{N} \sum_{i=1}^{N} \left(\theta_i^{k+1} + \lambda_i^{k+1} \right), \tag{2.4}$$

where $\rho > 0$ is the proximal parameter, $|\cdot|$ denotes the Euclidean distance and $\lambda_i^0 = 0$. We note that the ADMM dynamics elicit a communication network where each client i in the network downloads server parameters ω^k at round k, performs local computation to update local parameters $(\lambda_i^{k+1}, \theta_i^{k+1})$ via (2.3)[1], and uploads $\lambda_i^{k+1} + \theta_i^{k+1}$ to the server to enable updates to the server parameters (2.4). However, it is well known that distributed optimization schemes such as ADMM do not scale well to cross-device or even cross-silo FL problems (Zhou and Li, 2023; McMahan et al., 2017; Er et al., 2024; Li et al., 2022). We therefore follow a similar approach to (Er et al., 2024) by only enforcing client participation when a certain event is triggered. As a result, we demonstrate dramatic improvement in communication and computation efficiency to vanilla ADMM while also maintaining global convergence guarantees.

3. Federated Optimization with Controlled Participation

A participation event in ADMM takes place when the server sends the global parameters to a client i and client i sends its parameters to the server after doing some local computation, leading to N communication events at each round k. This is usually guaranteed in base-splitting schemes for distributed optimization (Ryu and Yin, 2022), and motivates federated optimization to reduce the number of events necessary to reach a (local) minimizer of (2.1). We address this issue by only selecting a client for participation if the difference between their local variables (λ_i, θ_i) and the server parameters ω exceeds a certain allowance $\delta \in \mathbb{R}_+$. Specifically, we define $z_i := \theta_i + \lambda_i$ and the identifier function

$$S_i^k(\delta) := \begin{cases} 1, & \text{if } |\omega^k - z_i^{\text{prev}}| \geq \delta, \\ 0, & \text{else}, \end{cases} \tag{3.1}$$

where z_i^{prev} is the most recent $z_i = \lambda_i + \theta_i$ received from client i. Thus, the identifier $S_i^k(\delta_i) = 1$ if and only if client i is selected to participate at round k. We note that with $\delta = 0$ we retrieve the original group consensus ADMM and $\delta \geq \delta_+$ (see Lemma 1 in Sec. 4) would create a deadlock in the network where no parameters would be updated. Our goal is therefore to find a sequence $\{\delta_i^k\}_{k=0}^{T-1}$ for each client i such that

$$\lim_{T\to\infty} \frac{1}{T} \sum_{k=0}^{T-1} S_i^k(\delta_i^k) = \bar{L}_i, \tag{3.2}$$

where T is the total number of rounds necessary to reach a locally optimal solution of (2.1) and $\bar{L}_i$ is the desired participation rate of client i over the course of the federated optimization procedure. We note that $\bar{L}_i$ may vary between clients as some may be willing to participate more/less frequently if they have access to more/less compute. In the forthcoming analysis, we assume $\bar{L}_i$ may differ between clients but only evaluate our method numerically for identical $\bar{L}_i$ in Sec. 5.

In the following, we propose a feedback control law that achieves (3.2), where we consider δ_i to be the control input and $S_i^k(\delta_i^k)$ measurement output. Our feedback controller is as follows:

$$\delta_i^{k+1} = \delta_i^k + K(L_i^k - \bar{L}_i), \tag{3.3}$$

1. It is common (necessary for FedAvg (McMahan et al., 2017)) to warm start the optimization problem (2.3) with the server parameters ω^k received at that round. Although this is not a necessity in ADMM, it generally demonstrates superior empirical performance.

Algorithm 1 Computing $\mathcal{S}_c$

Given: $\mathcal{I} = \{1, \ldots, N\}$, $\delta_i^k, \omega^k, z_i^{\text{prev}}$
Initialize: $I_{\text{s}}^k = \emptyset$, $\delta_i^0 = 0 \;\forall i \in \mathcal{I}$
for each client $i \in \mathcal{I}$ **do**
  if $S_i^k(\delta_i^k) = 1$ **then**
    $I_{\text{s}}^k \leftarrow I_{\text{s}}^k \cup i$
  end if
  $L_i^{k+1} \leftarrow (1-\alpha)L_i^k + \alpha S_i^k(\delta_i^k)$ {Compute current running load for client "i"}
  $\delta_i^{k+1} \leftarrow \delta_i^k + K(L_i^k - \bar{L}_i)$ {Update threshold (control input)}
end for
return I_{s}^k

where $K \in \mathbb{R}_+$ is the control gain and L_i^{k+1} is the output of a low pass filter applied to $S_i^k(\delta_i^k)$:

$$L_i^{k+1} = (1-\alpha)L_i^k + \alpha S_i^k(\delta_i^k), \tag{3.4}$$

with time constant $\alpha \in (0, 1)$ (typically $\alpha \approx 0.9$). The value of $L_i^k \in [0, 1]$ therefore represents an estimate of how much client i has communicated up to round k. Choosing a higher value for α will then emphasize recent measurements when computing L_i^k, whereas smaller α will emphasize the more distant past. We present the server-side event-based participation algorithm in Alg. 1, whereby $\mathcal{S}_c$ denotes the client selection operator and I_{s}^k denotes the set of clients participating in round k.

Finally, we state the main result of the paper, the FedBack algorithm which utilizes Alg. 1 to reduce the number of participation events necessary to compute a locally optimal solution to (2.1).

Algorithm 2 FedBack

Given: ρ, clients $\mathcal{I} = \{1, \ldots, N\}$, selection operator $\mathcal{S}_c$, accuracy $\{\varepsilon_k\}_{k=1}^T$
Initialize: $\theta_1^0 = \cdots = \theta_N^1 = z^0$, $\lambda_i^0 = 0 \,\forall i \in \mathcal{I}$
for each round $k = 0$ to $T - 1$ **do**
  Server selects $I_{\text{s}}^k = \mathcal{S}_c(\mathcal{I})$ {Compute via Alg. 1}
  // Client-side computation
  for each client $i \in I_{\text{s}}^k$ in parallel **do**
    Download ω^k from server
    $\lambda_i^{k+1} \leftarrow \lambda_i^k + \theta_i^k - \omega^k$
    $\theta_i^{k+1} \approx \text{argmin}_{\theta \in \mathbb{R}^d} f_i(\theta) + \frac{\rho}{2}\left|\theta - \omega^k + \lambda_i^{k+1}\right|^2$ {Local training: initialize θ with ω^k}
      such that $|\nabla_\theta f_i(\theta_i^{k+1}) + \rho(\theta_i^{k+1} - \omega^k + \lambda_i^{k+1})| \leq \varepsilon_k$
    $z_i^{\text{prev}} \leftarrow \lambda_i^{k+1} + \theta_i^{k+1}$
    Upload z_i^{prev} to the server
  end for
  for each client $i \notin I_{\text{s}}^k$ **do**
    $(\theta_i^{k+1}, \lambda_i^{k+1}) \leftarrow (\theta_i^k, \lambda_i^k)$
  end for
  // Server-side computation
  $\omega^{k+1} \leftarrow \frac{1}{N}\sum_{i=1}^N z_i^{\text{prev}}$ {Update global average}
end for

At each round k of Alg. 2, the server selects a subset I_{s}^k of clients to participate via Alg. 1. Each client i in the set I_{s}^k performs the local (2.3) update before uploading z_i^{prev} to the server. The primal updates (2.3) can be performed inexactly, and we only require convergence to a stationary point with accuracy ε_k, where $\{\varepsilon_k\}$ is a positive sequence converging to zero. The server then aggregates each z_i^{prev} to compute ω^k.

Algorithm 1 requires that the server holds z_i^{prev} in memory for each client $i \in \mathcal{I}$. We acknowledge that this is not scalable in a cross-device setting (McMahan et al., 2017). However, introducing dual variables already limits Alg. 2 to the cross-silo setting (Li et al., 2022) since each client has to maintain memory of λ_i^k, even when $i \notin I_{\mathrm{s}}^k$. Therefore, Alg. 2 still remains scalable in the cross-silo setting (≈ 100 clients), provided that the models are not too large (≈ 1 billion parameters). Future work may investigate compression strategies, such as the applications of the Johnson-Lindenstrauss Lemma (Dasgupta and Gupta, 2003), which could maintain scalability when dealing with even larger models.

We recall that $|\omega^k - z_i^{\text{prev}}| = |x_i^k + \lambda_i^k - \omega^k|$, which implies that the server selects clients with the largest proximal value in the primal update (2.3). Moreover, $|\lambda_i^{\text{prev}} + \theta_i^{\text{prev}} - \omega^k| = |\lambda_i^{k+1}|$ and $\lambda_i^{k+1} = \sum_{j=1}^{k}(\theta_i^j - \omega^j)$, implying that Alg. 1 naturally combats the client drift problem (McMahan et al., 2017; Li et al., 2020c) by selecting clients that have built up a history of deviating too far from the server parameters.

4. Theoretical Analysis

This section presents theoretical guarantees for Alg. 2 under mild assumptions on the local objectives f_i. We analyze the stability of participation dynamics and their limiting behavior.

4.1. Global Stability of Participation Dynamics

We first establish a key property of the participation dynamics, which ensures that the thresholds δ_i^k remain well-behaved over time.

Lemma 1 (Bounded Threshold) *Let the gradients in local training rounds (2.3) be bounded. Then, there exists a threshold value δ_+, such that the identifier function S_i^k in (3.1) satisfies*

$$S_i^k(\delta) = 0, \quad \forall \delta \geq \delta_+ > 0.$$

As a consequence, the following bound for the threshold at any time $k \geq 0$ holds,

$$\min\left\{\delta_i^0 - \frac{K}{\alpha}, -K\left(\frac{1+\alpha}{\alpha}\right)\right\} \leq \delta_i^k \leq \max\left\{\delta_+ + K\left(\frac{1+\alpha}{\alpha}\right), \delta_i^0 + \frac{K}{\alpha}\right\}.$$

Proof See (Cummins et al., 2024) for the proof. ∎

We now show that the participation rates converge to the target value $\bar{L}_i$ under the closed-loop dynamics.

Theorem 2 (Global Stability) *Let the sequence $\{\delta_i^k\}_{k=0}^{T-1}$ be generated by the closed loop dynamics (3.3) and (3.4). Then, the time-averaged participation rate $\frac{1}{T}\sum_{k=0}^{T-1} S_i^k(\delta_i^k)$ converges to the target value $\bar{L}_i$ at a rate of $\mathcal{O}\left(\frac{1}{T}\right)$, for any $\bar{L}_i \in [0,1]$ and for $K > 0$.*

Proof By rearranging (3.4), we express the dynamics of L_i^k as $L_i^{k+1} - L_i^k = -\alpha L_i^k + \alpha S_i^k(\delta_i^k)$, which concludes

$$L_i^T - L_i^0 = -\alpha \sum_{k=0}^{T-1} L_i^k + \alpha \sum_{k=0}^{T-1} S_i^k(\delta_i^k). \tag{4.1}$$

In a similar fashion, we rearrange (3.3)

$$\sum_{k=0}^{T-1} L_i^k = \frac{\delta_i^T - \delta_i^0}{K} + T\bar{L}_i. \tag{4.2}$$

Combining (4.1) and (4.2), we have

$$\frac{1}{T}\sum_{k=0}^{T-1} S_i^k(\delta_i^k) = \bar{L}_i + \frac{\delta_i^T - \delta_i^0}{KT} + \frac{L_i^T - L_i^0}{\alpha T}.$$

Using boundedness of δ_i^T in Lemma 1 and the fact $|L_i^T - L_i^0| \leq 1$, we establish upper and lower bounds as

$$\frac{c_1}{T} \leq \frac{1}{T}\sum_{k=0}^{T-1} S_i^k(\delta_i^k) - \bar{L}_i \leq \frac{c_2}{T},$$

with constants $c_1 = \min\left\{-\frac{2}{\alpha}, -\frac{\delta_i^0}{K} - \frac{(2+\alpha)}{\alpha}\right\}$ and $c_2 = \max\left\{\frac{\delta_+ - \delta_i^0}{K} + \frac{(2+\alpha)}{\alpha}, \frac{(2+\alpha)}{\alpha}\right\}$, which yields the desired result. ■

Remark 3 *Although we explicitly devote our attention to ADMM, we note that global stability is independent of the deployed decomposition algorithm. The distance metric utilized in* (3.1) *can be designed freely as long as client gradients are bounded in local training rounds.*

4.2. Global Convergence

To analyze global convergence, we define the global Lagrangian $\mathcal{L}$ and corresponding local Lagrangians $\mathcal{L}_i$ of (2.2) as

$$\mathcal{L}(\omega, \Theta, \Lambda) := F(\Theta) + \sum_{i=1}^{N} \left(\lambda_i^\intercal(\theta_i - \omega) + \frac{\rho}{2}|\theta_i - \omega|^2\right) = \sum_{i=1}^{N} \mathcal{L}_i(\omega, \theta_i, \lambda_i), \tag{4.3}$$

where $\Theta := (\theta_1, \ldots, \theta_N)$, $\Lambda := (\lambda_1, \ldots, \lambda_N)$ and $F(\Theta) := \sum_{i=1}^N f_i(\theta_i)$. By Lemma 4 (below), we show that none of the clients will stop participating in the optimization. This result guarantees global convergence to a stationary point of (4.3) by applying the result of (Zhou and Li, 2023).

Lemma 4 (Communication Guarantee) *Let $K > 0$ and $\bar{L}_i > 0$, then* $\limsup_{k\to\infty} \; S_i^k(\delta_i^k) = 1$.

Proof See (Cummins et al., 2024) for the proof. ■

Lemma 4 shows that FedBack satisfies one of the primary requirements for global convergence. We now state the assumptions on (4.3).

Assumption 1 *The objective function $F(\Theta)$ is coercive, i.e., $F(\Theta) \to \infty$ as $|\Theta| \to \infty$.*

Assumption 2 *The parameter* $\rho \in \mathbb{R}^+$ *satisfies* $\rho \geq \max\left\{\frac{3n_1 r_1}{n}, \ldots, \frac{3n_N r_N}{n}\right\}$*, where* n *is the number of data points in the global dataset and* n_i *is the number of data points in the* i^{th} *local dataset.*

Theorem 5 (Global Convergence) *Let Assumptions 1 and 2 hold and let* $K > 0$ *and* $\bar{L}_i \in (0, 1]$*. The following statements hold*

1. *The sequence* $\{(\omega^k, \Theta^k, \Lambda^k)\}$ *is bounded.*
2. *The sequences* $\{\mathcal{L}(\omega^k, \Theta^k, \Lambda^k)\}$*,* $\{F(\Theta^k)\}$ *and* $\{f(\omega^k)\}$ *all converge to the same value:*

$$\lim_{k\to\infty} \mathcal{L}(\omega^k, \Theta^k, \Lambda^k) = \lim_{k\to\infty} F(\Theta^k) = \lim_{k\to\infty} \sum_{i=1}^{N} f_i(\omega^k).$$

3. $\nabla F(\Theta^k)$ *and* $\nabla f_i(\omega^k)$ *eventually vanish, i.e.,* $\lim_{k\to\infty} \nabla F(\Theta^k) = \lim_{k\to\infty} \sum_{i=1}^{N} \nabla f_i(\omega^k) = 0$*.*
4. *Any accumulating point* $(\omega^\infty, \Theta^\infty, \Lambda^\infty)$ *of* $\{(\omega^k, \Theta^k, \Lambda^k)\}$ *is a stationary point of* (4.3)*.*

Proof As a result of Lemma 4, the result of (Zhou and Li, 2023) holds. This yields the desired conclusion. ■

We refer the reader to Er et al. (2024) for results on the convergence rate of ADMM with event-based communication. Due to the upper bound on δ_i^k, the results from Er et al. (2024) apply directly.

5. Numerical Evaluation

We evaluate FedBack against FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020c), and FedADMM (Zhou and Li, 2023) on MNIST and CIFAR-10 classification tasks, using non-i.i.d. data distributions across 100 clients. FedADMM is a natural comparison for FedBack and a version of FedAvg/FedProx may be recovered from FedADMM by enforcing $\rho = 0$ and $\lambda_i^{k+1} = 0$ respectively and performing a non-weighted aggregation on the server-side.

Metrics Due to the nature of Alg. 1, there will be a varying number of client's participating in each round of Alg. 2. We therefore consider the total number of participation events necessary to achieve a desired accuracy as our first metric. When evaluating the performance of FedADMM, FedAvg and FedProx, this metric is equivalent to the number of rounds required to reach a specified accuracy. Furthermore, we track the validation accuracy per round for each algorithm for a select $\bar{L} = \bar{L}_i$ ($\forall i \in \mathcal{I}$) to illustrate the superior convergence of FedBack and the advantage of using a deterministic client selection scheme. Finally, we support Thm. 2 by analyzing the realized participation rate among clients for a given $\bar{L}$.

MNIST Client data is distributed such that each client has an equal number of data points but is restricted to two unique digits. The classifier is a fully connected neural network with a single hidden layer of 200 neurons and ReLU activation. The expected accuracy of the classifier is 93% when training in a centralized fashion. Local updates are performed using SGD with learning rate 0.01, momentum factor 0.9 and batch size 42 for two epochs. We choose $K = 2$ and $\alpha = 0.9$.

CIFAR-10 Client data is distributed using a Dirichlet distribution (Li et al., 2021; Yurochkin et al., 2019; Wang et al., 2020a) with concentration parameter $\beta = 0.5$. The classifier is a CNN with three convolutional layers, three fully connected layers and ReLU activation. When training in the centralized case, the expected test accuracy is 80%. Local updates are performed using SGD with learning rate 0.01, momentum factor 0.9 and batch size 20 for four epochs. We again choose $\alpha = 0.9$ but increase $K = 5$, since the CIFAR-10 classifier contains significantly more parameters.

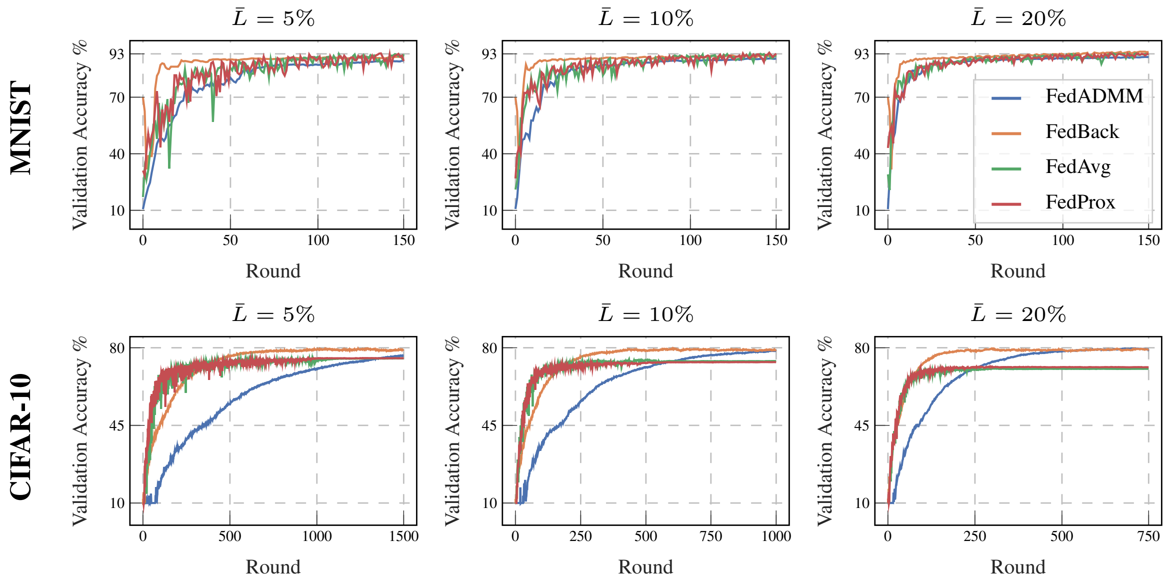

Figure 1: Validation accuracy of server parameters ω^k per round k for MNIST (top row) and CIFAR-10 (bottom row) classifiers for each FL algorithm with communication load references $\bar{L} = \{0.05, 0.1, 0.2\}$. For FedADMM, FedAvg and FedProx, we randomly sample an $\bar{L}$ proportion of clients, uniformly at random, for participation at each round.

	Algorithm	$\bar{L}=5\%$	$\bar{L}=10\%$	$\bar{L}=15\%$	$\bar{L}=20\%$	$\bar{L}=40\%$	$\bar{L}=60\%$
MNIST	**FedBack**	412	**430**	**493**	**538**	**677**	1274
	FedADMM	> 750	1280	1215	1340	1160	**1080**
	FedAvg	370	550	675	720	1280	2160
	FedProx	**335**	620	780	880	1240	2400
CIFAR10	**FedBack**	**3936**	**4167**	**4328**	**4402**	**4659**	**5064**
	FedADMM	> 7500	8960	9720	9260	9000	9840
	FedAvg	N/A	N/A	N/A	N/A	N/A	N/A
	FedProx	N/A	N/A	N/A	N/A	N/A	N/A

Table 1: Total number of participation events necessary for each algorithm to reach the target accuracy of 90% for the MNIST Classifier and 78% for the CIFAR-10 Classifier. The "N/A" entries represent cases where the algorithm could not achieve the desired accuracy within the given number of rounds.

All experiments are implemented in PyTorch with Nvidia A100 GPUs. Fig. 1 demonstrates a notable trade-off regarding the noise in the server parameters, particularly for FedProx and FedAvg. This noise can be attributed to the random sampling of clients when $\bar{L}$ is low, creating high variance in the server parameters between rounds. This is particularly unfavorable in practical FL scenarios where the server does not have access to a validation dataset. To ensure a low level of variance in ω^k between rounds, one would have to enforce a high $\bar{L}$ or run the FL procedure for longer than required to ensure convergence. In contrast, FedBack uses an adaptive participation mechanism, which reduces the variance in global model parameters, leading to more stable performance over time when dealing with a low $\bar{L}$.

We see from Tab. 1 that FedBack outperforms all algorithms in the majority of test cases regarding participation efficiency across both MNIST and CIFAR-10 datasets. In the case of MNIST, FedProx is the most efficient at $\bar{L} = 5\%$ but FedBack remains close with only 412 participation events and significantly less variance, as illustrated in Fig. 1. CIFAR-10 classifiers trained with FedAvg and FedProx fail to reach the desired accuracy. This is likely due to being a heuristic modification of the distributed gradient method (McMahan et al., 2017; Ryu and Yin, 2022), which is used to solve (2.2).

	$\bar{L}=5\%$	$\bar{L}=10\%$	$\bar{L}=15\%$	$\bar{L}=20\%$	$\bar{L}=40\%$	$\bar{L}=60\%$
CIFAR-10	5.54%	10.67%	15.81%	20.62%	40.71%	60.49%
MNIST	7.46%	12.07%	17.41%	22.58%	43.03%	64.93%

Table 2: Average participation rate of clients in the network for a given $\bar{L}$. We compute the percentage of how many rounds each client participates in across the entire FL procedure and compute the average.

In addition, our feedback policy proved particularly effective in stabilizing the communication load throughout the learning process. Tab. 2 demonstrates the communication load stabilizing within 0.9% precision over longer training rounds in the CIFAR-10 experiments. This is a significant improvement over the MNIST experiments, where the dynamics did not have sufficient time to stabilize, highlighting FedBack's potential for long-term, stable communication management in federated learning settings. Although our feedback policy demonstrated less accurate tracking of $\bar{L}$, Tab. 1 still demonstrates superiority in reaching the desired accuracy in much fewer participation events for $\bar{L}=\{0.10, 0.15, 0.20, 0.40\}$.

6. Conclusion

This paper introduced FedBack, a modification of the cross-silo FedADMM algorithm that applies a control theoretic methodology to client participation. We presented a theoretical analysis that justified our approach while also maintaining previously established global convergence properties. FedBack demonstrated a great advantage over FedAvg, FedProx and FedADMM during numerical evaluation. In most cases, FedBack was able to achieve the desired accuracy in almost half the number of participation events, while also demonstrating a drastic reduction in the variance of server parameters for a low $\bar{L}$. Moreover, our feedback control law was evaluated by accurately tracking $\bar{L}$ to an exceptional degree in the CIFAR-10 experiments and an acceptable degree in the MNIST experiments. Both FedBack and FedADMM were able to achieve the same validation accuracy as a model trained in a fully centralized fashion, with FedBack demonstrating a much faster convergence rate in the CIFAR-10 experiments.

In conclusion, by dynamically adjusting the communication threshold δ_i for each client, FedBack achieves better control over the communication load while maintaining stability in model training. This is particularly beneficial in practical FL scenarios where the communication resources are limited and/or changing over time. Further work will investigate more advanced feedback strategies and possibly integrate the feedback mechanism to other FL algorithms which utilize proximal terms in their objective, such as FedProx.

Acknowledgments

The authors thank Harun Siljak and Anthony Quinn for engaging in helpful discussions, and particularly thank Harun for enabling the collaboration. Guner Dilsad Er and Michael Muehlebach thank the German Research Foundation and the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for their support.

References

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *Proceedings of the International Conference on Learning Representations*, pages 1–36, 2021.

Karl J. Aström. *Event Based Control*. Springer Berlin Heidelberg, 2008.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Michael Cummins, Guner Dilsad Er, and Michael Muehlebach. Controlling participation in federated learning with feedback. *arXiv:2411.19242*, 2024.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22, 2003.

Florian Dörfler, Zhiyu He, Giuseppe Belgioioso, Saverio Bolognani, John Lygeros, and Michael Muehlebach. Towards a systems theory of algorithms. *IEEE Control Systems Letters*, 2024.

Guner Dilsad Er, Sebastian Trimpe, and Michael Muehlebach. Distributed event-based learning via ADMM. *arxiv:2405.10618*, 2024.

Yonghai Gong, Yichuan Li, and Nikolaos M. Freris. FedADMM: A robust federated deep learning framework with adaptivity to system heterogeneity. *Proceedings of the IEEE International Conference on Data Engineering*, pages 2575–2587, 2022.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. *Proceedings of International Conference on Machine Learning*, 119:5132–5143, 2020.

Pavel Kolev, Georg Martius, and Michael Muehlebach. Online learning under adversarial nonlinear constraints. *Advances in Neural Information Processing Systems*, 2023.

Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

Meng Li, Mehrdad Sanjabi, and Martin Jaggi. Federated optimization in heterogeneous networks. *Proceedings of the Conference on Machine Learning and Systems*, 2:429–450, 2020a.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722, 2021.

Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *Proceedings of the IEEE International Conference on Data Engineering*, pages 965–978, 2022.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3), 2020b.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of the Conference on Machine Learning and Systems*, 2:429–450, 2020c.

H Brendan McMahan, Eider Moore, Daniel Ramage, and Seth Hampson. Communication-efficient learning of deep networks from decentralized data. *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2017.

Marek Miskowicz. Send-on-delta concept: An event-based data reporting strategy. *Sensors*, 6(1): 49–63, 2006.

Michael Muehlebach and Michael I Jordan. Continuous-time lower bounds for gradient-based algorithms. *Proceedings of the International Conference on Machine Learning*, 2020.

Michael Muehlebach and Sebastian Trimpe. Distributed event-based state estimation for networked systems: An LMI approach. *IEEE Transactions on Automatic Control*, 63(1):269–276, 2017.

Robert Nishihara, Laurent Lessard, Benjamin Recht, Andrew Packard, and Michael I. Jordan. A general analysis of the convergence of ADMM. *Proceedings of the International Conference on Machine Learning*, 37:343–352, 2015.

James Blake Rawlings, David Q Mayne, Moritz Diehl, et al. *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing, 2017.

Ernest K. Ryu and Wotao Yin. *Large-Scale Convex Optimization: Algorithms and Analyses via Monotone Operators*. Cambridge University Press, 2022.

Yong Shi, Yuanying Zhang, Peng Zhang, Yang Xiao, and Lingfeng Niu. Federated learning with ℓ-1 regularization. *Pattern Recognition Letters*, 172:15–21, 2023.

Guanchun Tong and Michael Muehlebach. A dynamical systems perspective on discrete optimization. *Learning for Dynamics and Control Conference*, 211:1373–1386, 2023.

Quoc Tran Dinh, Nhan H Pham, Dzung Phan, and Lam Nguyen. FedDR–randomized Douglas-Rachford splitting algorithms for nonconvex federated composite optimization. *Advances in Neural Information Processing Systems*, 34:30326–30338, 2021.

Han Wang, Siddartha Marella, and James Anderson. FedADMM: A federated primal-dual algorithm allowing partial participation. *Proceedings of the Conference on Decision and Control*, pages 287–294, 2022.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *Proceedings of the International Conference on Learning Representations*, pages 1–16, 2020a.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in Neural Information Processing Systems*, 33:7611–7623, 2020b.

Andre Wibisono, Ashia C. Wilson, and Michael I Jordan. A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 113(47): E7351–E7358, 2016.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. *Proceedings of the International Conference on Machine Learning*, pages 7252–7261, 2019.

Shahryar Zehtabi, Seyyedali Hosseinalipour, and Christopher G. Brinton. Decentralized event-triggered federated learning with heterogeneous communication thresholds. *Proceedings of the Conference on Decision and Control*, pages 4680–4687, 2022.

Xinwei Zhang, Mingyi Hong, Sairaj Dhople, Wotao Yin, and Yang Liu. FedPD: A federated learning framework with adaptivity to non-iid data. *IEEE Transactions on Signal Processing*, 69: 6055–6070, 2021.

Shenglong Zhou and Geoffrey Ye Li. Federated learning via inexact ADMM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9699–9708, 2023.