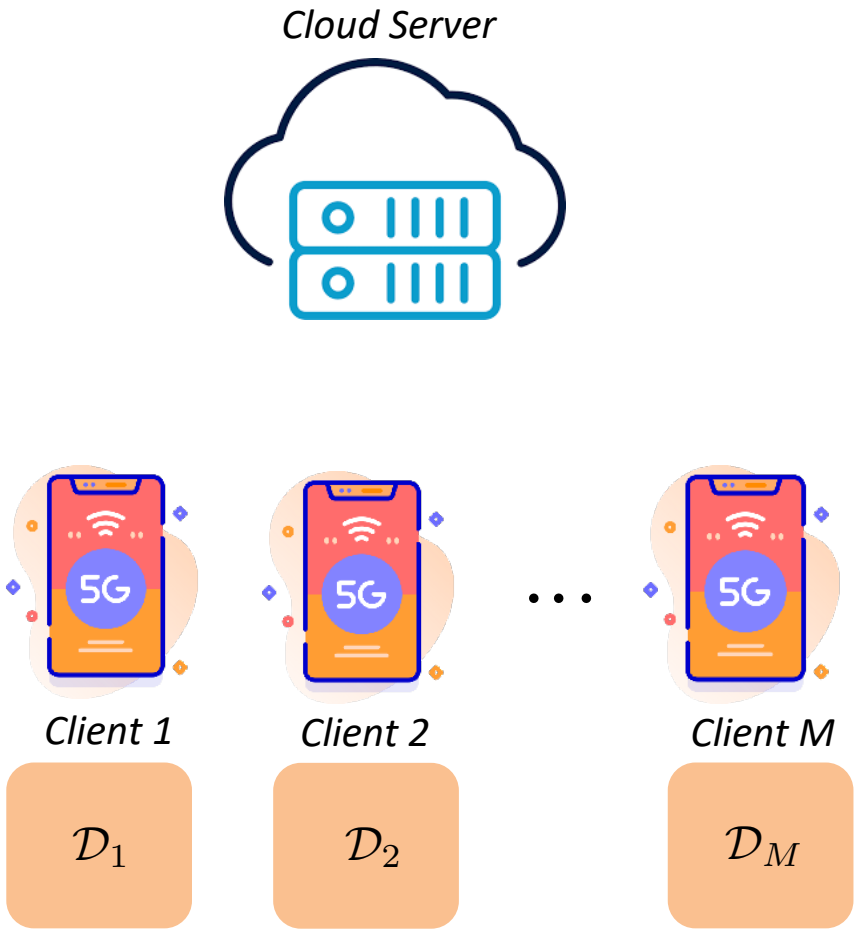
FedExP: Speeding Up Federated Averaging Via Extrapolation

Divyansh Jhunjhunwala[1], Shiqiang Wang[2], Gauri Joshi[1]

[1]*Carnegie Mellon University*, [2]*IBM Research*

Problem Formulation



Federated Learning (FL) Objective:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \frac{1}{M}\sum_{i=1}^{M} F_i(\mathbf{w})$$

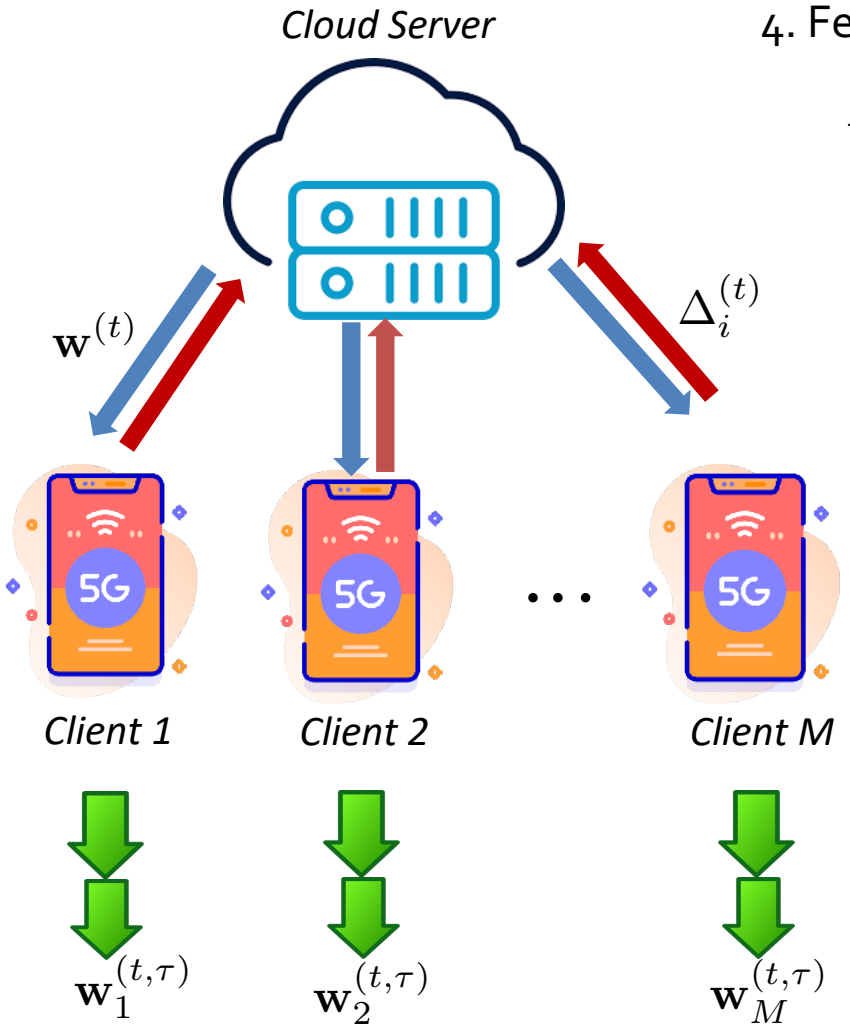
M is the number of clients, $F_i(\mathbf{w}) = \frac{1}{|\mathcal{D}_i|}\sum_{\mathbf{x}\in\mathcal{D}_i} \ell(\mathbf{w},\mathbf{x})$ is the local client objective.

- FedAvg [1] remains the most popular FL algorithm due to its simplicity and ease of implementation.

[1] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *AISTATS*, 2017.

FedAvg Overview

1. Server sends current global model $\mathbf{w}^{(t)}$ to clients.



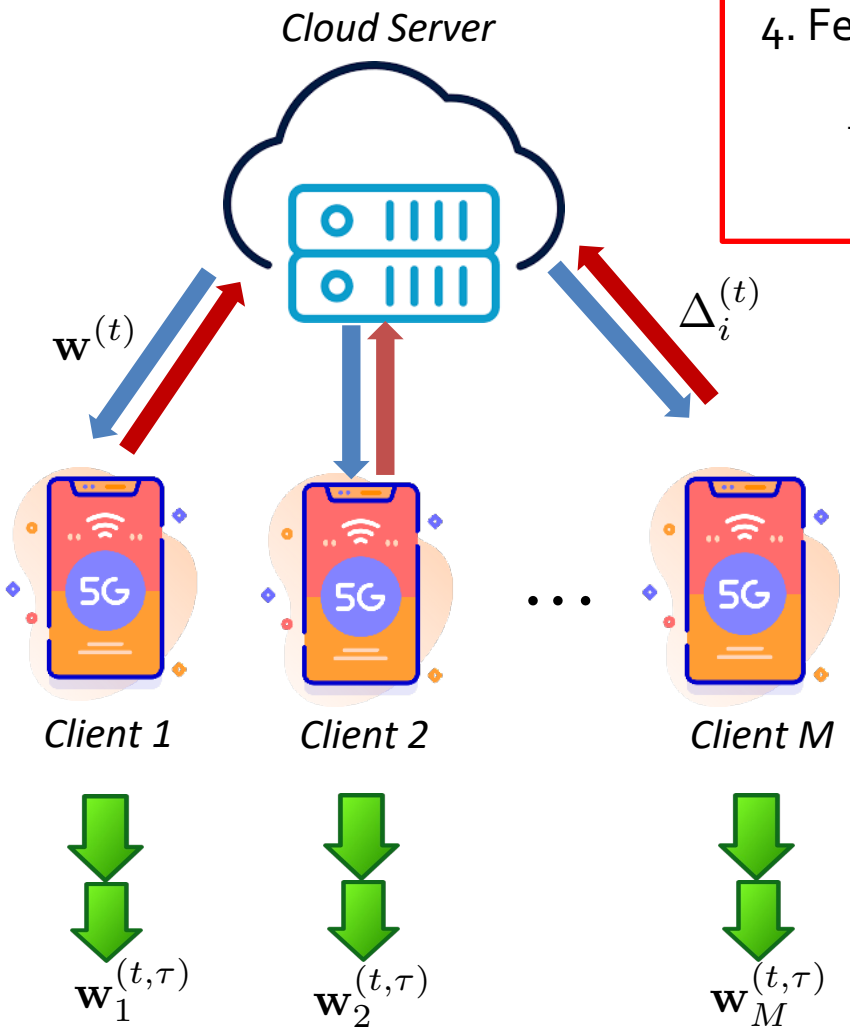
4. FedAvg global update:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \frac{1}{M}\sum_{i=1}^{M}\Delta_i^{(t)}$$

3. Clients send back updates $\Delta_i^{(t)} = \mathbf{w}_i^{(t,\tau)} - \mathbf{w}_i^{(t,\tau)}$

2. Clients perform τ steps of local training to get local model $\mathbf{w}_i^{(t,\tau)}$

Our Focus



4. FedAvg global update:

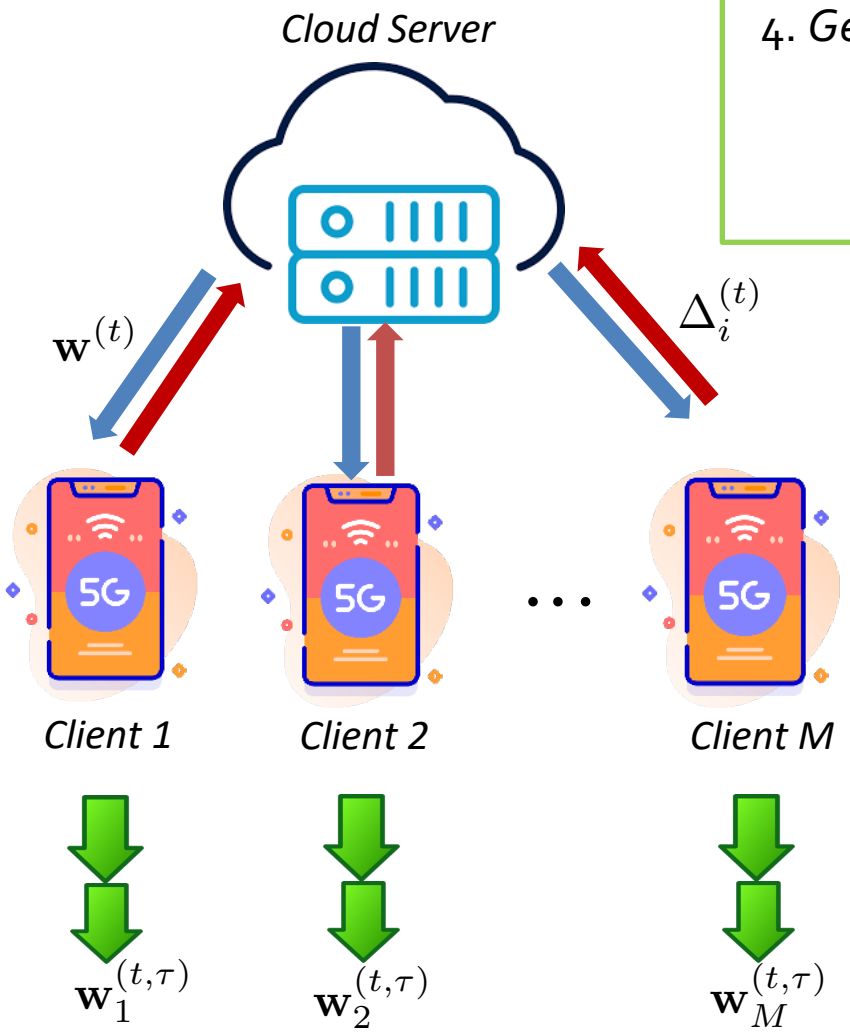
$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \frac{1}{M}\sum_{i=1}^{M}\Delta_i^{(t)}$$

Q. Can we do better than vanilla averaging?

Key Idea: *Treat client updates $\Delta_i^{(t)}$ as "pseudo-gradients" → server aggregation is a **gradient descent** step → multiply with a server step size η_g [2] .*

[2] Reddi, S., et al. "Adaptive federated optimization." *ICLR*, 2021

Our Focus



4. *Generalized* FedAvg global update:

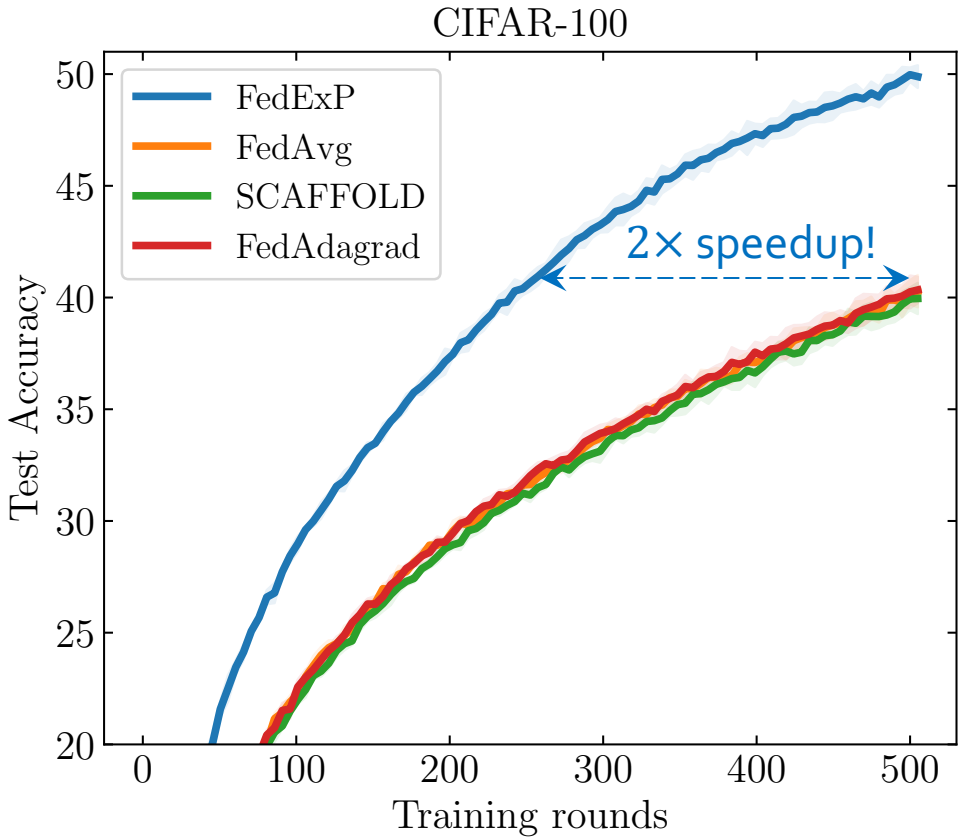
$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_g \frac{1}{M} \sum_{i=1}^{M} \Delta_i^{(t)}$$

Q. Can we do better than vanilla averaging?

Key Idea: *Treat client updates* $\Delta_i^{(t)}$ *as "pseudo-gradients" → server aggregation is a* ***gradient descent*** *step → multiply with a server step size* η_g.

Q. How to set server step size for faster convergence?

Proposed Algorithm Sketch



Experimental results on a federated split of the CIFAR-100 dataset.

Key Idea: ***Adapt*** *the server step size at every round based on heterogeneity of client updates in that round.*

FedExP server step size $\eta_g^{(t)} \propto \dfrac{\sum_{i=1}^{M} \|\Delta_i^{(t)}\|^2}{M\|\bar{\Delta}^{(t)}\|^2}$

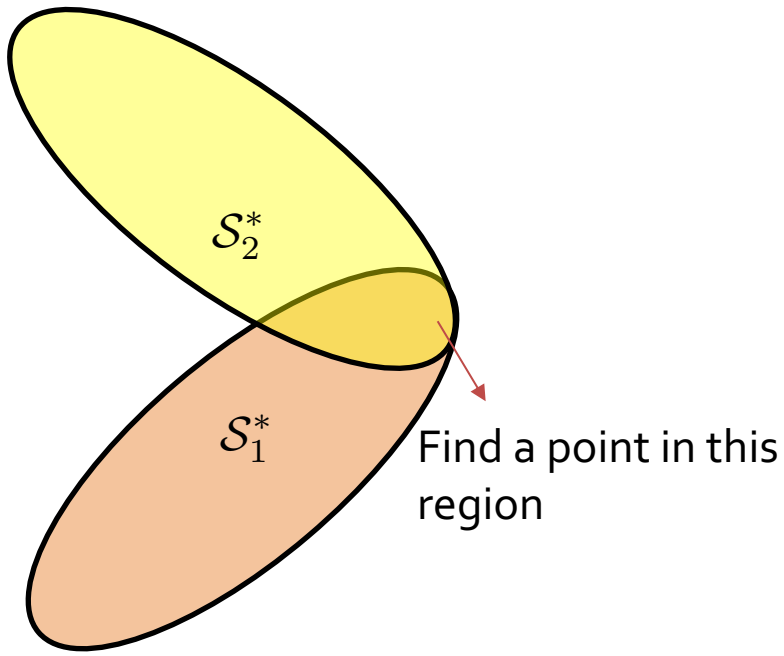
where

$$\bar{\Delta}^{(t)} = \left(\sum_{i=1}^{M} \Delta_i^{(t)}\right)/M$$

Motivation: To understand the motivation behind FedExP, we first consider the **overparameterized regime** and highlight:

- Connection between FedAvg and Projection on Convex Sets (POCS) algorithm.
- Check out paper for intuition for non-convex case and convergence guarantees!

Overparameterized Convex Regime in FL



Client solution sets intersect

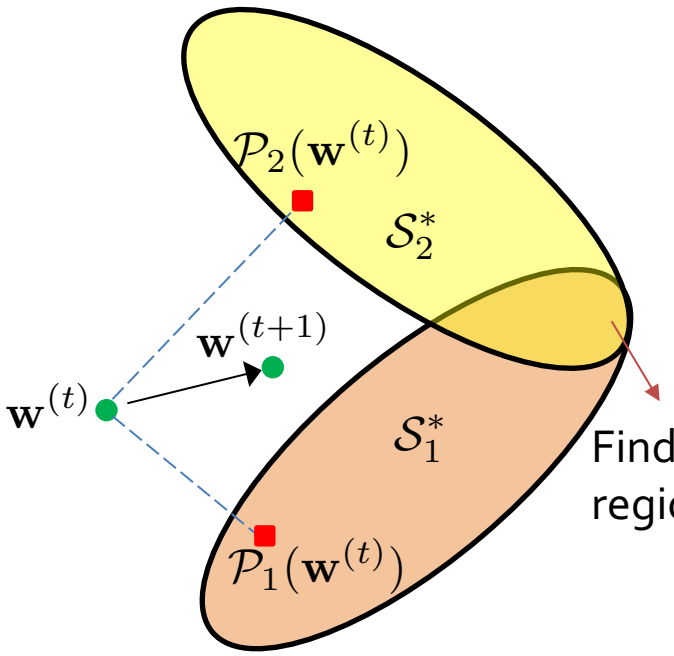
Overparameterized Convex Regime:

Assume $F_i(\mathbf{w})$ is convex for all $i \in [M]$ and have a common minimizer $\mathbf{w}^*$ (overparameterization).

- $\mathcal{S}_i^* = \left\{ \underset{\mathbf{w}}{\operatorname{argmin}}\ F_i(\mathbf{w}) \right\}$ is a convex set.
- Furthermore, intersection of $\mathcal{S}_i^*$ is non-empty since $\mathbf{w}^* \in \mathcal{S}^*\ \forall i \in [M]$.
- FL objective can alternately be thought of as trying to **find a point in the intersection of convex sets.**

$$\min_{\mathbf{w}} F(\mathbf{w}) \iff \text{Find } \mathbf{w} \text{ s.t. } \mathbf{w} \in \mathcal{S}_1^* \cap \mathcal{S}_2^* \cdots \cap \mathcal{S}_M^*$$

Projection on Convex Sets (POCS) Algorithm



Find a point in this region

Projection on Convex Sets Algorithm [3]:

Goal: Find a point that lies in the intersection of convex sets $\mathcal{S}_1^*, \mathcal{S}_2^*, \ldots \mathcal{S}_M^*$

Algorithm:

1. Compute $\mathcal{P}_i(\mathbf{w}^{(t)}) = \underset{\mathbf{w} \in \mathcal{S}_i^*}{\operatorname{argmin}} \|\mathbf{w} - \mathbf{w}^{(t)}\|^2$, projection of $\mathbf{w}^{(t)}$ on the set $\mathcal{S}_i^*$

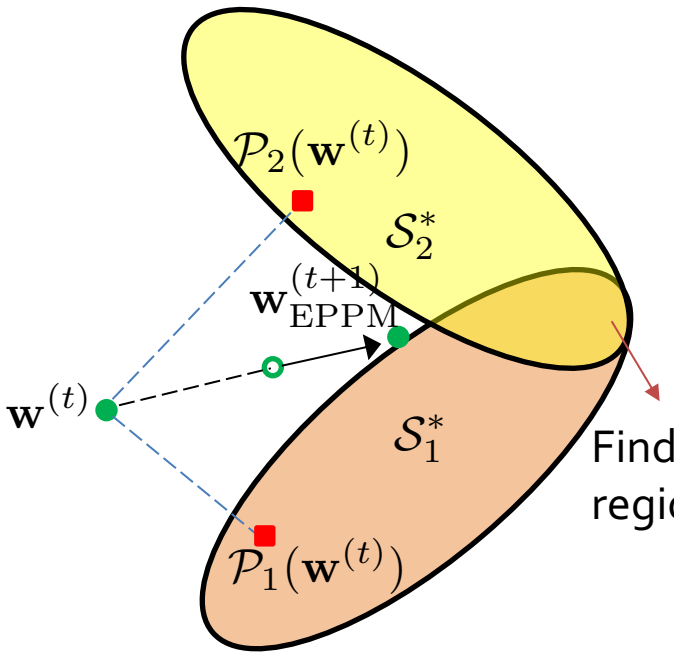
2. Update

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \lambda \frac{1}{M} \sum_{i=1}^{M} \left(\mathbf{w}^{(t)} - \mathcal{P}_i(\mathbf{w}^{(t)}) \right)$$

Here λ is known as the relaxation co-efficient.

[3] Combettes, Patrick L. "Convex set theoretic image recovery by extrapolated iterations of parallel subgradient projections." *IEEE Transactions on Image Processing* 6, no. 4 (1997)

Extrapolated Parallel Projection Method (EPPM)



Find a point in this region

Extrapolated Parallel Projection Method [4]:

1. Projection step is same as POCS.
2. *Adapt* λ at every iteration.

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \lambda^{(t)} \frac{1}{M} \sum_{i=1}^{M} \left(\mathbf{w}^{(t)} - \mathcal{P}_i(\mathbf{w}^{(t)}) \right)$$

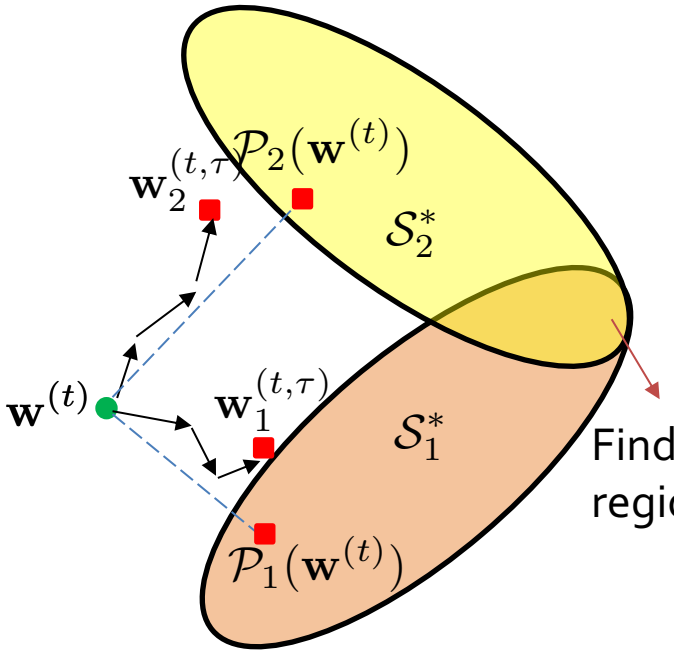
where $\lambda^{(t)} = \dfrac{\sum_{i=1}^{M} \|\mathbf{w}^{(t)} - \mathcal{P}_i(\mathbf{w}^{(t)})\|^2}{M \| \frac{1}{M} \sum_{i=1}^{M} (\mathbf{w}^{(t)} - \mathcal{P}_i(\mathbf{w}^{(t)})) \|^2}$

Known as "extrapolation" since we have $\lambda^{(t)} \geq 1 \ \forall t$ by Jensen's inequality.

Key property: $\|\mathbf{w}_{\mathrm{EPPM}}^{(t+1)} - \mathbf{w}^*\|^2 \leq \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2$

[4] Pierra, G. (1984). "Decomposition through formalization in a product space." *Mathematical Programming*, *28*(1), 96-115.

Extending extrapolation to FL



Find a point in this region

FedAvg with "approximate" projections

POCS update:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \lambda \frac{1}{M} \sum_{i=1}^{M} \left(\mathbf{w}^{(t)} - \mathcal{P}_i(\mathbf{w}^{(t)}) \right)$$

Generalized FedAvg update:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_g \frac{1}{M} \sum_{i=1}^{M} \left(\mathbf{w}^{(t)} - \mathbf{w}_i^{(t,\tau)} \right)$$

Local models $\mathbf{w}_i^{(t,\tau)}$ can be thought of as *approximate projections* $\approx \mathcal{P}_i(\mathbf{w}^{(t)})$.

→ η_g plays same role as λ . *Can apply extrapolation idea to tune* η_g .

Proposed Algorithm: FedExP

Propose *Federated Extrapolated Averaging* or FedExP for FL settings, where at every round the server step size is set as follows:

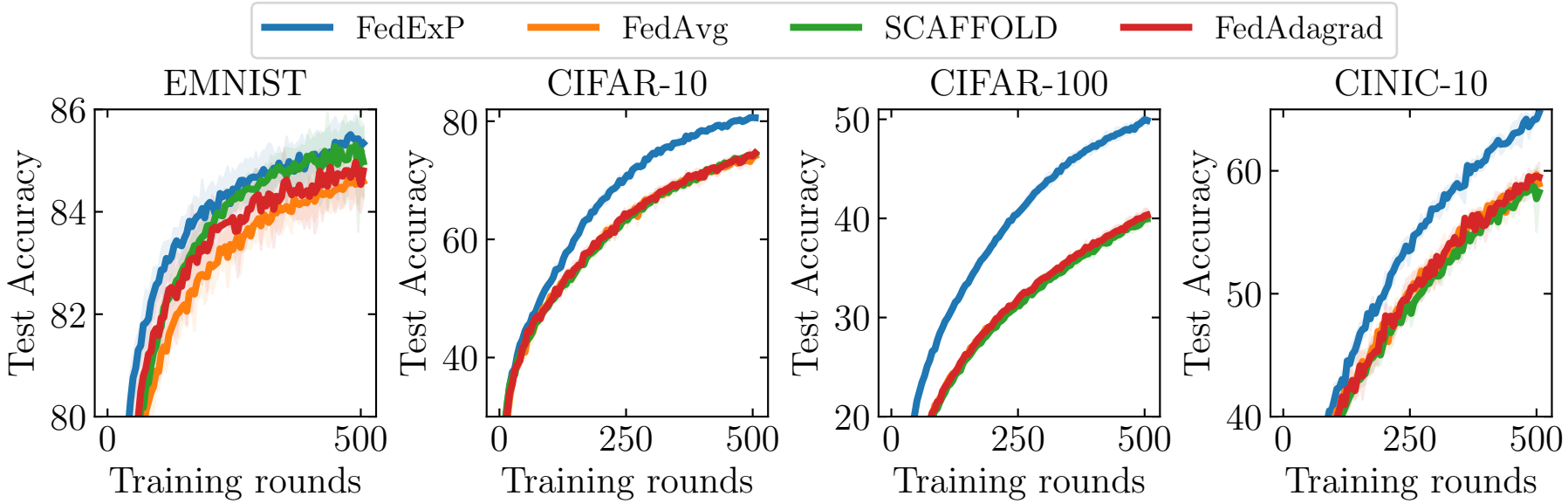
$$\eta_g^{(t)} = \max\left\{1, \frac{\sum_{i=1}^{M}\left\|\Delta_i^{(t)}\right\|^2}{2M\left(\left\|\frac{1}{M}\sum_{i=1}^{M}\Delta_i^{(t)}\right\|^2 + \epsilon\right)}\right\}$$

where $\Delta_i^{(t)} = \mathbf{w}^{(t)} - \mathbf{w}_i^{(t,\tau)}$.

Ease of implementation

- Virtually no additional communication at clients (just one scalar), no extra computation (just norm computation), no extra storage at clients or server.
- Can be integrated with secure aggregation and partial client participation. Thus, applicable for both cross-device and cross-silo FL.

Results on Federated Neural Network Training



Dataset	Target Acc.	FedExP	FedAvg	SCAFFOLD	FedAdagrad
EMNIST	84%	**186**	328 (1.76 ×)	232 (1.24 ×)	277 (1.48 ×)
CIFAR-10	72%	**267**	434 (1.62 ×)	429 (1.61 ×)	419 (1.56 ×)
CIFAR-100	40%	**242**	500 (2.06 ×)	> 500 (2.06 ×)	494 (2.04 ×)
CINIC-10	58%	**318**	450 (1.42 ×)	470 (1.48 ×)	444 (444 ×)

Key Takeaways

- Propose to adapt server step size in each round motivated by connection between FedAvg and POCS.

- Motivation for extrapolation as reducing *distance to optimum* by viewing local models as *approximate projections*.

- Proposed FedExP consistently outperforms baselines while adding no virtually *no additional computation, communication, storage at clients or server*. Also provide theoretical guarantees for convergence of FedExP for both convex and non-convex problems.