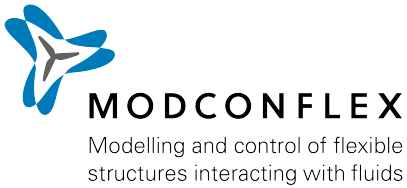
Efficient Federated Learning via ADMM

Ziqi Wang
joint work with Yongcun Song and Enrique Zuazua

Chair for Dynamics, Control, Machine Learning and Numerics
Alexander von Humboldt-Professorship
Friedrich-Alexander-Universität Erlangen-Nürnberg

August 26, 2024

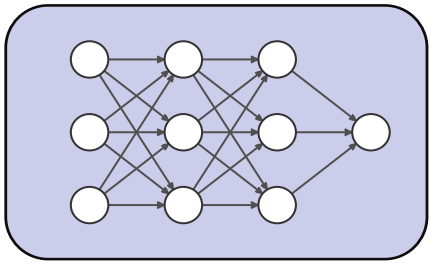


Funded by
the European Union

Outline

Centralized learning

MNIST[1]



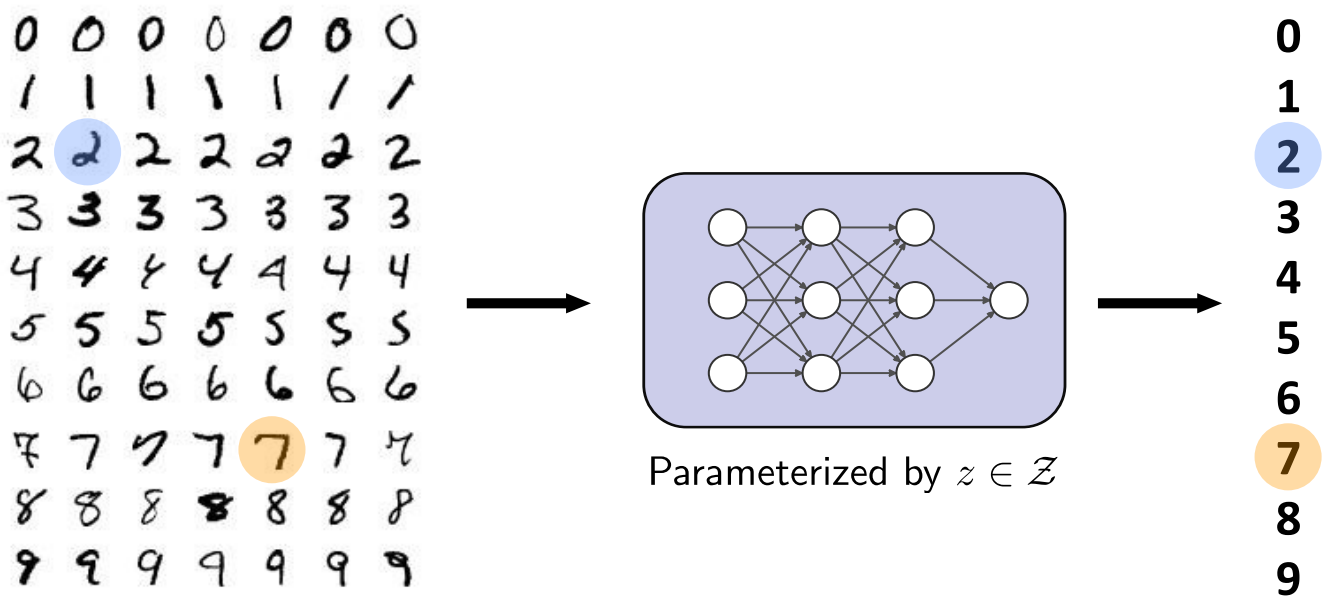
Parameterized by $z \in \mathcal{Z}$

0
1
2
3
4
5
6
7
8
9

$x \in \mathcal{X}$ $f_z : \mathcal{X} \to \mathcal{Y}$ $y \in \mathcal{Y}$

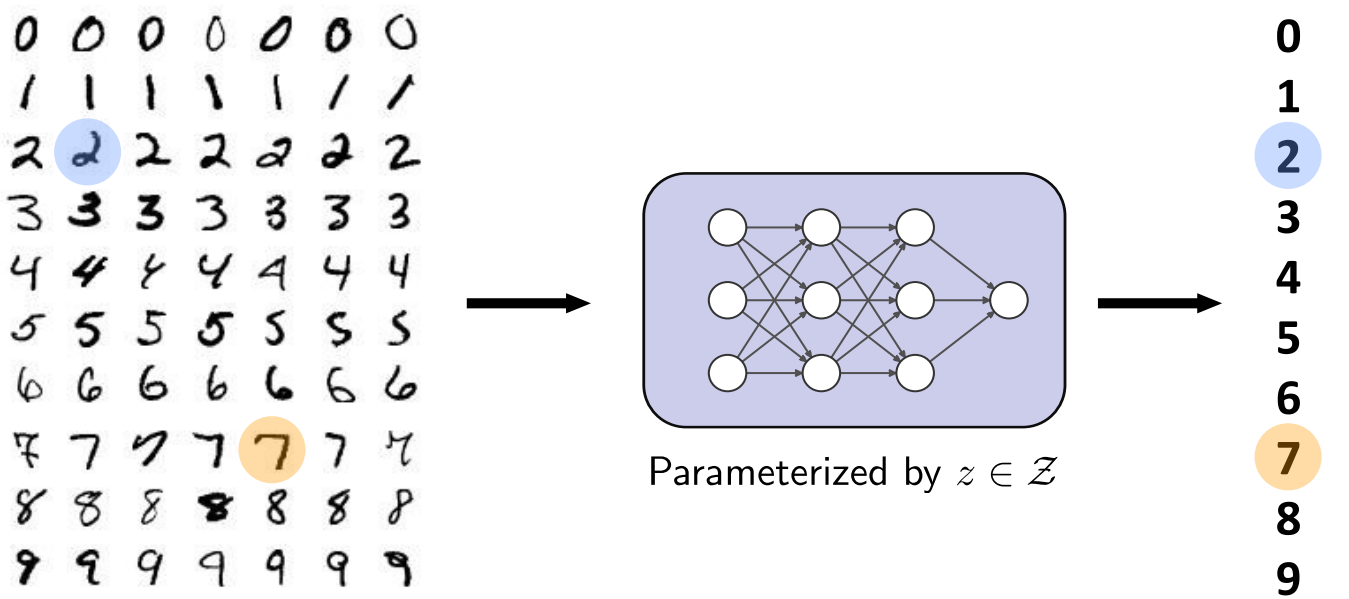
[1] http://yann.lecun.com/exdb/mnist/

Centralized learning



- **Applications**
  - image recognition (face ID)
  - next word prediction (keyboard, chatGPT)
  - wind turbines (blade icing prediction, wind power forecasting)

Centralized learning



- Training/learning is to optimize parameters z via

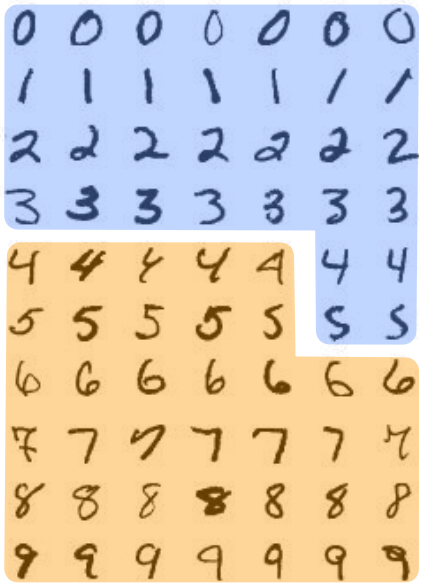
$$\min_{z \in \mathcal{Z}} \ell(z) := \frac{1}{N} \sum_{i=1}^{N} \text{dist}\left(f_z(x_i), y_i\right) + \text{reg}(z),$$

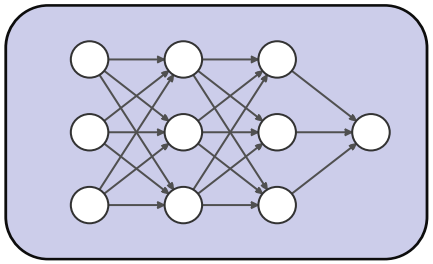
$$z := z - \eta \nabla \ell(z).$$

Centralized learning

Client 1

Client 2





Parameterized by $z \in \mathcal{Z}$

0
1
2
3
4
5
6
7
8
9

$x \in \mathcal{X}$ $\quad f_z : \mathcal{X} \to \mathcal{Y}$ $\quad y \in \mathcal{Y}$

- **Constraints**: distributed/private/heterogeneous data
  - smartphones (keyboard inputs, photo libraries)
  - hospitals/banks (medical/financial records)
  - energy companies (business statistics)

Federated learning - Definition

Federated learning (FL) [McMahan et al. 2017]

FL is a machine learning setting where multiple clients collaborate in solving a machine learning problem, under the coordination of a central server, without sharing their private data.

Challenges:

1. Efficiency: How to accelerate training under distributed data?
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?

Federated learning - Definition

Federated learning (FL) [McMahan et al. 2017]

FL is a machine learning setting where multiple clients collaborate in solving a machine learning problem, under the coordination of a central server, without sharing their private data.

Challenges:

1. Efficiency: How to accelerate training under distributed data?
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?

Federated learning - Definition

Federated learning (FL) [McMahan et al. 2017]

FL is a machine learning setting where multiple clients collaborate in solving a machine learning problem, under the coordination of a central server, without sharing their private data.

Challenges:

1. Efficiency: How to accelerate training under distributed data?
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?

Problem formulation of FL

- FL training problem:

$$\min_{z\in\mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(z), \tag{P1}$$

where ℓ_i is the local loss function, and α_i is the weight.

- By using a consensus constraint:

$$\min_{u_i, z\in\mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(u_i), \text{ s.t. } u_i = z, \forall\, i \in [m], \tag{P2}$$

where

- u_i is the local model parameter held by client i,
- z is the global model parameter held by the server.

Problem formulation of FL

- FL training problem:

$$\min_{z\in\mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(z), \tag{P1}$$

where ℓ_i is the local loss function, and α_i is the weight.

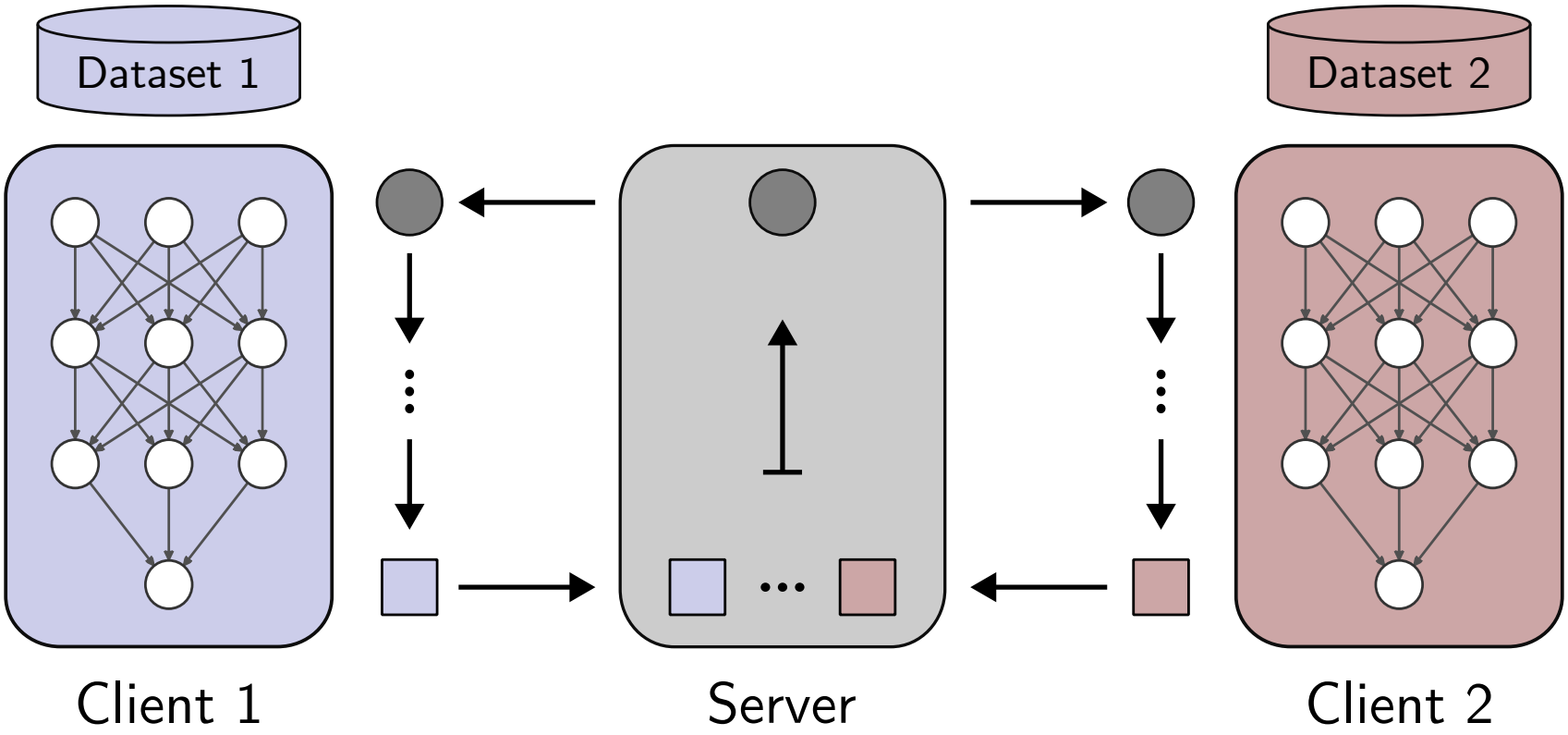
- By using a consensus constraint:

$$\min_{u_i, z\in\mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(u_i), \text{ s.t. } u_i = z, \forall\, i \in [m], \tag{P2}$$

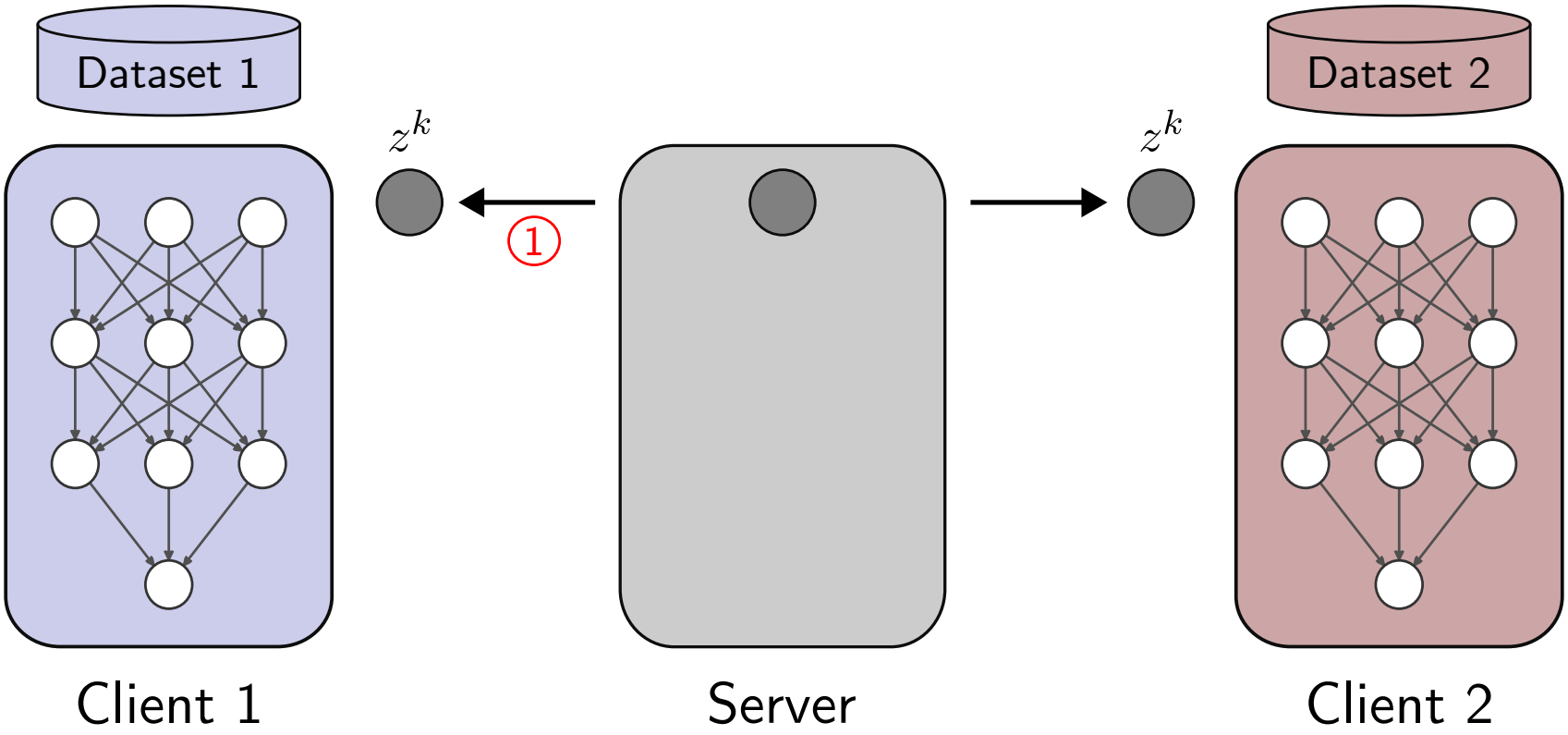
where

- u_i is the local model parameter held by client i,
- z is the global model parameter held by the server.

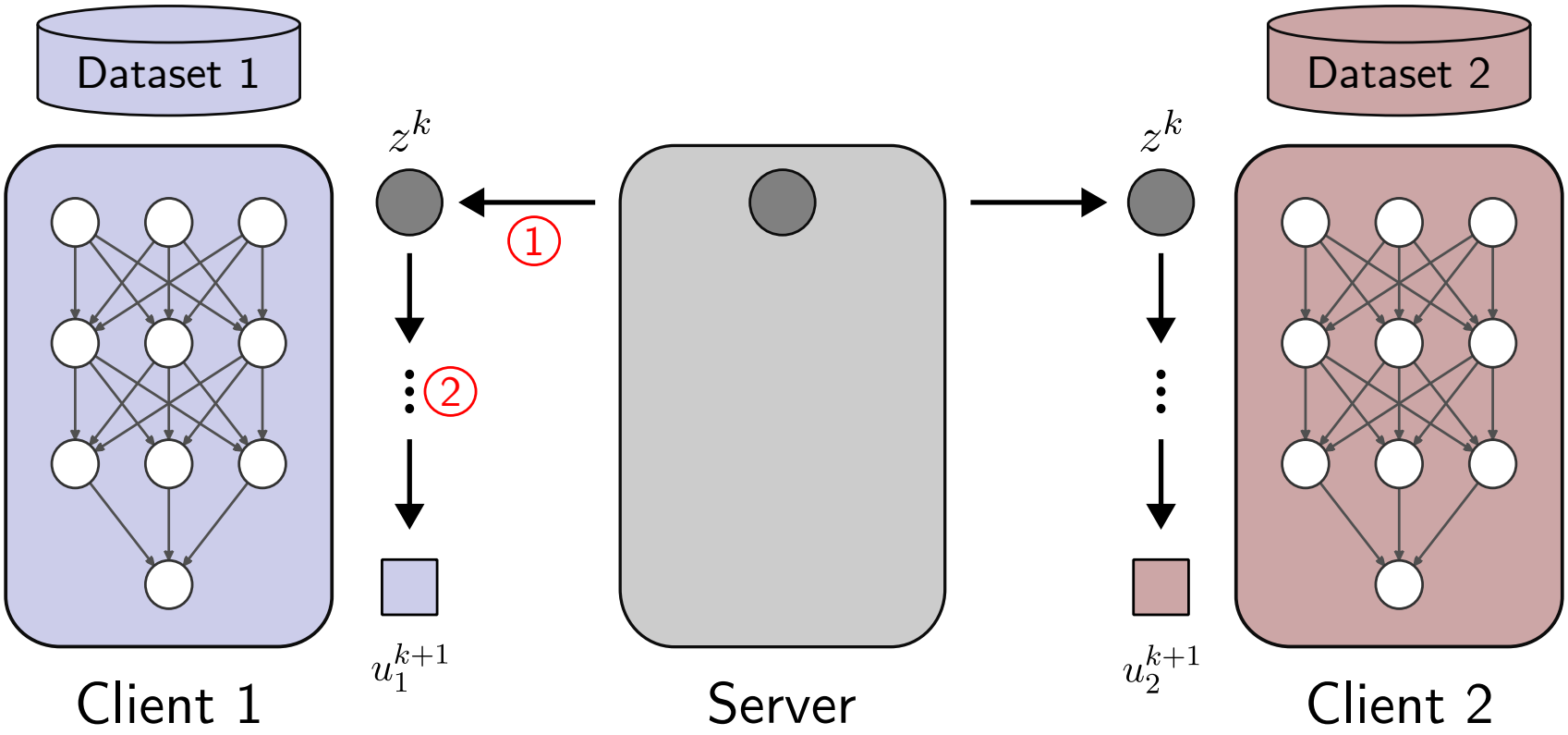
Training of FL: Exchange parameters instead of data



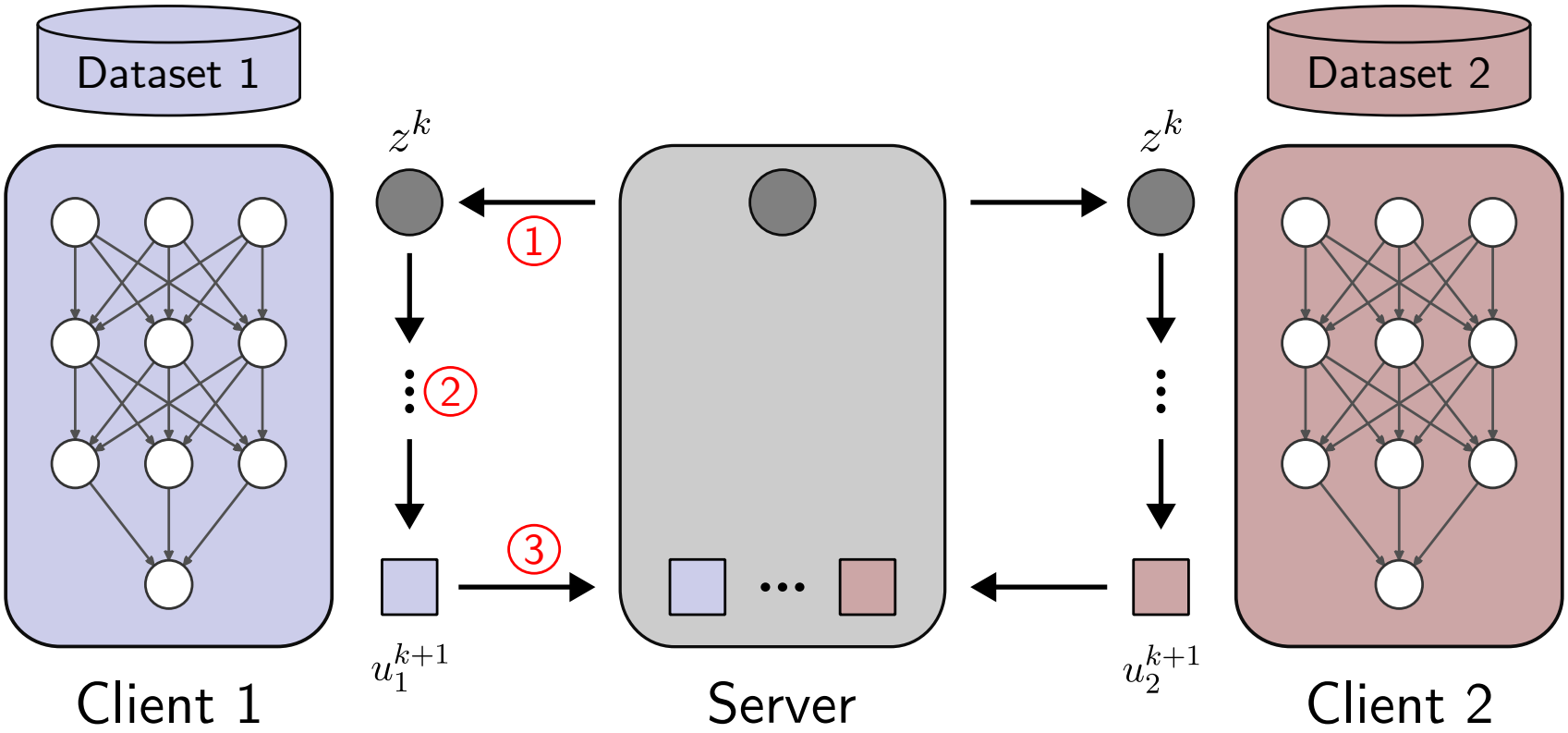
Training of FL: Exchange parameters instead of data



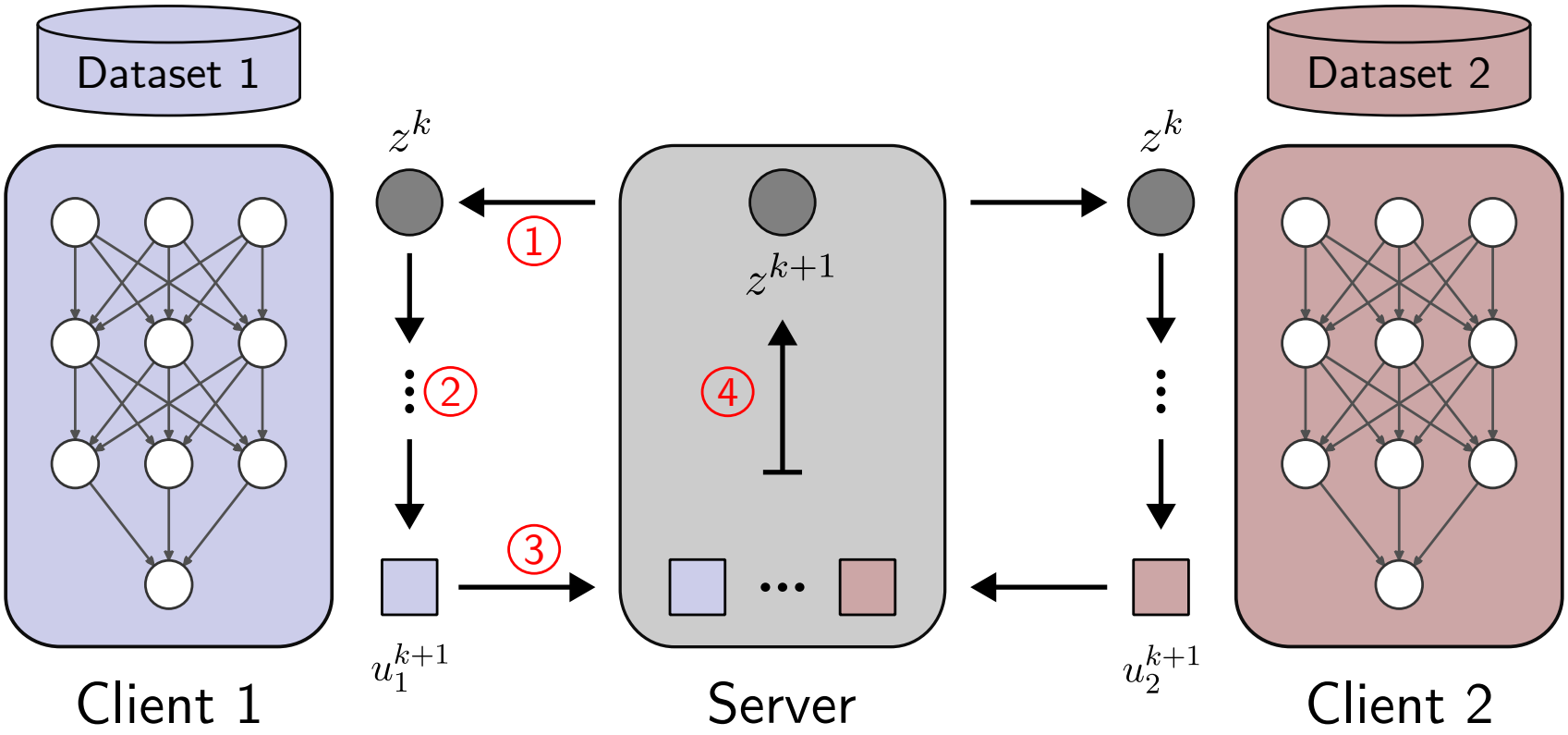
Training of FL: Exchange parameters instead of data



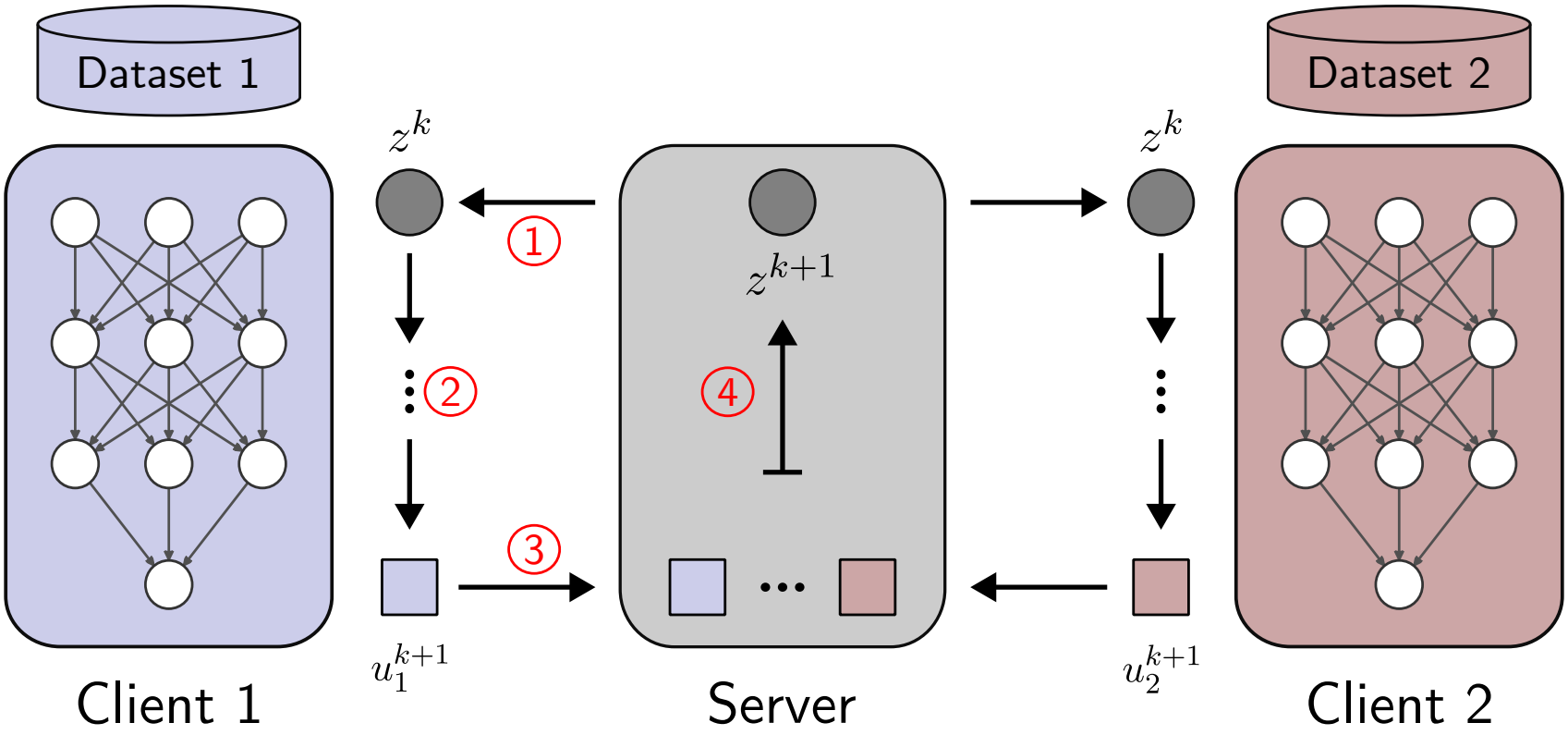
Training of FL: Exchange parameters instead of data



Training of FL: Exchange parameters instead of data



Training of FL: Exchange parameters instead of data



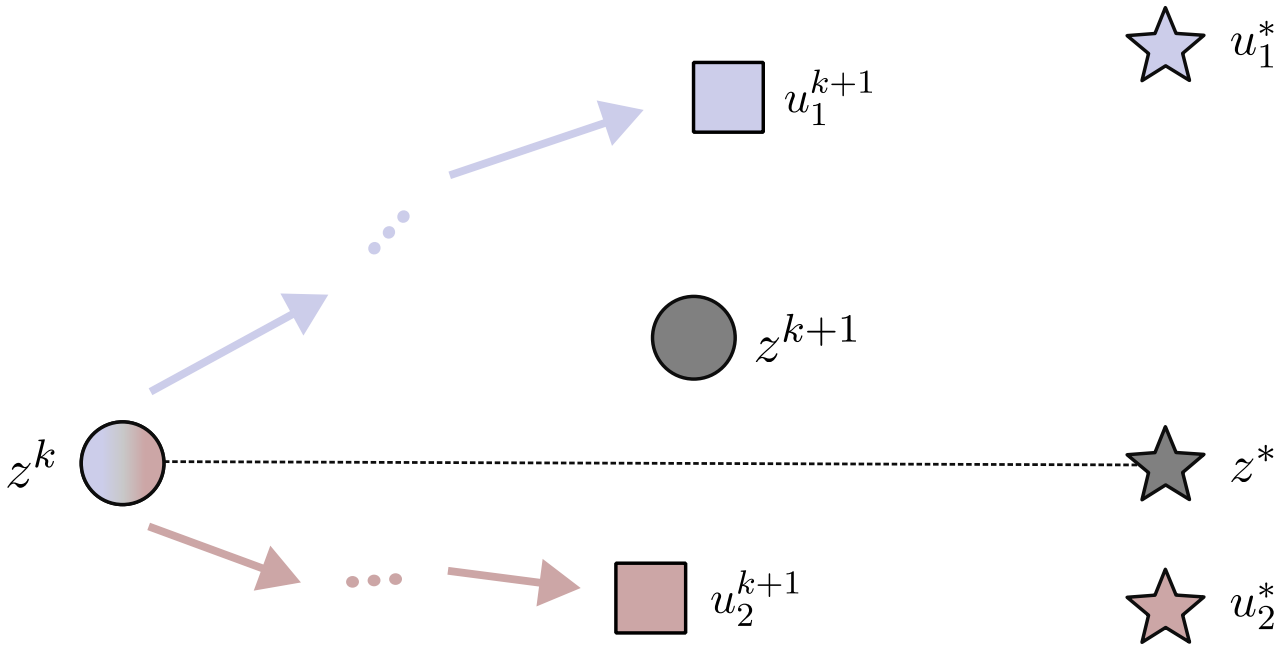
Client drift issue in FL

- The heterogeneous data results in $\mathbb{E}\left[u_i^*\right] \neq z^*$.

Client drift issue in FL

- The heterogeneous data results in $\mathbb{E}\left[u_i^*\right] \neq z^*$.

Client drift issue in FL



- The heterogeneous data results in $\mathbb{E}\left[u_i^*\right] \neq z^*$.
- Related work: FedProx (20), FedPD (21), FedADMM (23)
- Our solution: FedADMM-InSa[1]

  An inexact and self-adaptive ADMM for FL.

[1] Song, Y., Wang, Z., & Zuazua, E. (2024). FedADMM-InSa: An Inexact and Self-Adaptive ADMM for Federated Learning, submitted to *Neural Networks*, under review.

Alternating direction method of multipliers for FL

- Recall the FL problem:

$$\min_{u_i, z \in \mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(u_i), \text{ s.t. } u_i = z, \forall\, i \in [m].$$

- The augmented Lagrangian:

$$L(u, \lambda, z) = \sum_{i=1}^{m} \alpha_i \Big[\overbrace{\underbrace{\ell_i(u_i)}_{\text{old loss}} - \underbrace{\lambda_i{}^{\top} (u_i - z)}_{\text{Lagrange term}} + \underbrace{\frac{\beta_i}{2} \|u_i - z\|^2}_{\text{penalty term}}}^{\text{Each client's new loss } L_i(u_i, \lambda_i, z)} \Big].$$

- The alternating direction method of multipliers (ADMM) [Glowinski et al. 1975] optimize (u, λ, z) iteratively.

Alternating direction method of multipliers for FL

- Recall the FL problem:

$$\min_{u_i, z \in \mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(u_i), \text{ s.t. } u_i = z, \forall\, i \in [m].$$

- The augmented Lagrangian:

$$L(u, \lambda, z) = \sum_{i=1}^{m} \alpha_i \Big[\overbrace{\underbrace{\ell_i(u_i)}_{\text{old loss}} - \lambda_i{}^{\top}(u_i - z)}^{\text{Lagrange term}} + \overbrace{\frac{\beta_i}{2}\|u_i - z\|^2}^{\text{penalty term}} \Big]_{\text{Each client's new loss } L_i(u_i, \lambda_i, z)}.$$

- The alternating direction method of multipliers (ADMM) [Glowinski et al. 1975] optimize (u, λ, z) iteratively.

Alternating direction method of multipliers for FL

- Recall the FL problem:

$$\min_{u_i, z \in \mathbb{R}^n} \sum_{i=1}^{m} \alpha_i \ell_i(u_i), \text{ s.t. } u_i = z, \forall\, i \in [m].$$

- The augmented Lagrangian:

$$L(u, \lambda, z) = \sum_{i=1}^{m} \alpha_i \Big[\overbrace{\ell_i(u_i)}^{\text{old loss}} - \overbrace{\lambda_i{}^\top (u_i - z)}^{\text{Lagrange term}} + \overbrace{\frac{\beta_i}{2} \|u_i - z\|^2}^{\text{penalty term}} \Big]_{\text{Each client's new loss } L_i(u_i, \lambda_i, z)}.$$

- The alternating direction method of multipliers (ADMM) [Glowinski et al. 1975] optimize (u, λ, z) iteratively.

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

- For subproblem (S1), perform E steps (empirically):

$$u_i^{k,e+1} = u_i^{k,e} - \eta_i \nabla_{u_i} L_i\left(u_i^{k,e}, \lambda_i^k, z^k\right),\ e \in [E].$$

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

- For subproblem (S3):

$$z^{k+1} = \frac{1}{\sum_{i=1}^m \alpha_i \beta_i} \sum_{i=1}^m \alpha_i \left(\beta_i u_i^{k+1} - \lambda_i^{k+1}\right).$$

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

- **Q1:** How to decide the solving precision of (S1)?
- **Q2:** How to choose the penalty parameter β?

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

- **Q1:** How to decide the solving precision of (S1)? **Inexactness**
- **Q2:** How to choose the penalty parameter β?

FedADMM algorithm

- Client's local update

$$u_i^{k+1} = \arg\min_{u_i \in \mathbb{R}^n} L_i\left(u_i, \lambda_i^k, z^k\right), \tag{S1}$$

$$\lambda_i^{k+1} = \lambda_i^k - \beta_i\left(u_i^{k+1} - z^k\right). \tag{S2}$$

- Server's aggregation

$$z^{k+1} = \arg\min_{z \in \mathbb{R}^n} L\left(u^{k+1}, \lambda^{k+1}, z\right). \tag{S3}$$

- **Q1:** How to decide the solving precision of (S1)? **Inexactness**
- **Q2:** How to choose the penalty parameter β? **Self-adaptive**

FedADMM-**In**: **Inexactness** criterion for finding u_i^{k+1}

- For each client i, define $e_i^k(u_i)$ as

$$e_i^k(u_i) = \nabla_{u_i} L_i(u_i, \lambda_i^k, z^k).$$

- Then, each client i finds u_i^{k+1} such that

$$\|e_i^k(u_i^{k+1})\| \leqslant \sigma_i \|e_i^k(u_i^k)\|,$$

where σ_i is a given constant satisfying

$$0 < \sigma_i < \frac{\sqrt{2}}{\sqrt{2} + \sqrt{\tilde{\beta}_i}} < 1,$$

with $\tilde{\beta}_i = \beta_i / c_i$, and c_i is a constant.

FedADMM-**In**: **Inexactness** criterion for finding u_i^{k+1}

- For each client i, define $e_i^k(u_i)$ as

$$e_i^k(u_i) = \nabla_{u_i} L_i(u_i, \lambda_i^k, z^k).$$

- Then, each client i finds u_i^{k+1} such that

$$\|e_i^k(u_i^{k+1})\| \leqslant \sigma_i \|e_i^k(u_i^k)\|,$$

where σ_i is a given constant satisfying

$$0 < \sigma_i < \frac{\sqrt{2}}{\sqrt{2} + \sqrt{\tilde{\beta}_i}} < 1,$$

with $\tilde{\beta}_i = \beta_i / c_i$, and c_i is a constant.

Convergence analysis of FedADMM-In

Theorem (Song, Wang, Zuazua, 2024)

Assume the gradient of ℓ_i is Lipschitz and ℓ_i is c_i-strongly convex. Let $\{(u^k, \lambda^k, z^k)\}$ be the sequence generated by FedADMM-In (full clients participation). Then, for all $i \in [m]$, we have

$$u_i^k \overset{k\to\infty}{\longrightarrow} u_i^*,$$
$$\lambda_i^k \overset{k\to\infty}{\longrightarrow} \lambda_i^*,$$
$$z^k \overset{k\to\infty}{\longrightarrow} z^*.$$

The proof is inspired by [Glowinski et al. 2022, Sec. 3.3][1].

[1] Glowinski, R., Song, Y., Yuan, X., & Yue, H. (2022). Application of the alternating direction method of multipliers to control constrained parabolic optimal control problems and beyond. Ann. Appl. Math, 38(2), 115-158.

FedADMM-In**Sa**: **Self-adaptive** penalty parameter β_i^k

- Define the primal residual p_i^k and the dual residual d_i^k:

$$p_i^k = \beta_i^k \|u_i^{k+1} - u_i^k\|,$$
$$d_i^k = \|u_i^{k+1} - z^k\|.$$

- Then, update β_i^k by the following scheme:

$$\beta_i^{k+1} = \begin{cases} \beta_i^k \tau, & \text{if } d_i^k > \mu p_i^k, \\ \beta_i^k / \tau, & \text{if } p_i^k > \mu d_i^k, \\ \beta_i^k, & \text{otherwise}, \end{cases}$$

where $\mu, \tau > 1$, e.g. $\mu = 5$ and $\tau = 2$.

Remark (Compatibility with the inexactness criterion.)

$$\|e_i^k(u_i^{k+1})\| \leq \sigma_i^k \|e_i^k(u_i^k)\|, \, 0 < \sigma_i^k < \frac{\sqrt{2}}{\sqrt{2} + \sqrt{\beta_i^k / c_i}}.$$

FedADMM-In**Sa**: **Self-adaptive** penalty parameter β_i^k

- Define the primal residual p_i^k and the dual residual d_i^k:

$$p_i^k = \beta_i^k \|u_i^{k+1} - u_i^k\|,$$
$$d_i^k = \|u_i^{k+1} - z^k\|.$$

- Then, update β_i^k by the following scheme:

$$\beta_i^{k+1} = \begin{cases} \beta_i^k \tau, & \text{if } d_i^k > \mu p_i^k, \\ \beta_i^k / \tau, & \text{if } p_i^k > \mu d_i^k, \\ \beta_i^k, & \text{otherwise }, \end{cases}$$

where $\mu, \tau > 1$, e.g. $\mu = 5$ and $\tau = 2$.

Remark (Compatibility with the inexactness criterion.)

$$\left\|e_i^k(u_i^{k+1})\right\| \leq \sigma_i^k \left\|e_i^k(u_i^k)\right\|, 0 < \sigma_i^k < \frac{\sqrt{2}}{\sqrt{2} + \sqrt{\beta_i^k / c_i}}.$$

FedADMM-In**Sa**: **Self-adaptive** penalty parameter β_i^k

- Define the primal residual p_i^k and the dual residual d_i^k:

$$p_i^k = \beta_i^k \|u_i^{k+1} - u_i^k\|,$$
$$d_i^k = \|u_i^{k+1} - z^k\|.$$

- Then, update β_i^k by the following scheme:

$$\beta_i^{k+1} = \begin{cases} \beta_i^k \tau, & \text{if } d_i^k > \mu p_i^k, \\ \beta_i^k / \tau, & \text{if } p_i^k > \mu d_i^k, \\ \beta_i^k, & \text{otherwise }, \end{cases}$$

where $\mu, \tau > 1$, e.g. $\mu = 5$ and $\tau = 2$.

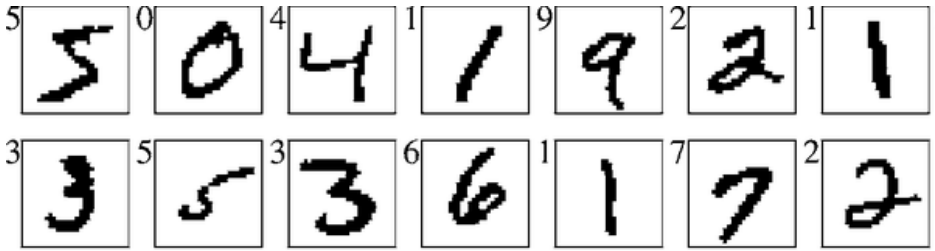
Remark (Compatibility with the inexactness criterion.)

$$\left\|e_i^k(u_i^{k+1})\right\| \leq \sigma_i^k \left\|e_i^k(u_i^k)\right\|, \, 0 < \sigma_i^k < \frac{\sqrt{2}}{\sqrt{2} + \sqrt{\beta_i^k / c_i}}.$$

Simulation setups

- We compare our FedADMM-In/InSa with vanilla FedADMM [Zhou et al. 2023] and FedAvg [McMahan et al. 2017].

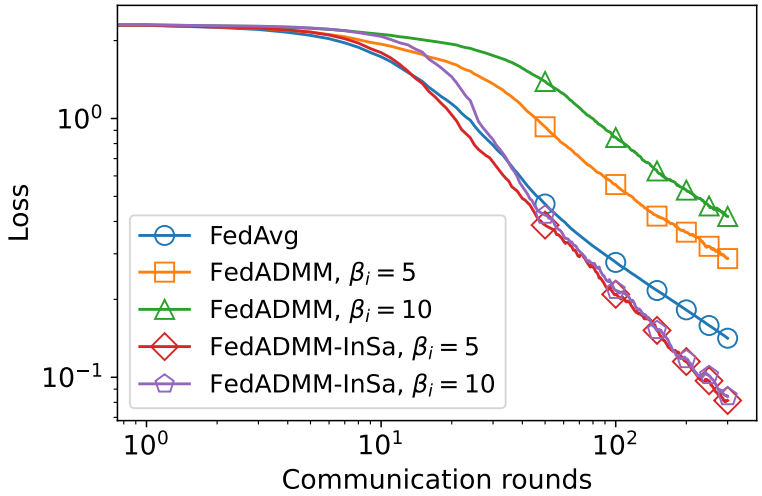
	Example 1 Linear regression	**Example 2** Image classification
Dataset	Synthetic	MNIST
Model	Linear	CNN
Training set size	50000	60000
Test set size	-	10000
Data dimension	5000	$28 \times 28 \times 1$
Number of clients	200	200
Data per client	250	300



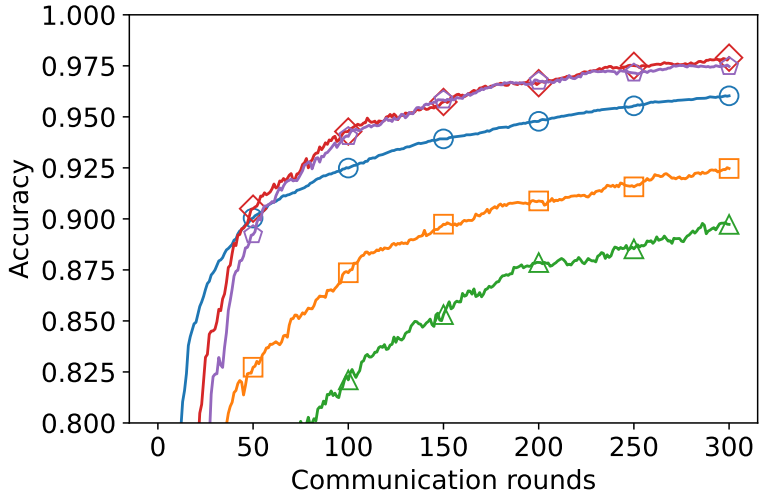
Simulation results

		Example 1 Linear regression		**Example 2** MNIST with CNN	
β_i	Algorithm	Loss	Epoch reduction	Accuracy	Epoch reduction
-	Fedavg	1.64	-	96.0%	-
1	FedADMM	1.53	-	97.7%	-
	FedADMM-In	1.55	58.5%	97.6%	58.7%
	FedADMM-InSa	**1.51**	18.8%	**97.9%**	61.0%
2	FedADMM	1.51	-	96.2%	-
	FedADMM-In	1.51	19.3%	96.7%	41.7%
	FedADMM-InSa	**1.51**	16.2%	**97.7%**	57.9%
5	FedADMM	1.51	-	92.5%	-
	FedADMM-In	1.51	3.9%	93.4%	31.1%
	FedADMM-InSa	**1.51**	12.5%	**97.9%**	55.9%
10	FedADMM	1.51	-	89.7%	-
	FedADMM-In	1.51	0.9%	90.2%	21.1%
	FedADMM-InSa	**1.51**	6.8%	**97.5%**	55.7%

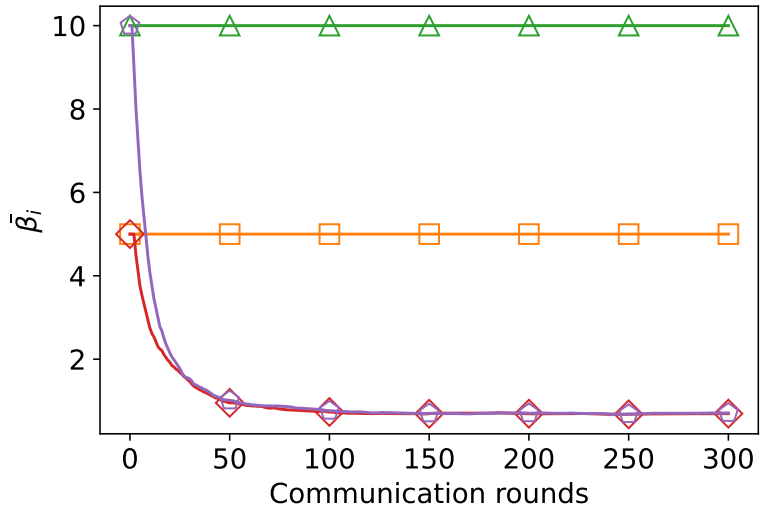
Simulation results - MNIST with CNN



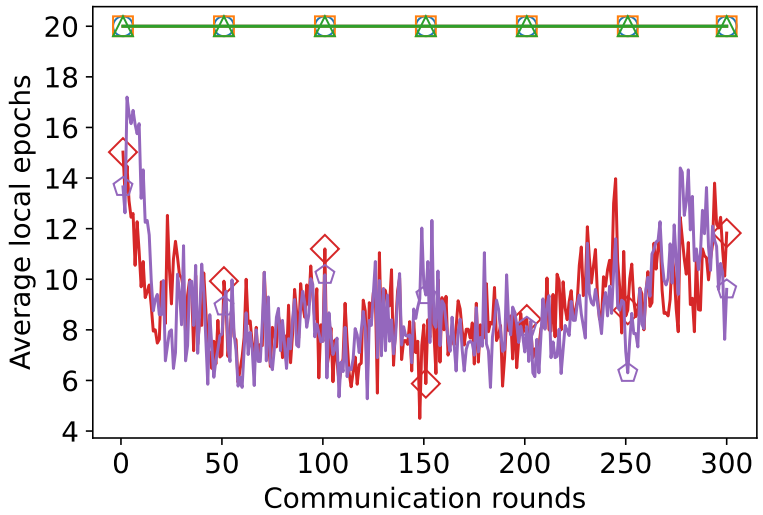
(a) Training loss.



(b) Test accuracy.

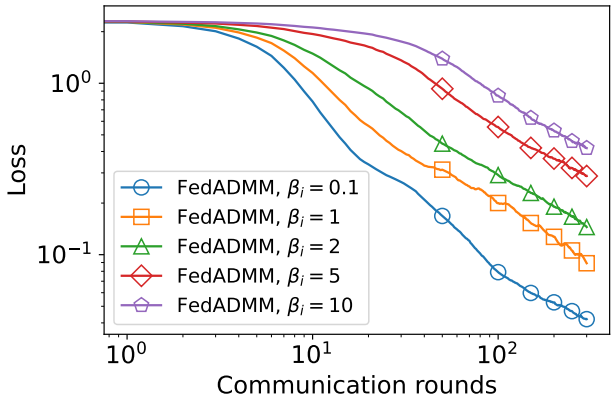


(c) Average penalty parameters.

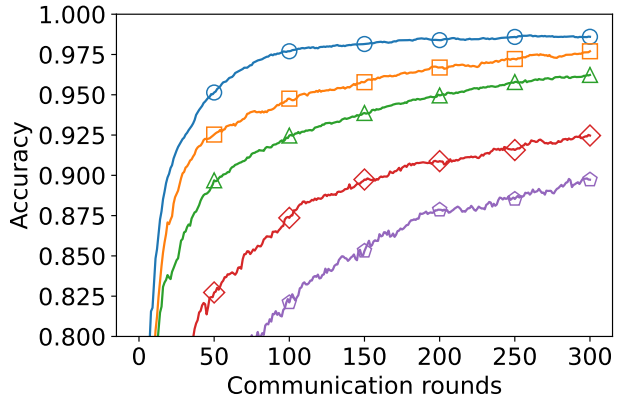


(d) Average local epochs.

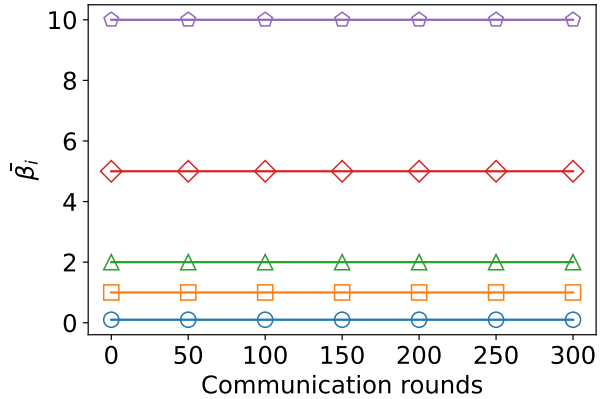
Simulation results - adaptive penalty scheme



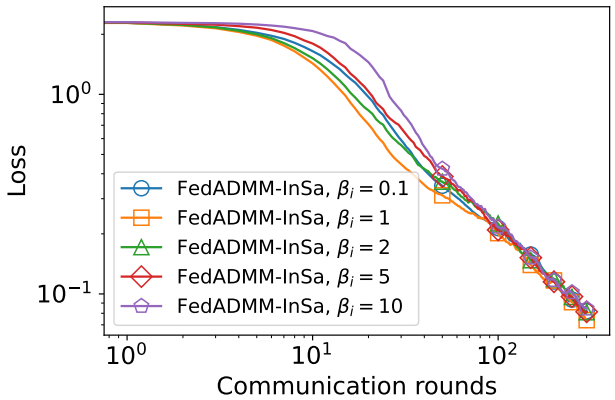
(a) Training loss.



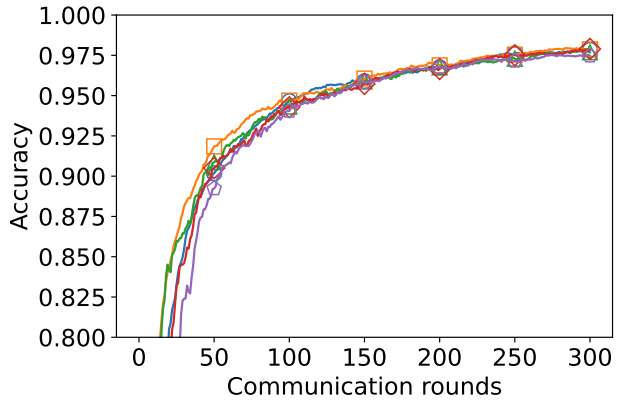
(b) Test accuracy.



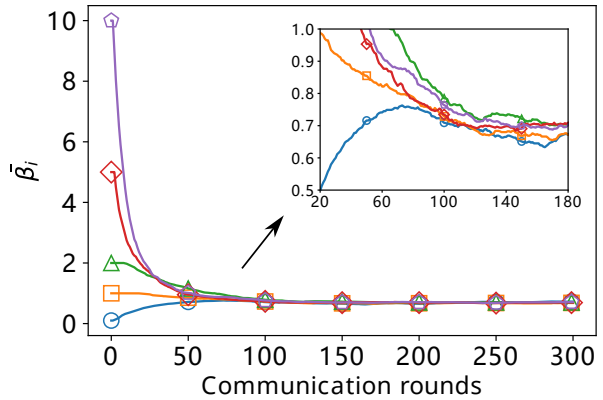
(c) Average penalty.



(d) Training loss.



(e) Test accuracy.



(f) Average penalty.

FedADMM (top row) vs. our FedADMM-InSa (bottom row).

Conclusions and Perspectives

1. Efficiency: How to accelerate training under distributed data?
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?

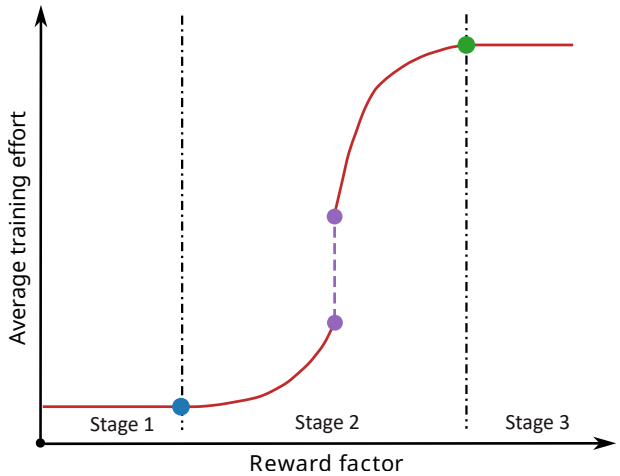
Conclusions and Perspectives

1. Efficiency: An **inexact** and **self-adaptive** FedADMM[1].
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?

[1] Song, Y., Wang, Z., & Zuazua, E. (2024). FedADMM-InSa: An Inexact and Self-Adaptive ADMM for Federated Learning, submitted to *Neural Networks*, under review.

Conclusions and Perspectives

1. Efficiency: An **inexact** and **self-adaptive** FedADMM[1].
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?



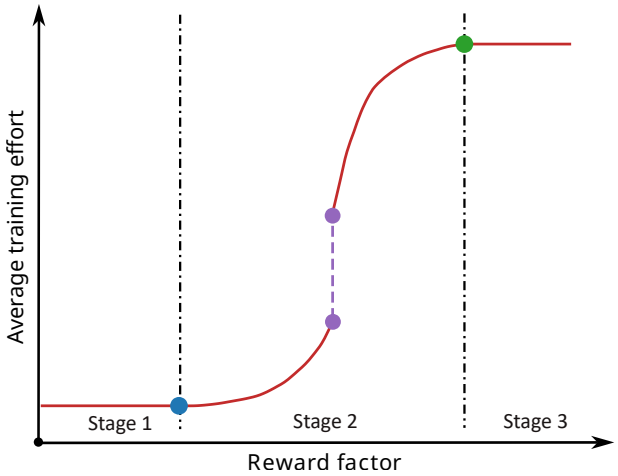
Incentive mechanism[2].

[1] Song, Y., Wang, Z., & Zuazua, E. (2024). FedADMM-InSa: An Inexact and Self-Adaptive ADMM for Federated Learning, submitted to *Neural Networks*, under review.

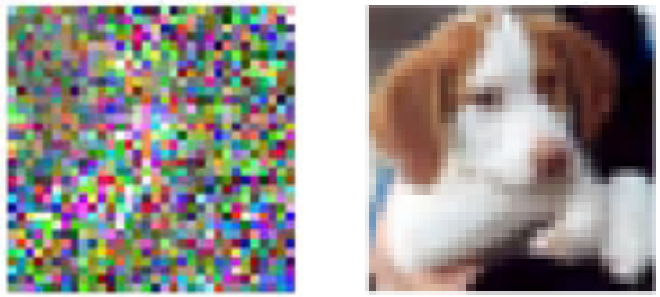
[2] Liu, K., Wang, Z., & Zuazua, E. (2024). Incentive Mechanisms in Federated Learning: A Potential Game-theoretical Perspective, in preparation.

Conclusions and Perspectives

1. Efficiency: An **inexact** and **self-adaptive** FedADMM[1].
2. Incentive: How to motivate clients to cooperate?
3. Privacy: How to safeguard clients' private data?



Incentive mechanism[2].



Data reconstruction attack[3].

[1] Song, Y., Wang, Z., & Zuazua, E. (2024). FedADMM-InSa: An Inexact and Self-Adaptive ADMM for Federated Learning, submitted to *Neural Networks*, under review.

[2] Liu, K., Wang, Z., & Zuazua, E. (2024). Incentive Mechanisms in Federated Learning: A Potential Game-theoretical Perspective, in preparation.

[3] Song, Y., Wang, Z., & Zuazua, E. (2023). Approximate and Weighted Data Reconstruction Attack in Federated Learning, submitted to *IEEE Transactions on Big Data*, under review.

Thanks for your attention!

Friedrich-Alexander-Universität
DYNAMICS, CONTROL,
MACHINE LEARNING
AND NUMERICS

MODCONFLEX
Modelling and control of flexible
structures interacting with fluids

Funded by
the European Union

References

- Glowinski, R. et al. (1975). "Sur l'approximation, Par Éléments Finis d'ordre Un, et La Résolution, Par Pénalisation-Dualité d'une Classe de Problèmes de Dirichlet Non Linéaires". In: *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique* 9.R2, pp. 41–76.
- Glowinski, R. et al. (2022). "Application of the Alternating Direction Method of Multipliers to Control Constrained Parabolic Optimal Control Problems and Beyond". In: *Annals of Applied Mathematics* 38.
- McMahan, B. et al. (2017). "Communication-Efficient Learning of Deep Networks from Decentralized Data". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1273–1282.
- Zhou, S. et al. (2023). "Federated Learning Via Inexact ADMM". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.8, pp. 9699–9708. ISSN: 1939-3539.