Riemannian Federated Learning via Averaging Gradient Streams with Partial Participation

Speaker: Wen Huang

Xiamen University

@Dalian University of Technology

Sep. 27. 2025

Joint work with Zhenwei Huang, Pratik Jawanpuria, and Bamdev Mishra

Outline

Outline

General Federated Learning Optimization:

$$\min_{x\in\mathbb{R}^n} F(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{1.1}$$

- N is the number of agents;
- f_i is the local objective of agent i, and covers

$$f_i(x) = \begin{cases} \mathbb{E}_{\xi\sim\mathcal{D}_i}[f_i(x;\xi)] & \text{with } \mathcal{D}_i \text{ being a local data distribution} \\ \frac{1}{N_i}\sum_{i=1}^{N_i} f_i(x; z_{i,j}) & \text{with } \mathcal{D}_i = \{z_{i,1}, \ldots, z_{i,N_i}\} \text{ being a local dataset;} \end{cases}$$

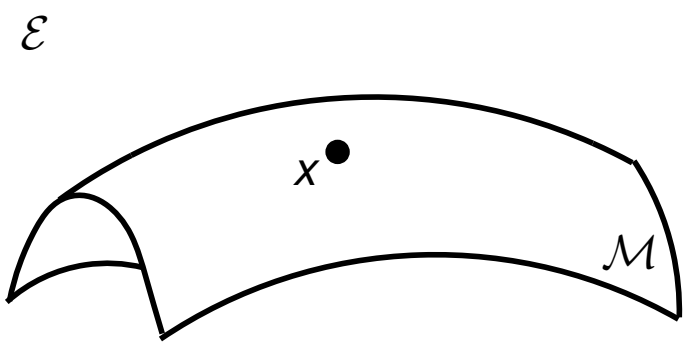
General Federated Learning Optimization:

$$\min_{x \in \mathcal{M}} F(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x), \tag{1.1}$$

- N is the number of agents;
- f_i is the local objective of agent i, and covers

$$f_i(x) = \begin{cases} \mathbb{E}_{\xi \sim \mathcal{D}_i}[f_i(x;\xi)] & \text{with } \mathcal{D}_i \text{ being a local data distribution} \\ \frac{1}{N_i} \sum_{i=1}^{N_i} f_i(x; z_{i,j}) & \text{with } \mathcal{D}_i = \{z_{i,1}, \ldots, z_{i,N_i}\} \text{ being a local dataset;} \end{cases}$$

Riemannian Federated Learning considers (1.1) with x in a manifold $\mathcal{M}$



General Federated Learning Optimization:

$$\min_{x \in \mathcal{M}} F(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x), \tag{1.1}$$

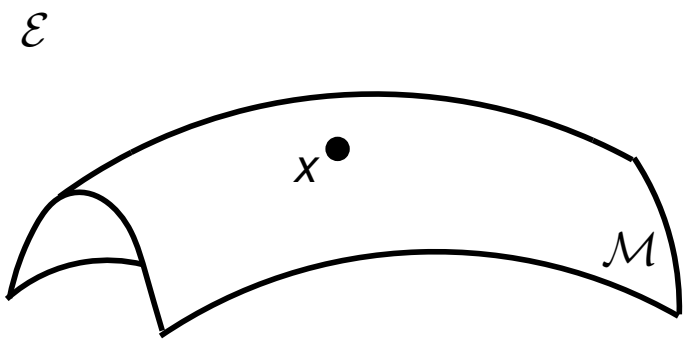
- N is the number of agents;
- f_i is the local objective of agent i, and covers

$$f_i(x) = \begin{cases} \mathbb{E}_{\xi \sim \mathcal{D}_i}[f_i(x;\xi)] & \text{with } \mathcal{D}_i \text{ being a local data distribution} \\ \frac{1}{N_i} \sum_{i=1}^{N_i} f_i(x; z_{i,j}) & \text{with } \mathcal{D}_i = \{z_{i,1}, \ldots, z_{i,N_i}\} \text{ being a local dataset;} \end{cases}$$

Riemannian Federated Learning considers (1.1) with x in a manifold $\mathcal{M}$

Applications:

- Matrix completion
- Principal component analysis
- Online learning
- Taxonomy embedding
- etc



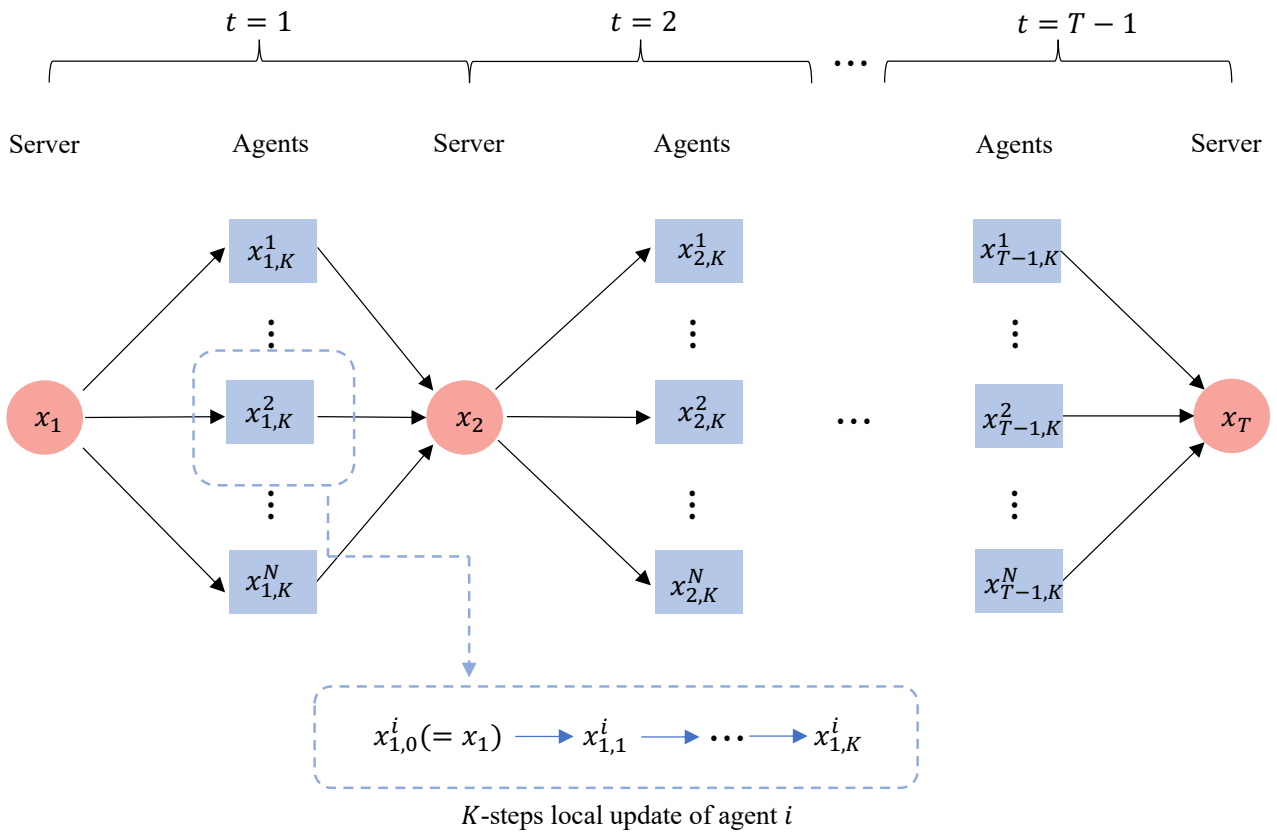


Figure 1: Flowchart of a federated learning algorithm

Euclidean version:

Algorithm: A representative federated averaging algorithm [4]

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of S agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j;$$

9. **end for**

$\min_{x\in\mathbb{R}^n} f_i(x)$

- Sever: Steps 2, 3, and 8;
- Agents: Steps 5 and 6;

[4] B. McMahan, E. Moore, D. Ramage, B. A. y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. Proceedings of Machine Learning Research, 54, P.1273-1282, 2017.

Euclidean version:

Algorithm: A representative federated averaging algorithm [4]

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of S agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j;$$

9. **end for**

$\min_{x\in\mathbb{R}^n} f_i(x)$

- Sever: Steps 2, 3, and 8;
- Agents: Steps 5 and 6;

[4] B. McMahan, E. Moore, D. Ramage, B. A. y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. Proceedings of Machine Learning Research, 54, P.1273-1282, 2017.

Euclidean to Riemannian

Algorithm: A Riemannian federated learning algorithm

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of s agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step Riemannian SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \text{ave}(x_{t,K}^j \mid j \in \mathcal{S}_j);$$

9. **end for**

$\boxed{\min_{x\in\mathcal{M}} f_i(x)}$

- Agents: Riemannian SGD [1]
- Sever: Aggregation

Euclidean to Riemannian

Algorithm: A Riemannian federated learning algorithm

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of s agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step Riemannian SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j \in \mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \text{ave}(x_{t,K}^j \mid j \in \mathcal{S}_j);$$

9. **end for**

$\min_{x \in \mathcal{M}} f_i(x)$

- Agents: Riemannian SGD [1]
- Sever: Aggregation

Euclidean to Riemannian

Algorithm: A Riemannian federated learning algorithm

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of s agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step Riemannian SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \text{ave}(x_{t,K}^j \mid j \in \mathcal{S}_j);$$

9. **end for**

$\min_{x\in\mathcal{M}} f_i(x)$

- Agents: Riemannian SGD [1]
- Sever: Aggregation

[1]: S. Bonnabel. "Stochastic gradient descent on Riemannian manifolds", *IEEE Transactions on Automatic Control*, 58:9, 2217-2229, 2013.

Euclidean to Riemannian

Algorithm: A Riemannian federated learning algorithm

1. **for** $t = 0, 1, \ldots, T-1$ **do**
2. The server uniformly selects a subset $\mathcal{S}_t$ of s agents at random;
3. The server upload global parameter x_t to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow x_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Agent j updates a local parameter $x_{t,K}^j$ by K-step Riemannian SGD with x_t being initial iterate;
6. Sent $x_{t,K}^j$ to the server;
7. **end for**
8. Server aggregates the received local parameters $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ by averaging

$$x_{t+1} \leftarrow \text{ave}(x_{t,K}^j \mid j \in \mathcal{S}_j);$$

9. **end for**

$\boxed{\min_{x\in\mathcal{M}} f_i(x)}$

- Agents: Riemannian SGD [1]
- Sever: Aggregation

How to aggregates $\{x_{t,K}^j\}_{j\in\mathcal{S}_t}$ on a manifold?

[1]: S. Bonnabel. "Stochastic gradient descent on Riemannian manifolds", *IEEE Transactions on Automatic Control*, 58:9, 2217-2229, 2013.

Euclidean to Riemannian (Aggregation):

- Naive generalization:

$$x_{t+1} \leftarrow \frac{1}{S} \sum_{j \in \mathcal{S}_t} x_{t,K}^j \not\Longrightarrow \text{Riemannian setting}$$

Euclidean to Riemannian (Aggregation):

- Naive generalization:

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \not\Longrightarrow \text{Riemannian setting}$$

- An alternative approach:

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \|x - x_{t,K}^j\|_F^2$$

$$\Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \mathrm{dist}^2(x, x_{t,K}^j) \Longrightarrow \text{Riemannian setting;}$$

Euclidean to Riemannian (Aggregation):

- Naive generalization:

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \not\Longrightarrow \text{Riemannian setting}$$

- An alternative approach:

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \|x - x_{t,K}^j\|_F^2$$

$$\Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \mathrm{dist}^2(x, x_{t,K}^j) \Longrightarrow \text{Riemannian setting;}$$

- $x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \mathrm{dist}^2(x, x_{t,K}^j)$: computationally expensive;

Euclidean to Riemannian (Aggregation):

- Naive generalization:
$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \not\Longrightarrow \text{Riemannian setting}$$
- An alternative approach:

$$x_{t+1} \leftarrow \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j \Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \|x - x_{t,K}^j\|_F^2$$

$$\Longleftrightarrow x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \mathrm{dist}^2(x, x_{t,K}^j) \Longrightarrow \text{Riemannian setting;}$$

- $x_{t+1} = \arg\min_x \frac{1}{S}\sum_{j\in\mathcal{S}_j} \mathrm{dist}^2(x, x_{t,K}^j)$: computationally expensive;
- One step of Riemannian gradient descent (called tangent mean) [3]:

$$x_{t+1} \leftarrow \mathrm{Exp}_{x_t}\left(\frac{1}{S}\sum_{j\in\mathcal{S}_t} \mathrm{Exp}_{x_t}^{-1}(x_{t,K}^j)\right); \qquad \text{(TM)}$$

[3] Jiaxiang Li and Shiqian Ma. Federated learning on Riemannian manifolds. Applied Set-Valued Analysis and Optimization, 5(2), 2023.

Existing Riemannian Federated Learning:

- Federated Learning on Riemannian Manifolds [3]
  - Integrate SVRG technique within Riemannian federated learning
  - Use tangent mean as the server aggregation
  - Requirements for convergence
    - Full agent participation, and one step of local update;
    - One agent participates, and multiple steps of local update.

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.

Existing Riemannian Federated Learning:

- Federated Learning on Riemannian Manifolds [3]
  - Integrate SVRG technique within Riemannian federated learning
  - Use tangent mean as the server aggregation
  - Requirements for convergence
    - Full agent participation, and one step of local update;
    - One agent participates, and multiple steps of local update.

- Federated Learning on Riemannian Manifolds with Differential Privacy [2]
  - Use differential privacy to enhance the privacy of federated learning;
  - Use tangent mean as the server aggregation.
  - Requirements for convergence similar to those in [3].

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[2] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.

Existing Riemannian Federated Learning:

- Federated Learning on Riemannian Manifolds [3]
  - Integrate SVRG technique within Riemannian federated learning
  - Use tangent mean as the server aggregation
  - Requirements for convergence
    - Full agent participation, and one step of local update;
    - One agent participates, and multiple steps of local update.

- Federated Learning on Riemannian Manifolds with Differential Privacy [2]
  - Use differential privacy to enhance the privacy of federated learning;
  - Use tangent mean as the server aggregation.
  - Requirements for convergence similar to those in [3].

- Riemannian Federated Learning on Compact Submanifolds with Heterogeneous Data [6]
  - Use projection onto the manifold
  - Require full agents to participate while allow multiple local updates

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[2] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.
[6] J. Zhang and J. Hu and A. M.-C. So and M. Johansson. Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data. NeurIPS, 2024.

Limitations

- Full agent participation and one step of local update [3, 2]
- Compact submanifolds embedded in Euclidean spaces [6]

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[2] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.
[6] J. Zhang and J. Hu and A. M.-C. So and M. Johansson. Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data. NeurIPS, 2024.

Limitations

- Full agent participation and one step of local update [3, 2]
- Compact submanifolds embedded in Euclidean spaces [6]

Our goals

- Design a new server aggregation working for generic manifolds without the inverse of exponential (even retraction)
- Allow partial participation

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[2] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.
[6] J. Zhang and J. Hu and A. M.-C. So and M. Johansson. Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data. NeurIPS, 2024.

Outline

Euclidean aggregation: $x_{t+1} = \frac{1}{S}\sum_{j\in\mathcal{S}_t} x^j_{t,K}$

$$x^j_{t,K} = x^j_{t,K-1} - \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b\in\mathcal{B}^j_{t,k}} \nabla f_j(x^j_{t,K-1}; \xi^j_{t,K-1,b}) = \cdots$$

$$= x_t - \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b\in\mathcal{B}^j_{t,k}} \nabla f_j(x^j_{t,k}; \xi^j_{t,k,b})$$

$$\implies x_{t+1} - x_t = -\frac{1}{S}\sum_{j\in\mathcal{S}_t}\sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b\in\mathcal{B}^j_{t,k}} \nabla f_j(x^j_{t,k}; \xi^j_{t,k,b}).$$

Euclidean aggregation: $x_{t+1} = \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j$

$p_t = x_{t+1} - x_t = -\frac{1}{S}\sum_{j\in\mathcal{S}_t}\sum_{k=0}^{K-1}\frac{\alpha_{t,k}}{B_{t,k}}\sum_{b\in\mathcal{B}_{t,k}^j}\nabla f_j(x_{t,k}^j;\xi_{t,k,b}^j)$

Tangent mean: $x_{t+1} = \mathrm{Exp}_{x_t}\left(\frac{1}{S}\sum_{i\in\mathcal{S}_t}\mathrm{Exp}_{x_t}^{-1}(x_{t,K}^j)\right)$

$$x_{t,K}^j = \mathrm{Exp}_{x_{t,K-1}^j}\left(-\frac{\alpha_{t,K-1}}{B_{t,K-1}}\sum_{b\in\mathcal{B}_{t,K-1}^j}\mathrm{grad} f_j(x_{t,K-1}^j;\xi_{t,K-1,b}^j)\right)$$

$$\cdots$$

$$x_{t,1}^j = \mathrm{Exp}_{x_t}\left(-\frac{\alpha_{t,0}}{B_{t,0}}\sum_{b\in\mathcal{B}_{t,0}^j}\mathrm{grad} f_j(x_{t,0}^j;\xi_{t,0,b}^j)\right)$$

Euclidean aggregation: $x_{t+1} = \frac{1}{S}\sum_{j\in\mathcal{S}_t} x_{t,K}^j$

$p_t = x_{t+1} - x_t = -\frac{1}{S}\sum_{j\in\mathcal{S}_t}\sum_{k=0}^{K-1}\frac{\alpha_{t,k}}{B_{t,k}}\sum_{b\in\mathcal{B}_{t,k}^j}\nabla f_j(x_{t,k}^j;\xi_{t,k,b}^j)$

Tangent mean: $x_{t+1} = \mathrm{Exp}_{x_t}\left(\frac{1}{S}\sum_{i\in\mathcal{S}_t}\mathrm{Exp}_{x_t}^{-1}(x_{t,K}^j)\right)$

$$x_{t,K}^j = \mathrm{Exp}_{x_{t,K-1}^j}\left(-\frac{\alpha_{t,K-1}}{B_{t,K-1}}\sum_{b\in\mathcal{B}_{t,K-1}^j}\mathrm{grad} f_j(x_{t,K-1}^j;\xi_{t,K-1,b}^j)\right)$$

$$\cdots$$

$$x_{t,1}^j = \mathrm{Exp}_{x_t}\left(-\frac{\alpha_{t,0}}{B_{t,0}}\sum_{b\in\mathcal{B}_{t,0}^j}\mathrm{grad} f_j(x_{t,0}^j;\xi_{t,0,b}^j)\right)$$

Exp and Exp^{-1} are short of linearity!

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^{j} := x_t - x_{t,K}^{j} = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^{j}} \nabla f_j(x_{t,k}^{j}; \xi_{t,k,b}^{j}).$$

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^{j} := x_t - x_{t,K}^{j} = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^{j}} \nabla f_j(x_{t,k}^{j}; \xi_{t,k,b}^{j}).$$

Then one has

$$x_{t+1} = x_t - p_t, \text{ with } p_t = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \Delta_{t,K}^{j}.$$

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^{j} := x_t - x_{t,K}^{j} = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^{j}} \nabla f_j(x_{t,k}^{j}; \xi_{t,k,b}^{j}).$$

Then one has

$$x_{t+1} = x_t - p_t, \text{ with } p_t = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \Delta_{t,K}^{j}.$$

In the Euclidean setting:

- agent j sends $\Delta_{t,k}^{j}$ to the server
- the server averages these $\Delta_{t,K}^{j}$
- the server generates a new global parameter x_{t+1}

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^{j} := x_t - x_{t,K}^{j} = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^{j}} \nabla f_j(x_{t,k}^{j}; \xi_{t,k,b}^{j}).$$

Then one has

$$x_{t+1} = x_t - p_t, \text{ with } p_t = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \Delta_{t,K}^{j}.$$

In the Euclidean setting:

- agent j sends $\Delta_{t,k}^{j}$ to the server
- the server averages these $\Delta_{t,K}^{j}$
- the server generates a new global parameter x_{t+1}

In existing works, sending $\Delta_{t,K}^{j}$ is to use acceleration technique in the server aggregation.

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^j := x_t - x_{t,K}^j = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^j} \nabla f_j(x_{t,k}^j; \xi_{t,k,b}^j).$$

Then one has

$$x_{t+1} = x_t - p_t, \text{ with } p_t = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \Delta_{t,K}^j.$$

In the Euclidean setting:

- agent j sends $\Delta_{t,k}^j$ to the server
- the server averages these $\Delta_{t,K}^j$
- the server generates a new global parameter x_{t+1}

In the Riemannian setting, we proposed a similar aggregation

- agent j sends the "$\Delta_{t,K}^j$" to the server;
- the server averages these "$\Delta_{t,K}^j$";
- the server retracts the average into the manifold;

Back to the Euclidean aggregation, note that

$$\Delta_{t,K}^j := x_t - x_{t,K}^j = \sum_{k=0}^{K-1} \frac{\alpha_{t,k}}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^j} \nabla f_j(x_{t,k}^j; \xi_{t,k,b}^j).$$

Then one has

$$x_{t+1} = x_t - p_t, \text{ with } p_t = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \Delta_{t,K}^j.$$

In the Euclidean setting:

- agent j sends $\Delta_{t,k}^j$ to the server
- the server averages these $\Delta_{t,K}^j$
- the server generates a new global parameter x_{t+1}

In the Riemannian setting, we proposed a similar aggregation

- agent j sends the "$\Delta_{t,K}^j$" to the server;
- the server averages these "$\Delta_{t,K}^j$";
- the server retracts the average into the manifold;

What is "$\Delta_{t,K}^j$" in the Riemannian manifold?

Construct the "$\Delta_{t,K}^j$", which is dented by $\zeta_{t,K}^j$ in the Riemannian setting:

- The local mini-batch gradients, $\frac{1}{B_{t,0}} \sum_{b \in \mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j), \ldots, \frac{1}{B_{t,K-1}} \sum_{b \in \mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j)$ are inside different tangent spaces.

Construct the "$\Delta_{t,K}^j$", which is dented by $\zeta_{t,K}^j$ in the Riemannian setting:

- The local mini-batch gradients,
  $\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j), \ldots, \frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j)$
  are inside different tangent spaces.
- Transport the local mini-batch gradients to the tangent space $\mathrm{T}_{x_t}\mathcal{M}$, i.e.,
  $\mathcal{T}_{\tilde{\eta}_{t,0}^j}\left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j)\right), \ldots, \mathcal{T}_{\tilde{\eta}_{t,K-1}^j}\left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j\right),$

Construct the "$\Delta_{t,K}^j$", which is dented by $\zeta_{t,K}^j$ in the Riemannian setting:

- The local mini-batch gradients,
  $\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j), \ldots, \frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j)$
  are inside different tangent spaces.
- Transport the local mini-batch gradients to the tangent space $\mathrm{T}_{x_t}\mathcal{M}$, i.e.,
  $\mathcal{T}_{\tilde{\eta}_{t,0}^j}\left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j)\right), \ldots, \mathcal{T}_{\tilde{\eta}_{t,K-1}^j}\left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j\right),$
- Add these transported together to get to $\zeta_{t,K}^j$:
  $\zeta_{t,K}^j = \sum_{k=0}^{K-1} \alpha_{t,k} \mathcal{T}_{\tilde{\eta}_{t,k}^j}\left(\frac{1}{B_{t,k}} \sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)\right);$

Construct the "$\Delta_{t,K}^j$", which is dented by $\zeta_{t,K}^j$ in the Riemannian setting:

- The local mini-batch gradients, $\frac{1}{B_{t,0}}\sum_{b\in\mathcal{B}_{t,0}^j}\operatorname{grad} f_j(x_{t,0}^j;\xi_{t,0,b}^j),\ldots,\frac{1}{B_{t,K-1}}\sum_{b\in\mathcal{B}_{t,K-1}^j}\operatorname{grad} f_j(x_{t,K-1}^j;\xi_{t,K-1,b}^j)$ are inside different tangent spaces.
- Transport the local mini-batch gradients to the tangent space $\mathrm{T}_{x_t}\mathcal{M}$, i.e., $\mathcal{T}_{\tilde{\eta}_{t,0}^j}\left(\frac{1}{B_{t,0}}\sum_{b\in\mathcal{B}_{t,0}^j}\operatorname{grad} f_j(x_{t,0}^j;\xi_{t,0,b}^j)\right),\ldots,\mathcal{T}_{\tilde{\eta}_{t,K-1}^j}\left(\frac{1}{B_{t,K-1}}\sum_{b\in\mathcal{B}_{t,K-1}^j}\operatorname{grad} f_j(x_{t,K-1}^j;\xi_{t,K-1,b}^j\right),$
- Add these transported together to get to $\zeta_{t,K}^j$: $\zeta_{t,K}^j=\sum_{k=0}^{K-1}\alpha_{t,k}\mathcal{T}_{\tilde{\eta}_{t,k}^j}\left(\frac{1}{B_{t,k}}\sum_{b\in\mathcal{B}_{t,k}^j}\operatorname{grad} f_j(x_{t,k}^j;\xi_{t,k,b}^j)\right);$

The proposed server aggregation is given by

$$
\begin{aligned}
x_{t+1} &= \mathrm{R}_{x_t}\left(-\frac{1}{S}\sum_{j\in\mathcal{S}_t}\zeta_{t,K}^j\right)\\
&= \mathrm{R}_{x_t}\left(-\frac{1}{S}\sum_{j\in\mathcal{S}_t}\sum_{k=0}^{K-1}\alpha_{t,k}\mathcal{T}_{\tilde{\eta}_{t,k}^j}\left(\frac{1}{B_{t,k}}\sum_{b\in\mathcal{B}_{t,k}^j}\operatorname{grad} f_j(x_{t,k}^j;\xi_{t,k,b}^j)\right)\right). \qquad \text{(AGS-RS)}
\end{aligned}
$$

Construct the "$\Delta_{t,K}^j$", which is dented by $\zeta_{t,K}^j$ in the Riemannian setting:

- The local mini-batch gradients,
  $\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j), \ldots, \frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j)$
  are inside different tangent spaces.
- Transport the local mini-batch gradients to the tangent space $\mathrm{T}_{x_t}\mathcal{M}$, i.e.,
  $\mathcal{T}_{\tilde{\eta}_{t,0}^j}\left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j)\right), \ldots, \mathcal{T}_{\tilde{\eta}_{t,K-1}^j}\left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j\right),$
- Add these transported together to get to $\zeta_{t,K}^j$:
  $\zeta_{t,K}^j = \sum_{k=0}^{K-1} \alpha_{t,k} \mathcal{T}_{\tilde{\eta}_{t,k}^j}\left(\frac{1}{B_{t,k}} \sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)\right);$

The proposed server aggregation is given by

$$x_{t+1} = \mathrm{R}_{x_t}\left(-\frac{1}{S}\sum_{j\in\mathcal{S}_t} \zeta_{t,K}^j\right)$$

$$= \mathrm{R}_{x_t}\left(-\frac{1}{S}\sum_{j\in\mathcal{S}_t}\sum_{k=0}^{K-1} \alpha_{t,k}\mathcal{T}_{\tilde{\eta}_{t,k}^j}\left(\frac{1}{B_{t,k}}\sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)\right)\right). \quad \text{(AGS-RS)}$$

The proposed aggregation is another generalization of the Euclidean aggregation.

Motivation

- The number of agents in FL system is large

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown.

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown. $\Longrightarrow$ Random Sampling is impossible

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown. $\Longrightarrow$ Random Sampling is impossible

Arbitrary partial participation!

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown. $\Longrightarrow$ Random Sampling is impossible

Arbitrary partial participation!

Assumption 2.1

Assume that each agent i independently participates in any round of communication with probability $p_i > 0$.

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown. $\Longrightarrow$ Random Sampling is impossible

Arbitrary partial participation!

Assumption 2.1

Assume that each agent i independently participates in any round of communication with probability $p_i > 0$.

Let $\mathcal{A}_i = \{\text{agent } i \text{ participates in communication at the } t\text{-th round}\}$. Then, Assumption 2.1 means $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$.

Motivation

- The number of agents in FL system is large $\Longrightarrow$ full participation is impossible.
- That agents participates in communication is unknown. $\Longrightarrow$ Random Sampling is impossible

Arbitrary partial participation!

Assumption 2.1

Assume that each agent i independently participates in any round of communication with probability $p_i > 0$.

Let $\mathcal{A}_i = \{\text{agent } i \text{ participates in communication at the } t\text{-th round}\}$. Then, Assumption 2.1 means $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$.

Assumption 2.1 covers multiple cases:

- **Case 1:** Full participation with $p_i \equiv 1$;
- **Case 2:** Uniform Sampling with $p_i = \frac{S}{N}$.

2.2 Arbitrary partial participation

Participation errors are introduced by Assumption 2.1!

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad} f_j(x)\right] = \sum_{i=1}^{N}\tilde{p}_i \mathrm{grad} f_i(x)$, *with* $\tilde{p}_i = p_i\int_0^1\prod_{j\neq i}^{N}(1-p_j+p_j t)\mathrm{d}t$.

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad}f_j(x)\right] = \sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i(x)$, *with* $\tilde{p}_i = p_i\int_0^1\prod_{j\neq i}^{N}(1-p_j+p_jt)\mathrm{d}t$.

$p_i \neq p_j$ for some $i \neq j$ $\Longrightarrow$ $\tilde{p}_i \neq \tilde{p}_j$ $\Longrightarrow$ $\nexists\ \chi > 0$ s.t. $\sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i = \chi\mathrm{grad}F$

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad}f_j(x)\right] = \sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i(x)$, *with* $\tilde{p}_i = p_i\int_0^1\prod_{j\neq i}^{N}(1-p_j+p_jt)\mathrm{d}t$.

$p_i \neq p_j$ for some $i \neq j$ $\Longrightarrow$ $\tilde{p}_i \neq \tilde{p}_j$ $\Longrightarrow$ $\nexists\ \chi > 0$ s.t. $\sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i = \chi\mathrm{grad}F$

Using the aggregation via random sampling $x_{t+1} = \mathrm{R}_{x_t}(-\frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\zeta_{t,K}^i)$ may lead to the algorithm solving another problem $\min_{x\in\mathcal{M}}\tilde{F}(x) := \sum_{i=1}^{N}\tilde{p}_i f_i(x)$ different from the original problem.

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad}f_j(x)\right] = \sum_{i=1}^N \tilde{p}_i \mathrm{grad}f_i(x)$, *with* $\tilde{p}_i = p_i\int_0^1\prod_{j\neq i}^N(1-p_j+p_jt)\mathrm{d}t$.

$p_i \neq p_j$ for some $i \neq j$ $\Longrightarrow$ $\tilde{p}_i \neq \tilde{p}_j$ $\Longrightarrow$ $\nexists\, \chi > 0$ s.t. $\sum_{i=1}^N \tilde{p}_i \mathrm{grad}f_i = \chi \mathrm{grad}F$

Using the aggregation via random sampling $x_{t+1} = \mathrm{R}_{x_t}(-\frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\zeta_{t,K}^i)$ may lead to the algorithm solving another problem $\min_{x\in\mathcal{M}}\tilde{F}(x) := \sum_{i=1}^N \tilde{p}_i f_i(x)$ different from the original problem.

Some adaptations!

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad}f_j(x)\right] = \sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i(x)$, *with* $\tilde{p}_i = p_i\int_0^1\prod_{j\neq i}^{N}(1-p_j+p_jt)\mathrm{d}t$.

Back again to Assumption 2.1, at the t-th round of communication, note that

$$\mathbb{E}\left[\sum_{i\in\mathcal{S}_t}\frac{1}{p_iN}\mathrm{grad}f_i(x)\right] = \mathbb{E}\left[\sum_{i=1}^{N}\frac{1}{p_iN}\mathbb{I}_{\mathcal{S}_t}(i)\mathrm{grad}f_i(x)\right] = \sum_{i=1}^{N}\frac{1}{p_iN}\mathbb{E}\left[\mathbb{I}_{\mathcal{S}_t}(i)\mathrm{grad}f_i(x)\right]$$
$$= \sum_{i=1}^{N}\frac{1}{p_iN}\left(p_i\mathrm{grad}f_i(x)\right) = \mathrm{grad}F(x), \tag{2.1}$$

where $\mathbb{I}_{\mathcal{S}_t}(i) = 1$ if $i \in \mathcal{S}_t$ otherwise $\mathbb{I}_{\mathcal{S}_t}(i) = 0$.

Participation errors are introduced by Assumption 2.1!

Theorem 1

Under Assumption 2.1, let $\mathcal{S}_t$ denotes the set of agents who respond to the server at the t-th round of communication. Then,
$\mathbb{E}\left[\frac{1}{|\mathcal{S}_t|}\sum_{j\in\mathcal{S}_t}\mathrm{grad}f_j(x)\right]=\sum_{i=1}^{N}\tilde{p}_i\mathrm{grad}f_i(x)$, *with* $\tilde{p}_i=p_i\int_0^1\prod_{j\neq i}^{N}(1-p_j+p_jt)\mathrm{d}t$.

Back again to Assumption 2.1, at the t-th round of communication, note that

$$\mathbb{E}\left[\sum_{i\in\mathcal{S}_t}\frac{1}{p_iN}\mathrm{grad}f_i(x)\right]=\mathbb{E}\left[\sum_{i=1}^{N}\frac{1}{p_iN}\mathbb{I}_{\mathcal{S}_t}(i)\mathrm{grad}f_i(x)\right]=\sum_{i=1}^{N}\frac{1}{p_iN}\mathbb{E}\left[\mathbb{I}_{\mathcal{S}_t}(i)\mathrm{grad}f_i(x)\right]$$
$$=\sum_{i=1}^{N}\frac{1}{p_iN}\left(p_i\mathrm{grad}f_i(x)\right)=\mathrm{grad}F(x), \tag{2.1}$$

where $\mathbb{I}_{\mathcal{S}_t}(i)=1$ if $i\in\mathcal{S}_t$ otherwise $\mathbb{I}_{\mathcal{S}_t}(i)=0$.
A feasible aggregation can be chosen as

$$x_{t+1}=\mathrm{R}_{x_t}\left(-\varpi\sum_{i\in\mathcal{S}_t}\frac{1}{p_iN}\zeta_{t,K}^{i}\right)\ \text{with}\ \varpi>0\ \text{the global step size}\qquad\text{(AGS-AP)}$$

2.3 Riemannian Federated Learning via Averaging Gradient Streams with Partial Participation

Algorithm: Riemannian Federated Averaging Gradient Stream

for $t = 1, 2, \ldots, T$ **do**
  The server broadcasts x_t to all agents, i.e., $x_{t,0}^j \leftarrow x_t$ for all $j \in [N]$;
  for agent $j \in [N]$ in parallel **do**
    Set $\zeta_{t,0}^j \leftarrow 0_{x_t}$;
    for $k = 0, 1, \ldots, K-1$ **do**
      Agent j randomly samples an i.i.d. mini-batch $\mathcal{B}_{t,k}^j$ of size B_t;
      Set $\eta_{t,k}^j \leftarrow \frac{1}{B_t} \sum_{b \in \mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)$;
      Set $x_{t,k+1}^j \leftarrow \mathrm{R}_{x_{t,k}^j}(-\alpha_t \eta_k^j)$;
      Set $\zeta_{t,k+1}^j \leftarrow \zeta_{t,k}^j + \mathcal{T}_{\tilde{\eta}_{t,k}^j}(\alpha_t \eta_{t,k}^j)$ with $\tilde{\eta}_{t,k}^j$ satisfying $\mathrm{R}_{x_{t,k}^j}(\tilde{\eta}_{t,k}^j) = x_t$;
    end for
    Upload $\zeta_{t,K}^j$ to the server;
  end for
  The server computes the approximate probability $q_t^j, \forall j \in \mathcal{S}_t$;
  The server updates the new global model x_{t+1} by (AGS-AP) with q_t^j replacing p_j;
end for

2.3 Riemannian Federated Learning via Averaging Gradient Streams with Partial Participation

Algorithm: Riemannian Federated Averaging Gradient Stream

01. **for** $t = 1, 2, \ldots, T$ **do**
02. The server broadcasts x_t to all agents, i.e., $x_{t,0}^j \leftarrow x_t$ for all $j \in [N]$;
03. **for** agent $j \in [N]$ in parallel **do**
04. Set $\zeta_{t,0}^j \leftarrow 0_{x_t}$;
05. **for** $k = 0, 1, \ldots, K-1$ **do**
06. Agent j randomly samples an i.i.d. mini-batch $\mathcal{B}_{t,k}^j$ of size B_t;
07. Set $\eta_{t,k}^j \leftarrow \frac{1}{B_t} \sum_{b \in \mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)$;
08. Set $x_{t,k+1}^j \leftarrow \mathrm{R}_{x_{t,k}^j}(-\alpha_t \eta_k^j)$;
09. Set $\zeta_{t,k+1}^j \leftarrow \zeta_{t,k}^j + \mathcal{T}_{\tilde{\eta}_{t,k}^j}(\alpha_t \eta_{t,k}^j)$ with $\tilde{\eta}_{t,k}^j$ satisfying $\mathrm{R}_{x_{t,k}^j}(\tilde{\eta}_{t,k}^j) = x_t$;
10. **end for**
11. Upload $\zeta_{t,K}^j$ to the server;
12. **end for**
13. The server computes the approximate probability q_t^j, $\forall j \in \mathcal{S}_t$;
14. The server updates the new global model x_{t+1} by (AGS-AP) with q_t^j replacing p_j;
15. **end for**

competent for unknown p_i

2.3 Riemannian Federated Learning via Averaging Gradient Streams with Partial Participation

Algorithm: Riemannian Federated Averaging Gradient Stream

for $t = 1, 2, \ldots, T$ **do**
  The server broadcasts x_t to all agents, i.e., $x_{t,0}^j \leftarrow x_t$ for all $j \in [N]$;
  for agent $j \in [N]$ in parallel **do**
    Set $\zeta_{t,0}^j \leftarrow 0_{x_t}$;
    for $k = 0, 1, \ldots, K-1$ **do**
      Agent j randomly samples an i.i.d. mini-batch $\mathcal{B}_{t,k}^j$ of size B_t;
      Set $\eta_{t,k}^j \leftarrow \frac{1}{B_t} \sum_{b \in \mathcal{B}_{t,k}^j} \operatorname{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j)$;
      Set $x_{t,k+1}^j \leftarrow \mathrm{R}_{x_{t,k}^j}(-\alpha_t \eta_k^j)$;
      Set $\zeta_{t,k+1}^j \leftarrow \zeta_{t,k}^j + \mathcal{T}_{\tilde{\eta}_{t,k}^j}(\alpha_t \eta_{t,k}^j)$ with $\tilde{\eta}_{t,k}^j$ satisfying $\mathrm{R}_{x_{t,k}^j}(\tilde{\eta}_{t,k}^j) = x_t$;
    end for
    Upload $\zeta_{t,K}^j$ to the server;
  end for
  The server computes the approximate probability q_t^j, $\forall j \in \mathcal{S}_t$;
  The server updates the new global model x_{t+1} by (AGS-AP) with q_t^j replacing p_j;
end for

competent for unknown p_i

- The communication cost remains unchanged.
- The computational cost of the server remains unchanged.
- $K-1$ times more transport calculations on the agent.
- The algorithm works for general manifolds.

Outline

Assumption 3.1

The retraction R *is such that its restriction to* $\mathrm{T}_x\mathcal{M}$ *for all* $x \in \mathcal{M}$, R_x, *is of class* C^2, *and the associated vector transport* $\mathcal{T}$ *is continuous and bounded in the sense that there exists a constant* $\Upsilon > 0$ *such that for any* $x \in \mathcal{M}$, $\zeta_x, \eta_x \in \mathrm{T}_x\mathcal{M}$, *it holds that* $\|\mathcal{T}_{\eta_x}(\zeta_x)\| \leq \Upsilon\|\zeta_x\|$.

Assumption 3.2

For a sequence of the outer iterates $\{x_t\}_{t\geq 1}$ *and a sequence of the inner iterates* $\{\{\{x_{t,k}^j\}_{j=1}^N\}_{k=0}^{K-1}\}_{t\geq 1}$ *generated by Algorithm 1, there exists a* W*-totally retractive set* $\mathcal{W} \subset \mathcal{M}$ *such that* $\{x_t\}_{t\geq 1} \subset \mathcal{W}$ *and* $\{\{\{x_{t,k}^j\}_{j=1}^N\}_{k=0}^{K-1}\}_{t\geq 1} \subset \mathcal{W}$. *The minimizers of Problem (1.1) are inside* $\mathcal{W}$. *Additionally, there exists a compact and connected set* $\mathcal{X} \subset \mathcal{M}$ *such that* $\mathcal{W} \subset \mathcal{X}$.

Assumption 3.3

The cost function F is continuously differentiable in $\mathcal{W}$, the local cost functions $f_1, \ldots, f_N$ are continuously differentiable in $\mathcal{W}$, and their components $f_j(\cdot, \xi)$ for $\xi \sim \mathcal{D}_j$ with $j \in [N]$ are continuously differentiable in $\mathcal{W}$.

Assumption 3.4

The local objective functions f_j, $j \in [N]$, are L_f-Lipschitz continuously differentiable in $\mathcal{W}$ with the retraction R and the vector transport $\mathcal{T}$, implying that F is also L_f-Lipschitz continuously differentiable:

$$\|\mathcal{T}_\eta(\mathrm{grad} f_i(x)) - \mathrm{grad} f_i(y)\| \leq L_f \|\eta\| \text{ with } y = \mathrm{R}_x(\eta)$$

Assumption 3.5

F is L_g-retraction smooth over $\mathcal{W}$ with respect to R:

$$F(y) \leq F(x) + \langle \mathrm{grad} F(x), \eta \rangle + \frac{L_g}{2} \|\eta\|^2 \text{ with } y = \mathrm{R}_x(\eta)$$

Assumption 3.6

For any parameter $x \in \mathcal{M}$, the Riemannian stochastic gradient $\mathrm{grad} f_j(x;\xi^j)$ is an unbiased estimator of the gradient $\mathrm{grad} f_j(x)$, i.e., $\mathbb{E}_{\xi^j}[\mathrm{grad} f_j(x;\xi^j)] = \mathrm{grad} f_j(x), \forall j \in [N]$.

Assumption 3.7

For any fixed parameter $x \in \mathcal{M}$, there exists a positive constant σ_L such that for all $j \in [N]$, it holds that $\mathbb{E}[\| \frac{1}{B} \sum_{b\in\mathcal{B}^j} \mathrm{grad} f_j(x;\xi_b^j) - \mathrm{grad} f_j(x)\|^2] \leq \frac{\sigma_L^2}{B}$.

Reasonability: all Assumptions 3.1-3.7 have been used in existing Riemannian optimization or federated learning algorithms.

Assumption 3.8

There exist constants $q_{\min}, q_{\max} \in (0, 1]$ *and* $G \geq 0$ *independent of* $t \geq 1$ *and* $i \in [N]$, *such that the approximate probabilities* q_t^i*'s satisfy* $\left| \frac{1}{q_t^i} - \frac{1}{p_i} \right| \leq \sqrt{G\alpha_t}$, *and* $q_{\min} \leq q_t^i \leq q_{\max}, \forall t \geq 1, i \in [N]$, *where* α_t *is the local step size in the* t*-th round of communication.*

The constant G controls the accuracy of the approximate probabilities.
If the true probabilities are available to the server, G can take zero.

Remark 3.1

In [5], the authors imposed the following bound on the approximate probabilities: $\sum_{i=1}^{N} p_i^2 \left(\frac{1}{q_t^i} - \frac{1}{p_i} \right)^2 \leq \frac{N}{81}$. *This bound essentially requires that* $\left| \frac{1}{q_t^i} - \frac{1}{p_i} \right|$ *is less than some constant, which is consistent with Assumption 3.8 in fixed step size cases. Note that this assumption is considered in [5] only for fixed step size cases, but Assumption 3.8 considers another situation where the bound varies over time* t *when decaying step sizes are used.*

By the L-retraction smoothness of F, we have

$$F(x_{t+1}) - F(x_t) \leq \left\langle \mathrm{grad} F(x_t), \mathrm{R}_{x_t}^{-1}(x_{t+1}) \right\rangle + \frac{L}{2} \|\mathrm{R}_{x_t}^{-1}(x_{t+1})\|^2.$$

Recalling (TM) and (AGS-RS), we have:

$$\mathrm{Exp}_{x_t}^{-1}(x_{t+1}) = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \mathrm{Exp}_{x_t}^{-1}(x_{t,K}^j)$$

$$\mathrm{R}_{x_t}^{-1}(x_{t+1}) = -\frac{1}{S} \sum_{j \in \mathcal{S}_t} \sum_{k=0}^{K-1} \alpha_{t,k} \mathcal{T}_{\tilde{\eta}_{t,k}^j} \left(\frac{1}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j) \right)$$

By the L-retraction smoothness of F, we have

$$F(x_{t+1}) - F(x_t) \leq \left\langle \mathrm{grad} F(x_t), \mathrm{R}_{x_t}^{-1}(x_{t+1}) \right\rangle + \frac{L}{2} \|\mathrm{R}_{x_t}^{-1}(x_{t+1})\|^2.$$

Recalling (TM) and (AGS-RS), we have:

$$\mathrm{Exp}_{x_t}^{-1}(x_{t+1}) = \frac{1}{S} \sum_{j \in \mathcal{S}_t} \mathrm{Exp}_{x_t}^{-1}(x_{t,K}^j)$$

$$\mathrm{R}_{x_t}^{-1}(x_{t+1}) = -\frac{1}{S} \sum_{j \in \mathcal{S}_t} \sum_{k=0}^{K-1} \alpha_{t,k} \mathcal{T}_{\tilde{\eta}_{t,k}^j} \left(\frac{1}{B_{t,k}} \sum_{b \in \mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j) \right)$$

More suitable for analysis

Theorem 2

Let Assumptions 3.1-3.8 hold. Suppose Algorithm 1 is run with a fixed global step size $\varpi > 0$ and a decaying local step size sequence $\{\alpha_t\}$ satisfying Conditions $\sum_{t=1}^{\infty} \alpha_t = \infty, \sum_{t=1}^{\infty} \alpha_t^2 < \infty$. Then,

$$\liminf_{t \to \infty} \mathbb{E}[\|\mathrm{grad} F(x_t)\|^2] = 0.$$

Theorem 3

Under the same conditions as Theorem 2 except that the local step size sequence $\{\alpha_t\}$ *is determined by* $\alpha_t = \frac{\alpha_0}{(\beta+t)^p}$ *with constants* $\alpha_0, \beta > 0$ *and* $p \in (1/2, 1]$ *satisfying* $\varpi\alpha_1 KL_g \leq 1$*, the weighted average norm of the squared gradients satisfy, with* $A_T = \sum_{t=1}^{T} \alpha_t$*,*

$$\frac{1}{A_T}\sum_{t=1}^{T}\alpha_t\mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq \begin{cases} \mathcal{O}(\frac{1}{\ln(\beta+T)}) & p = 1, \\ \mathcal{O}(\frac{1}{(\beta+T)^{1-p}}) & p \in (1/2, 1). \end{cases}$$

Remark 3.2

In particular, if the full agent participate in any round of communication and agents use the full local gradient in local update, i.e., $G = 0$ *and* $\sigma_L = 0$*, one can relax the step sizes to* $\alpha_t = \frac{\alpha_0}{(\beta+t)^p}$ *where* $p = 1/3 + a$ *with* $a \in (0, 2/3)$*. In this case, for large* T*, the upper bound can be improved to* $\frac{1}{A_T}\sum_{t=1}^{T}\alpha_t\mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq \mathcal{O}(\frac{1}{(\beta+T)^{2/3-a}})$*.*

Sublinearly converge to a neighborhood of a stationary point.

Theorem 4

Under Assumptions 3.1-3.8, suppose that F satisfies RPL condition, i.e., there exists a constant $\mu > 0$, such that for all $x \in \mathcal{W}$, it holds that $F(x) - F(x^*) \leq \frac{1}{2\mu}\|\mathrm{grad} F(x)\|^2$. If we run Algorithm 1 with the batch size $B_t \in [B_{\mathrm{low}}, B_{\mathrm{up}}]$ and the step sizes satisfying $\alpha_t = \frac{\beta}{\gamma+t}$ for some $\gamma > 0$ and $\beta > \frac{1}{\mu\varpi K}$ such that $\alpha_1 \varpi K L_g \leq 1$, then the iterates $\{x_t\}_{t\geq 1}$ satisfy

$$\mathbb{E}[F(x_t)] - F(x^*) \leq \frac{\nu}{\gamma + t}, \quad \text{and} \quad \mathbb{E}[\|\mathrm{grad} F(x_t)\|^2] \leq \frac{2L_g\nu}{\gamma + t}, \tag{3.1}$$

where $\nu = \max\left\{\frac{\varpi K \beta^2 Q(K, B_{\mathrm{low}}, \alpha_1, \varpi)}{\beta\mu\varpi K - 1}, (\gamma+1)\Theta(x_1)\right\}$, $\Theta(x_1) = F(x_1) - F(x^*)$.

Theorem 4

Under Assumptions 3.1-3.8, suppose that F satisfies RPL condition, i.e., there exists a constant $\mu > 0$, such that for all $x \in \mathcal{W}$, it holds that $F(x) - F(x^*) \leq \frac{1}{2\mu}\|\mathrm{grad}F(x)\|^2$. If we run Algorithm 1 with the batch size $B_t \in [B_{\mathrm{low}}, B_{\mathrm{up}}]$ and the step sizes satisfying $\alpha_t = \frac{\beta}{\gamma+t}$ for some $\gamma > 0$ and $\beta > \frac{1}{\mu\varpi K}$ such that $\alpha_1\varpi K L_g \leq 1$, then the iterates $\{x_t\}_{t\geq 1}$ satisfy

$$\mathbb{E}[F(x_t)] - F(x^*) \leq \frac{\nu}{\gamma + t}, \quad \textit{and} \quad \mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq \frac{2L_g\nu}{\gamma + t}, \tag{3.1}$$

where $\nu = \max\left\{\frac{\varpi K\beta^2 Q(K, B_{\mathrm{low}}, \alpha_1, \varpi)}{\beta\mu\varpi K - 1}, (\gamma+1)\Theta(x_1)\right\}$, $\Theta(x_1) = F(x_1) - F(x^*)$.

That is, Algorithm 1 converges sublinearly to the minimizer in expectation. Convergence rate is improved from $\mathcal{O}\left(\frac{1}{\ln(t)}\right)$ to $\mathcal{O}\left(\frac{1}{t}\right)$.

Theorem 5

Suppose that Assumptions 3.1-3.8 hold. We run Algorithm 1 with a fixed global step size ϖ, a fixed batch size B, and a fixed number of local updates K.

1. *If the fixed step sizes α and ϖ satisfy $\alpha\varpi KL_g \leq 1$, then*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq \frac{2\Theta(x_1)}{\varpi\alpha KT} + 2\alpha Q(K,B,\alpha,\varpi). \tag{3.2}$$

Sublinearly converge to a neighborhood of a stationary point.

Theorem 5

Suppose that Assumptions 3.1-3.8 hold. We run Algorithm 1 with a fixed global step size ϖ, a fixed batch size B, and a fixed number of local updates K.

2. *If local full gradient descent step is performed in local updates, i.e., $\sigma_L = 0$, and one takes a local fixed step size $\alpha > 0$ such that*

$$\alpha = \sqrt{\frac{\Theta(x_1)}{2\varpi P^2(G\delta_2^2+\Upsilon^2\delta_4^2 KL_g\varpi)KT}} \text{ with } T \text{ satisfying}$$

$$T \geq \max\left\{\frac{\varpi KL_g^2\Theta(x_1)}{2P^2(G\delta_2^2+KL_g\varpi\Upsilon^2\delta_4^2)}, \frac{\Theta(x_1)(2K-1)^2(K-1)^2L_f^4\delta_1^4(\varpi^2L_g^2J^2K^2+P^2H^2)^2}{72P^2L_g^4K^4\varpi^5(G\delta_2^2+KL_g\varpi\Upsilon^2\delta_4^2)^3}\right\}, \text{ then}$$

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq 4P\sqrt{2\Theta(x_1)\left(\frac{G\delta_2^2}{\varpi KT}+\frac{L_g\Upsilon^2\delta_4^2}{T}\right)}.$$

Sublinearly converge to a neighborhood of a stationary point.

Theorem 5

Suppose that Assumptions 3.1-3.8 hold. We run Algorithm 1 with a fixed global step size ϖ, a fixed batch size B, and a fixed number of local updates K.

3. *If the true probabilities are known, meaning $G = 0$, and one takes local and global step sizes α and ϖ such that $\alpha\varpi = \sqrt{\frac{\Theta(x_1)B}{(\delta_3^2\sigma_L^2+2P^2\delta_4^2KB)\Upsilon^2 L_g KT}}$ with T satisfying*

$$T \geq \max\left\{ \frac{KL_g\Theta(x_1)B}{(\delta_3^2\sigma_L^2+2P^2\delta_4^2KB)\Upsilon^2}, \frac{\Theta(x_1)(2K-1)^2(K-1)^2L_f^4\delta_1^4P^4(L_g^2\varpi^2J^2K^2+P^2H^2)^2B^3}{9(\delta_3^2\sigma_L^2+2P^2\delta_4^2KB)^3\Upsilon^6L_g^7\varpi^6K^5} \right\},$$

then

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\|\mathrm{grad}F(x_t)\|^2] \leq 4\Upsilon\sqrt{L_g\Theta(x_1)\left(\frac{\delta_3^2\sigma_L^2}{KTB}+\frac{2P^2\delta_4^2}{T}\right)}.$$

Sublinearly converge to a neighborhood of a stationary point.

Remark 3.3

If the probabilities p_i are known, i.e., $q_t^i = p_i$, and $p_{\min} = \min_i\{p_i\}$ is not too small and not fairly far away from $p_{\max} = \max_i\{p_i\}$ then, Item 2 and Item 3 give the upper bounds as

$$\mathcal{O}\left(\frac{1}{\sqrt{\varpi KT}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NT}}\right) \text{ and } \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NKTB}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NT}}\right).$$

In particular, if the probabilities are the same across agents, e.g., $p_i = \frac{S}{N}$ with $S \leq N$, then Item 2 and Item 3 give the upper bounds as

$$\mathcal{O}\left(\frac{1}{\sqrt{\varpi KT}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{ST}}\right) \text{ and } \mathcal{O}\left(\frac{1}{\sqrt{SKTB}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{ST}}\right).$$

Remark 3.3

If the probabilities p_i are known, i.e., $q_t^i = p_i$, and $p_{\min} = \min_i\{p_i\}$ is not too small and not fairly far away from $p_{\max} = \max_i\{p_i\}$ then, Item 2 and Item 3 give the upper bounds as

$$\mathcal{O}\left(\frac{1}{\sqrt{\varpi KT}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NT}}\right) \text{ and } \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NKTB}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{p_{\min} NT}}\right).$$

In particular, if the probabilities are the same across agents, e.g., $p_i = \frac{S}{N}$ with $S \leq N$, then Item 2 and Item 3 give the upper bounds as

$$\mathcal{O}\left(\frac{1}{\sqrt{\varpi KT}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{ST}}\right) \text{ and } \mathcal{O}\left(\frac{1}{\sqrt{SKTB}}\right) + \mathcal{O}\left(\frac{1}{\sqrt{ST}}\right).$$

$\frac{1}{\sqrt{T}} \longrightarrow \frac{1}{\sqrt{ST}}, \frac{1}{\sqrt{S}}$ speed up.

Theorem 6

Under Assumptions 3.1-3.8, suppose that F satisfies RPL condition with a constant $\mu > 0$. If we run Algorithm 1 with batch size $B_t \in [B_{\text{low}}, B_{\text{up}}]$ and step sizes $\alpha_t = \alpha$ and ϖ satisfying $\alpha\varpi K \leq \min\{1/L_g, 1/\mu\}$, then the resulting iterates $\{x_t\}_{t=1}^T$ satisfy

$$\mathbb{E}[F(x_T)] - F(x^*) \leq (1 - \mu\varpi K\alpha)^{T-1}\Theta(x_1) + \frac{\alpha}{\mu}Q(K, B_{\text{low}}, \alpha, \varpi)$$
$$\xrightarrow{T\to\infty} \frac{\alpha}{\mu}Q(K, B_{\text{low}}, \alpha, \varpi).$$

Linearly converge to a neighborhood of the solution.

Approximating true probabilities

- $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$
- $q_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)$ and $\lim_{t\to\infty} \mathbb{P}\{|q_t^i/t - p_i| < \epsilon\} = 1$ for any small $\epsilon > 0$

Approximating true probabilities

- $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$
- $\mathfrak{q}_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)$ and $\lim_{t\to\infty} \mathbb{P}\{|\mathfrak{q}_t^i/t - p_i| < \epsilon\} = 1$ for any small $\epsilon > 0$

Use frequencies, i.e., $q_t^i = \frac{\mathfrak{q}_t^i}{t}$

Approximating true probabilities

- $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$
- $\mathfrak{q}_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)$ and $\lim_{t\to\infty} \mathbb{P}\{|\mathfrak{q}_t^i/t - p_i| < \epsilon\} = 1$ for any small $\epsilon > 0$

Use frequencies, i.e., $q_t^i = \frac{\mathfrak{q}_t^i}{t}$

Theorem 7

Under Assumption 2.1, for each agent i, we have

$$\mathbb{P}\left\{\left|\frac{1}{q_t^i} - \frac{1}{p_i}\right| \le \mathcal{G} t^{-\frac{a}{2}}\right\} \ge 1 - \min\left\{2e^{-\frac{tp_i^2}{2}}, \frac{4(1-p_i)}{tp_i}\right\} - \min\left\{2e^{-\frac{\mathcal{G}^2 p_i^4}{2} t^{1-a}}, \frac{4(1-p_i)}{\mathcal{G}^2 p_i^3 t^{1-a}}\right\},$$

where $q_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)/t$, and $\mathcal{G} \ge 0$ and $a \ge 0$ are constants.

Approximating true probabilities

- $\mathcal{A}_i \sim \text{Bernoulli}(p_i)$
- $\mathfrak{q}_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)$ and $\lim_{t\to\infty} \mathbb{P}\{|\mathfrak{q}_t^i/t - p_i| < \epsilon\} = 1$ for any small $\epsilon > 0$

Use frequencies, i.e., $q_t^i = \frac{\mathfrak{q}_t^i}{t}$

Theorem 7

Under Assumption 2.1, for each agent i, we have

$$\mathbb{P}\left\{\left|\frac{1}{q_t^i} - \frac{1}{p_i}\right| \le \mathcal{G} t^{-\frac{a}{2}}\right\} \ge 1 - \min\left\{2e^{-\frac{tp_i^2}{2}}, \frac{4(1-p_i)}{tp_i}\right\} - \min\left\{2e^{-\frac{\mathcal{G}^2 p_i^4}{2} t^{1-a}}, \frac{4(1-p_i)}{\mathcal{G}^2 p_i^3 t^{1-a}}\right\},$$

where $q_t^i = \sum_{\tau=1}^{t} \mathbb{I}_{\mathcal{S}_\tau}(i)/t$, and $\mathcal{G} \ge 0$ and $a \ge 0$ are constants.

- step size takes the form of $\alpha_t = \mathcal{O}(t^{-a})$ with $a \in (1/2, 1] \cup \{0\}$;
- $a = 0$ fixed step size, $a \in (1/2, 1]$ decaying step size;
- Assumption 3.8 holds with probability not less than

$$1 - \min\left\{2e^{-\frac{tp_i^2}{2}}, \frac{4(1-p_i)}{tp_i}\right\} - \min\left\{2e^{-\frac{\mathcal{G}^2 p_i^4}{2} t^{1-a}}, \frac{4(1-p_i)}{\mathcal{G}^2 p_i^3 t^{1-a}}\right\}$$

Outline

- The experiments conducted on the empirical risk minimization

$$\min_{x\in\mathcal{M}} F(x) := \frac{1}{N}\sum_{i=1}^{S} f_i(x;\mathcal{D}_i) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{S}\sum_{j=1}^{S} f(x;z_{i,j}),$$

where N is the number of agents, $D_i = \{z_{i,1},\ldots,z_{i,S}\}$ is the local dataset with size of S held by agent i.

For decaying step sizes cases, the step sizes are computed by the following formulation:

$$\bar{\alpha}_t = \begin{cases} \alpha_0 & \text{if } t = 0 \\ \alpha_0/(\beta + c_t) & \text{if } t > 0, \end{cases} \quad \text{with } c_t = \begin{cases} 0 & \text{if } t = 0, \\ c_{t-1} + 1 & \text{if } \mathrm{mod}(t,\mathrm{dec}) = 0, \\ c_{t-1} & \text{otherwise}, \end{cases}$$

where α_0 is the initial step size, β is the decaying parameter, and dec is the decaying gap.

Computing principal eigenvector over sphere manifolds (CPESph)

- $\min_{x \in \mathrm{S}^d} F(x) := -\frac{1}{N} \sum_{i=1}^{N} \frac{1}{S} \sum_{i=1}^{S} x^T (z_{i,j} z_{i,j}^T) x$ with $\mathrm{S}^d = \{x \in \mathbb{R}^{d+1} : \|x\|_2 = 1\}$
- The objective locally satisfies RPL condition
- Synthetize the samples $\mathcal{D}_i = \{z_{i,1}, \ldots, z_{i,N}\}$ for all $i = 1, \ldots, N$:
  - Diagonal matrix $\Sigma_i = \mathrm{diag}\{1, 1 - 1.1\nu, \ldots, 1 - 1.4\nu, \frac{|y_1|}{(d+1)}, \frac{|y_2|}{(d+1)}, \ldots\}$ of size $(d+1) \times (d+1)$ with ν being the eigengap and $y_i \in \mathbb{R}$ being sampled from the standard Gaussian distribution
  - Set $Z_i = U_i \Sigma_i V_i$ with $U_i \in \mathbb{R}^{S \times (d+1)}$ and $V \in \mathbb{R}^{(d+1) \times (d+1)}$ being two orthonormal matrices
  - Set $z_{i,j} = Z_i(j, :)^T$

Computing Fréchet mean over SPD manifolds (CFMSPD)

- $\min_{X \in \mathbb{S}^d_{++}} F(X) := \frac{1}{N} \sum_{i=1}^{N} \frac{1}{S} \sum_{j=1}^{S} \|\mathrm{logm}(X^{-1/2} Z_{i,j} X^{-1/2})\|_F^2$ with $\mathbb{S}^d_{++} = \{X \in \mathbb{R}^{d \times d} : X \succ 0\}$
- The objective locally satisfies the RPL condition
- Synthetize the samples $\mathcal{D}_i = \{Z_{i,1}, \ldots, Z_{i,S}\} \subset \mathbb{S}^d_{++}$:
  - Each data point is sampled from the Wishart distribution $W(I_d/d, d)$ with a diameter $D_{\mathcal{W}}$

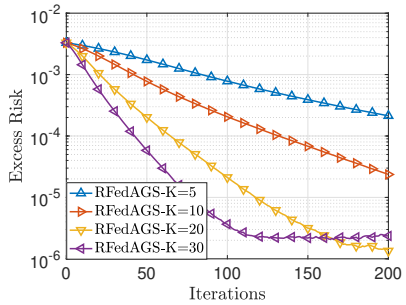
Minimization of the Brockett cost function over Stiefel manifolds (MBCFSti)

- $\min_{X \in \mathrm{St}(p,d)} F(X) = \frac{1}{S} \sum_{i=1}^{S} \sum_{j=1}^{N} \mathrm{trace}(X^T A_{i,j} X H)$ with $\mathrm{St}(p,d) = \{X \in \mathbb{R}^{d \times p} : X^T X = I_p\}$
- $H = \mathrm{diag}\{p, p-1, \ldots, 1\}$
- The objective locally satisfies the RPL condition
- Synthetize the samples $\mathcal{D}_i = \{A_{i,1}, \ldots, A_{i,N}\} \subset \mathbb{S}^d_{++}$:
  - Set $A_{i,j} = B + B^T$ with B being drawn from the standard Gaussian distribution

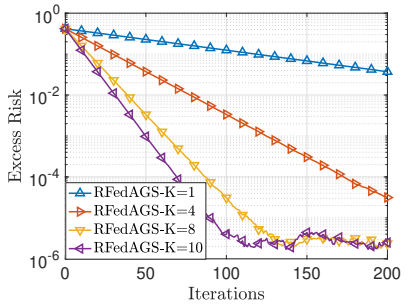
Table 1: The parameters of the three problems and RFedAGS. Notation $a.b_k$ denotes a number $a.b \times 10^k$ and the dash "$-$" means that the parameter does not exist in the problem.

Parameters	Problem-related				Algorithm-related						
Problems	d	p	ν	$D_{\mathcal{W}}$	S	N	$\bar{\alpha}$	α_0	β	dec	$\bar{B}$
CPESph	2.5_1	–	1_{-3}	–	1.0_1	8.0_1	1	1	1.0_{-1}	5.0_1	6.4_1
CFMSPD	2	–	–	1	1.0_1	6.0_1	3.0_{-3}	8.0_{-3}	1.0_{-1}	2.0_1	3.0_1
MBCFSti	2.5_1	2	–	–	2.0_1	5.0_1	3.0_{-3}	2.0_{-2}	1.0_{-1}	5.0_1	2.5_1

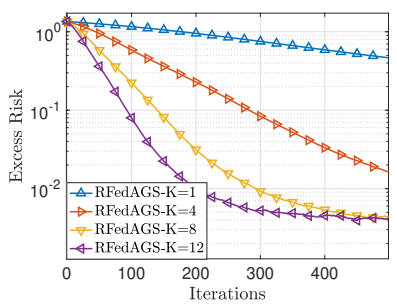
4.1 Three simulation experiments: full participation



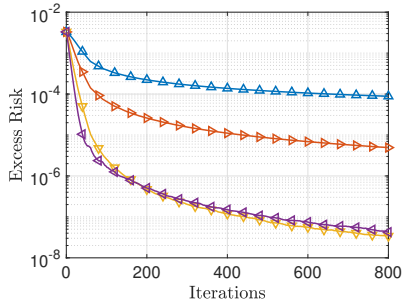
(a) CPESph, fixed



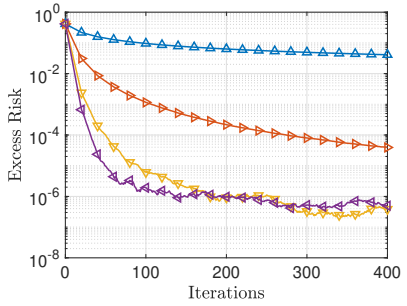
(b) CFMSPD, fixed



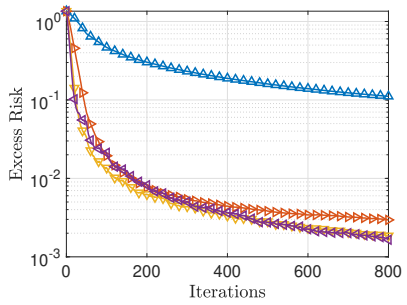
(c) MBCSti, fixed



(d) CPESph, decaying



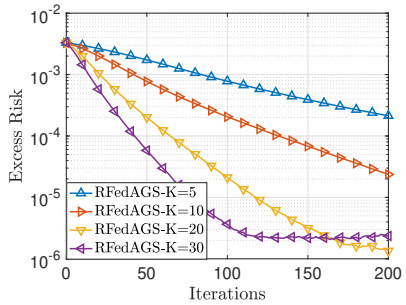
(e) CFMSPD, decaying



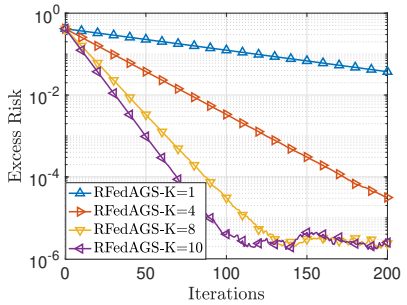
(f) MBCSti, decaying

Figure 2: The influence of the different number, K, of local updates on synthetic data. Fixed step size cases (first row) and decaying step size cases (second row).

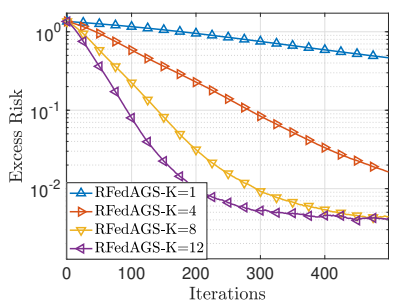
4.1 Three simulation experiments: full participation



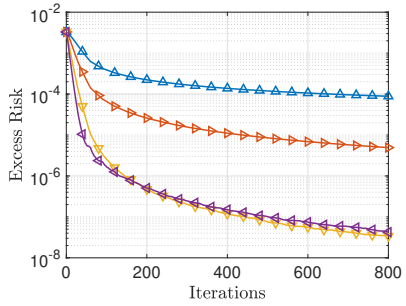
(a) CPESph, fixed



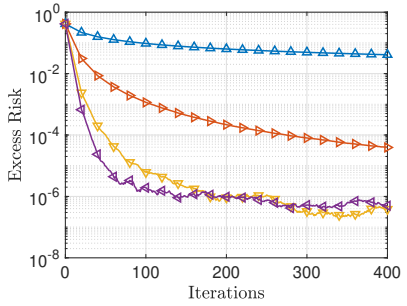
(b) CFMSPD, fixed



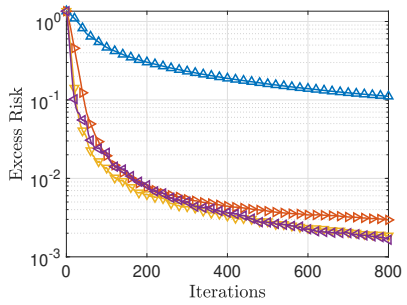
(c) MBCSti, fixed



(d) CPESph, decaying



(e) CFMSPD, decaying



(f) MBCSti, decaying

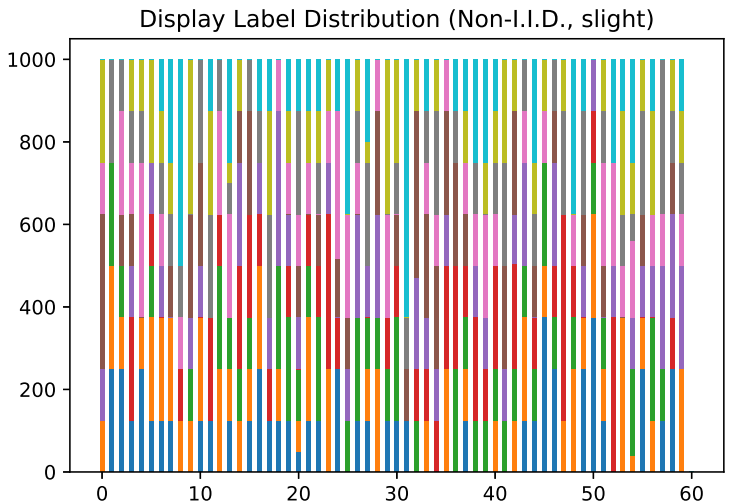
Figure 2: The influence of the different number, K, of local updates on synthetic data. Fixed step size cases (first row) and decaying step size cases (second row).

- Linear convergence if fixed step sizes
- More accurate if decaying step size
- The larger K is, the faster the algorithms converges
- Too large K may diminish the accuracy for large T

Computing principal eigenvector over sphere manifolds (CPESph)

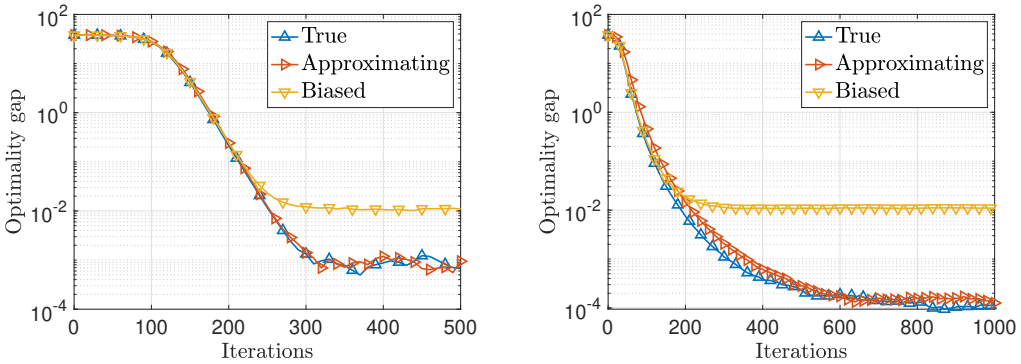
$$\min_{x\in\mathrm{S}^d} F(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x), \text{ with } f_i(x) = -\frac{1}{S}\sum_{i=1}^{S} x^T(z_{i,j}z_{i,j}^T)x$$

MNIST dataset: $(N, S) = (60, 1000)$, $d = 783$, $\alpha = 8 \times 10^{-5}$, $(\alpha_0, \beta, \mathrm{d}) = (3.5 \times 10^{-4}, 0.1, 20)$, $B = 0.5S$, $K = 5$, $p_i \sim \mathrm{U}(0, 1)$



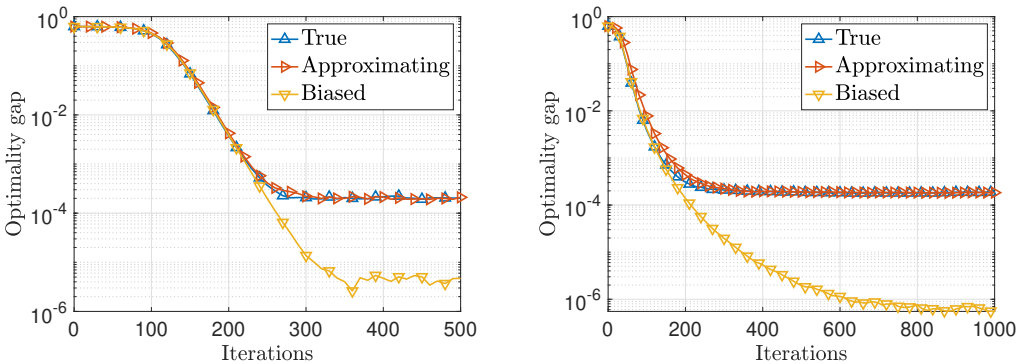
(a) Non-I.I.D. (slight)

Figure 3: Sample distributions across different agents on MNIST dataset. x-axis is the ID of each agents and y-axis is the number of local samples.



(a) Fixed step size (b) Decaying step size

Figure 4: RFedAGS with the two aggregations (AGS-RS) and (AGS-AP) solve the original problem $\arg\min_{x\in\mathcal{M}} F(x)$.



(a) Fixed step size (b) Decaying step size

Figure 5: RFedAGS with the two aggregations solve the re-weighted problem $\arg\min_{x\in\mathcal{M}} \tilde{F}(x)$.

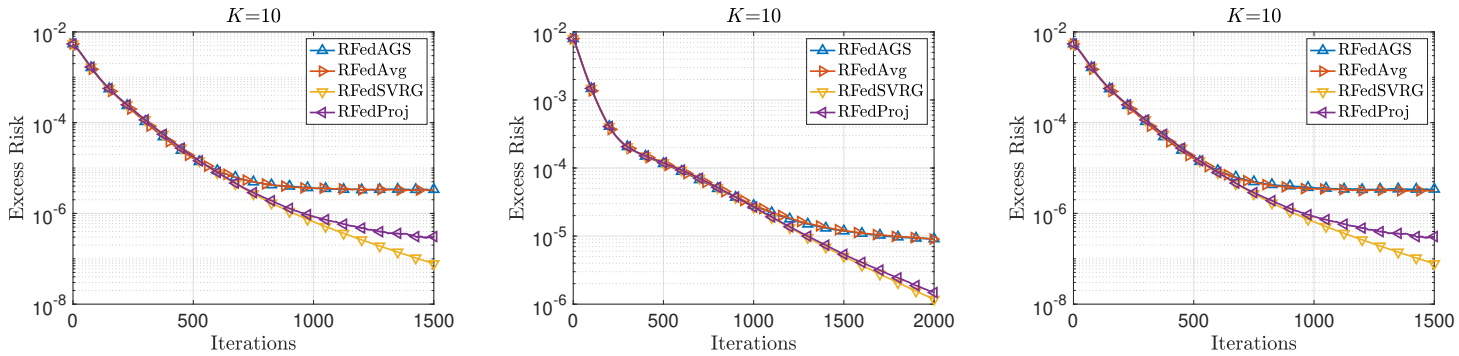
Principal component analysis (PCA) over the Stiefel manifold

$$\min_{x\in \mathrm{St}(r,d)} F(X) := \frac{1}{N}\sum_{i=1}^{N} f_i(X), \text{ with } f_i(X) = -\frac{1}{S}\sum_{j=1}^{S} \mathrm{tr}(X^T(Z_{ij}Z_{ij}^T)X)$$

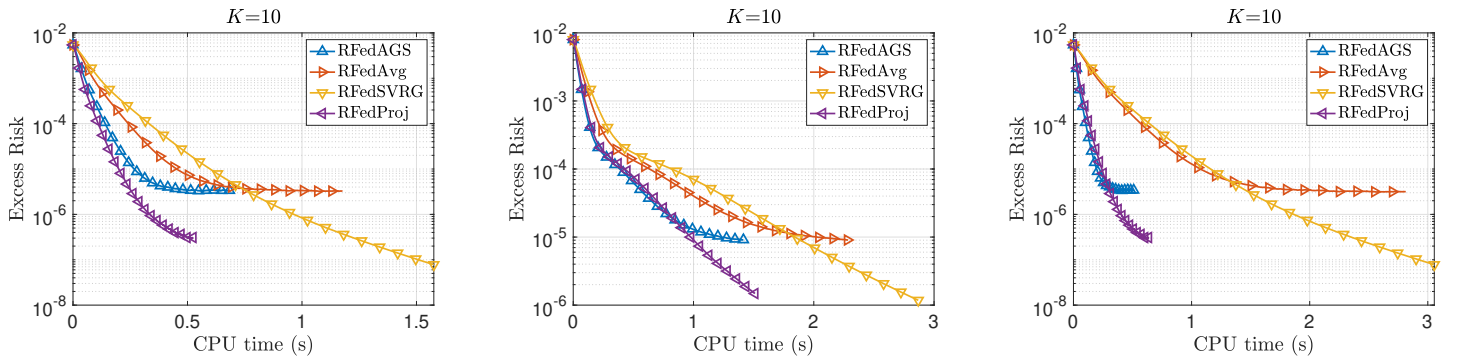
- The samples generated identically to those in CPESph
- Compared to RFedAvg [3], RFedSVRG [3], RFedProj [6]
- $(N, S) = (40, 80)$, $\alpha = 0.8$, $B = 40$, $K = 10$, and $(d, p) = (30, 5), (30, 10)$.

[2] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[6] J. Zhang and J. Hu and A. M.-C. So and M. Johansson. Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data. arxiv:2406.08465, 2024.

4.3 Principal component analysis: full participation



(a) $(d, p) = (30, 5)$ (b) $(d, p) = (30, 10)$ (c) $(d, p) = (30, 5)$



(d) $(d, p) = (30, 5)$ (e) $(d, p) = (30, 10)$ (f) $(d, p) = (30, 5)$

Figure 6: Firs twos columns: Cayley transform. Third column: QR and by projection.

- CPU time = server CPU time + max(agent's CPU time);
- Above: perform similarly initially and RFedSVRG and RFedProj find more accurate solutions;
- RFedAvg, RFedSVRG: inverse of retraction in server;
- RFedProj: orthogonal projection costs more for large p.

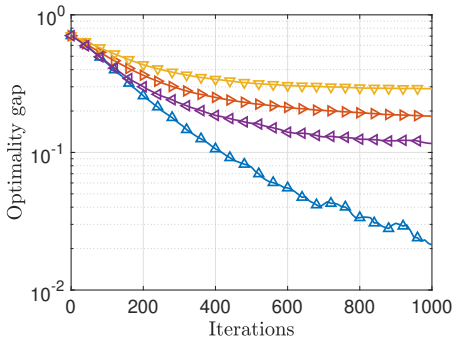
Principal component analysis (PCA) over the Stiefel manifold

$$\min_{x\in \mathrm{St}(r,d)} F(X) := \frac{1}{N}\sum_{i=1}^{N} f_i(X), \text{ with } f_i(X) = -\frac{1}{S}\sum_{j=1}^{S} \mathrm{tr}(X^T(Z_{ij}Z_{ij}^T)X)$$

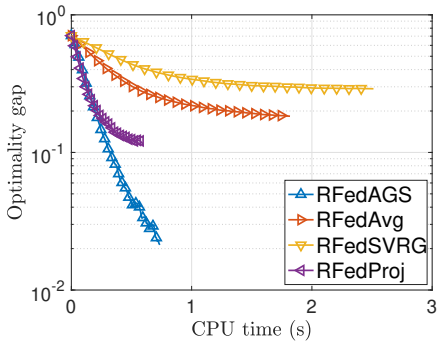
- The samples generated identically to those in CPESph
- Compared to RFedAvg [3], RFedSVRG [3], RFedProj [6]
- Synthetic data: $Z_{i,j} \sim \mathcal{N}(0, \frac{i}{N})$, $(r, d) = (5, 100)$, $(N, S) = (40, 100)$, $\alpha = 6 \times 10^{-3}$, $B = 50$, $K = 5$, $p_i \sim \mathrm{U}(0, 1)$
- CIFAR10 dataset: $(r, d) = (4, 3072)$, $(N, S) = (50, 1000)$, $\alpha = 3 \times 10^{-5}$, $B = 500$, $K = 5$, $p_i \sim \mathrm{U}(0, 1)$

[3] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[6] J. Zhang and J. Hu and A. M.-C. So and M. Johansson. Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data. NeurIPS, 2024.

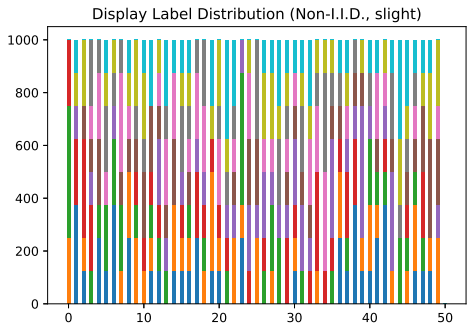
4.4 Principal component analysis: arbitrary participation



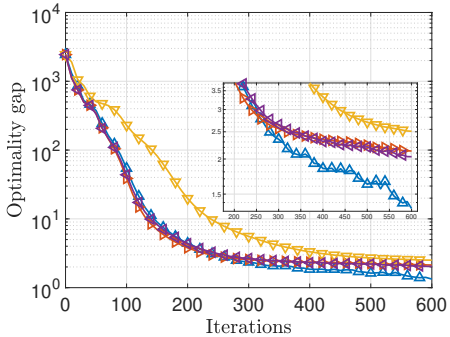
(a)



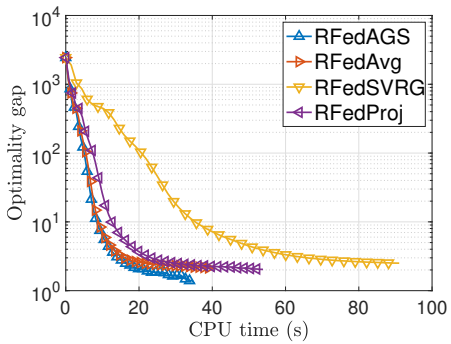
(b)



(c) Local dataset distributions



(d)



(e)

Figure 7: First row: synthetic data. Second row: CIFAR10 dataset

- RFedAGS uniformly outperforms the other algorithms.

Summary

For more details, please see

Zhenwei Huang, Wen Huang, Pratik Jawanpuria, and Bamdev Mishra. Riemannian federated learning via averaging gradient streams. arxiv.org/abs/2409.07223, 2024.

- Introduced the federated learning
- Propose a new server aggregation
- Extend the new aggregation to arbitrary participation
- Convergence analysis
- Numerical experiments

Thank you

Thank you for your attention!

References I

[1] S. Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

[2] Zhenwei Huang, Wen Huang, Pratik Jawanpuria, and Bamdev Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy, 2024.

[3] Jiaxiang Li and Shiqian Ma. Federated learning on Riemannian manifolds. *Applied Set-Valued Analysis and Optimization*, 5(2), 2023.

[4] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.

[5] Shiqiang Wang and Mingyue Ji. A lightweight method for tackling unknown participation statistics in federated averaging. In *International Conference on Learning Representations*, 2024.

[6] Jiaojiao Zhang, Jiang Hu, Anthony Man-Cho So, and Mikael Johansson. Nonconvex federated learning on compact smooth submanifolds with heterogeneous data. *Advances in Neural Information Processing Systems*, 37:109817–109844, 2024.