Riemannian Federated Learning via Averaging Gradient Stream

Speaker: Wen Huang

Xiamen University

May 17, 2025

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

Outline

1. Federated Learning Review
2. Riemannian Federated Learning Averaging Gradient Stream
3. Convergence Analysis
4. Numerical Experiments
5. Summary

Outline

1. Federated Learning Review

2. Riemannian Federated Learning Averaging Gradient Stream

3. Convergence Analysis

4. Numerical Experiments

5. Summary

General Federated Learning Optimization:

$$\min_{x\in\mathbb{R}^n} F(x) = \sum_{i=1}^{S} p_i f_i(x), \text{ with } p_i \geq 0 \text{ and } \sum_{i=1}^{S} p_i = 1, \tag{1.1}$$

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

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

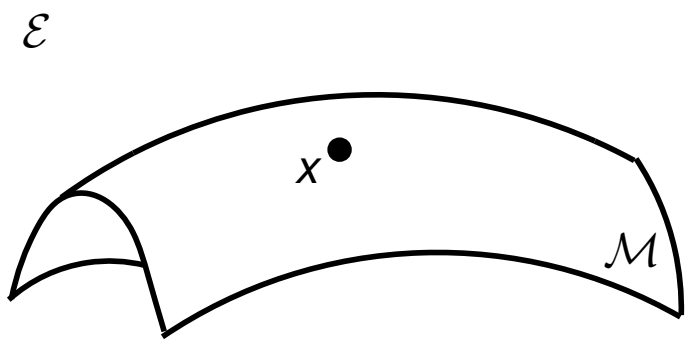
General Federated Learning Optimization:

$$\min_{x\in\mathcal{M}} F(x) = \sum_{i=1}^{S} p_i f_i(x), \text{ with } p_i \geq 0 \text{ and } \sum_{i=1}^{S} p_i = 1, \tag{1.1}$$

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

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

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



General Federated Learning Optimization:

$$\min_{x\in\mathcal{M}} F(x) = \sum_{i=1}^{S} p_i f_i(x), \text{ with } p_i \geq 0 \text{ and } \sum_{i=1}^{S} p_i = 1, \tag{1.1}$$

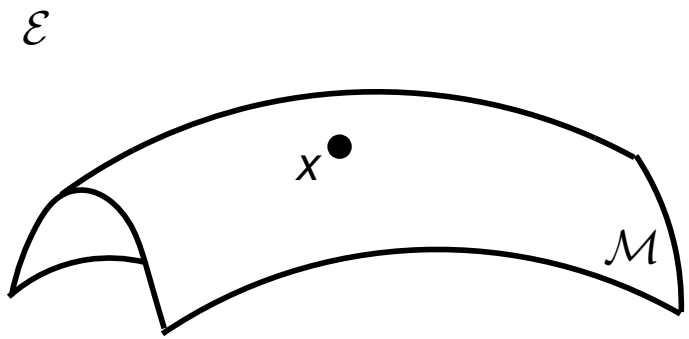
- S is the number of agents;
- f_i is the local objective of agent i, and covers

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

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

Applications:

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



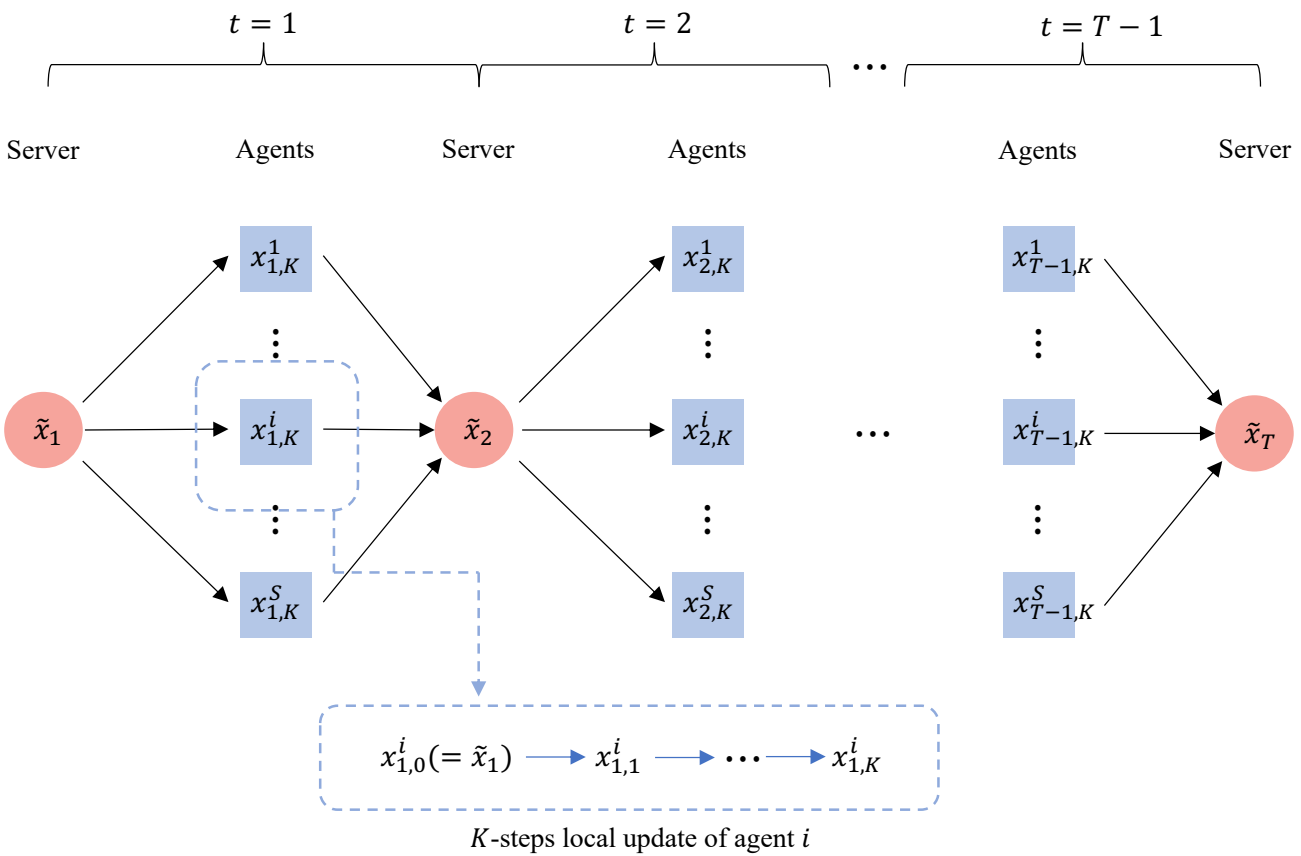


Figure 1: Flowchart of a federated learning algorithm

Euclidean version:

Algorithm: A representative federated averaging algorithm [McM+17]

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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

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

[McM+17] 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 [McM+17]

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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

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

[McM+17] 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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

- Agents: Riemannian SGD [Bon13]
- 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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

- Agents: Riemannian SGD [Bon13]
- 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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

- Agents: Riemannian SGD [Bon13]
- 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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
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 $\tilde{x}_t$ 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

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

9. **end for**

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

- Agents: Riemannian SGD [Bon13]
- Sever: Aggregation

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

Euclidean to Riemannian (Aggregation: an existing approach):

- Naive generalization:

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

Euclidean to Riemannian (Aggregation: an existing approach):

- Naive generalization:

$$\tilde{x}_{t+1} \leftarrow \sum_{j\in\mathcal{S}_t} \frac{p_j}{\sum_{j\in\mathcal{S}_t} p_j} x^j_{t,K} \not\Longrightarrow \text{Riemannian setting}$$

- An alternative approach:

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

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

Euclidean to Riemannian (Aggregation: an existing approach):

- Naive generalization:

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

- An alternative approach:

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

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

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

Euclidean to Riemannian (Aggregation: an existing approach):

- Naive generalization:

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

- An alternative approach:

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

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

- $\tilde{x}_{t+1} = \arg\min_x \sum_{j\in\mathcal{S}_j} \frac{p_j}{\sum_{j\in\mathcal{S}_t} p_j} \mathrm{dist}^2(x, x_{t,K}^j)$: computationally expensive;
- One step of Riemannian gradient descent (called tangent mean) [LM23]:

$$\tilde{x}_{t+1} \leftarrow \mathrm{Exp}_{\tilde{x}_t}\left(\sum_{j\in\mathcal{S}_t} \frac{p_j}{\sum_{i\in\mathcal{S}_t} p_i} \mathrm{Exp}_{\tilde{x}_t}^{-1}(x_{t,K}^j)\right);$$

[LM23] 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 [LM23]
  - 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.

[LM23] 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 [LM23]
  - 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 [Hua+24]
  - Use differential privacy to enhance the privacy of federated learning;
  - Use tangent mean as the server aggregation.
  - Requirements for convergence similar to that in [LM23].

[LM23] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.

[Hua+24] 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 [LM23]
  - 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 [Hua+24]
  - Use differential privacy to enhance the privacy of federated learning;
  - Use tangent mean as the server aggregation.
  - Requirements for convergence similar to that in [LM23].

- Riemannian Federated Learning on Compact Submanifolds with Heterogeneous Data [Zha+24]
  - Use projection onto the manifold
  - Allow multiple agents and multiple local updates

[LM23] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.

[Hua+24] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.

[Zha+24] 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.

Limitations

- Full agent participation and one step of local update or one agent participation and multiple of local updates [LM23; Hua+24]
- Compact submanifolds embedded in Euclidean spaces [Zha+24]

[LM23] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[Hua+24] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.
[Zha+24] 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.

Limitations

- Full agent participation and one step of local update or one agent participation and multiple of local updates [LM23; Hua+24]
- Compact submanifolds embedded in Euclidean spaces [Zha+24]

Proposed Riemannian federated learning algorithm overcomes these limitations!

[LM23] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.
[Hua+24] Z. Huang, W. Huang, P. Jawanpuria, B. Mishra. Federated Learning on Riemannian Manifolds with Differential Privacy. arxiv:2404.10029, 2024.
[Zha+24] 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.

Outline

1. Federated Learning Review

2. Riemannian Federated Learning Averaging Gradient Stream

3. Convergence Analysis

4. Numerical Experiments

5. Summary

A new server aggregation: average of gradient stream

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

$$x^j_{t,K} = \underbrace{x^j_{t,K-1} - \frac{\alpha_{t,K-1}}{B_{t,K-1}} \sum_{b\in\mathcal{B}^j_{t,K-1}} \nabla f_j(x^j_{t,K-1};\xi^j_{t,K-1,b})}_{\text{RSGD for instance}}$$

$$= x^j_{t,K-2} - \frac{\alpha_{t,K-2}}{B_{t,K-2}} \sum_{b\in\mathcal{B}^j_{t,K-2}} \nabla f_j(x^j_{t,K-2};\xi^j_{t,K-2,b}) - \frac{\alpha_{t,K-1}}{B_{t,K-1}} \sum_{b\in\mathcal{B}^j_{t,K-1}} \nabla f_j(x^j_{t,K-1};\xi^j_{t,K-1,b})$$

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

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

A new server aggregation: average of gradient stream

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

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

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

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

$$\cdots$$

$$x^j_{t,1} = \mathrm{Exp}_{\tilde{x}_t}\left(-\frac{\alpha_{t,0}}{B_{t,0}} \sum_{b\in\mathcal{B}^j_{t,0}} \mathrm{grad} f_j(x^j_{t,0}; \xi^j_{t,0,b})\right)$$

A new server aggregation: average of gradient stream

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

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

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

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

$$\cdots$$

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

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

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

Then one has

$$\tilde{x}_{t+1} = \tilde{x}_t - d_t, \text{ with } d_t = \sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \Delta^j_{t,K}.$$

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

Then one has

$$\tilde{x}_{t+1} = \tilde{x}_t - d_t, \text{ with } d_t = \sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \Delta^j_{t,K}.$$

In the Euclidean setting:

- agent j sends $\Delta^j_{t,k}$ to the server
- the server averages these $\Delta^j_{t,K}$
- the server generates a new global parameter $\tilde{x}_{t+1}$

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

Then one has

$$\tilde{x}_{t+1} = \tilde{x}_t - d_t, \text{ with } d_t = \sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \Delta^j_{t,K}.$$

In the Euclidean setting:

- agent j sends $\Delta^j_{t,k}$ to the server
- the server averages these $\Delta^j_{t,K}$
- the server generates a new global parameter $\tilde{x}_{t+1}$

In existing works [Kar+20; Red+21], sending $\Delta^j_{t,K}$ is to use acceleration technique in the server aggregation.

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

Then one has

$$\tilde{x}_{t+1} = \tilde{x}_t - d_t, \text{ with } d_t = \sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \Delta^j_{t,K}.$$

In the Euclidean setting:

- agent j sends $\Delta^j_{t,k}$ to the server
- the server averages these $\Delta^j_{t,K}$
- the server generates a new global parameter $\tilde{x}_{t+1}$

In the Riemannian setting, we proposed a similar aggregation

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

A new server aggregation: average of gradient stream

Back to the Euclidean aggregation, note that

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

Then one has

$$\tilde{x}_{t+1} = \tilde{x}_t - d_t, \text{ with } d_t = \sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \Delta^j_{t,K}.$$

In the Euclidean setting:

- agent j sends $\Delta^j_{t,k}$ to the server
- the server averages these $\Delta^j_{t,K}$
- the server generates a new global parameter $\tilde{x}_{t+1}$

In the Riemannian setting, we proposed a similar aggregation

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

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

A new server aggregation: average of gradient stream

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.

A new server aggregation: average of gradient stream

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 $\mathrm{T}_{\tilde{x}_t}\mathcal{M}$, i.e.,
  $\Gamma_{x_{t,0}^j}^{\tilde{x}_t}\left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j;\xi_{t,0,b}^j)\right), \ldots, \Gamma_{x_{t,1}^j}^{\tilde{x}_t}\left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j;\xi_{t,K-1,b}^j)\right),$

A new server aggregation: average of gradient stream

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 $\mathrm{T}_{\tilde{x}_t}\mathcal{M}$, i.e.,
  $\Gamma_{x_{t,0}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j) \right), \ldots, \Gamma_{x_{t,1}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j) \right),$

- Add these transported together to get to $\zeta_{t,K}^j$:
  $\zeta_{t,K}^j = \sum_{k=0}^{K-1} \alpha_{t,k} \Gamma_{x_{t,k}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,k}} \sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j) \right);$

A new server aggregation: average of gradient stream

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 $\mathrm{T}_{\tilde{x}_t}\mathcal{M}$, i.e.,
  $\Gamma_{x_{t,0}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j) \right), \ldots, \Gamma_{x_{t,1}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j) \right),$

- Add these transported together to get to $\zeta_{t,K}^j$:
  $$\zeta_{t,K}^j = \sum_{k=0}^{K-1} \alpha_{t,k} \Gamma_{x_{t,k}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,k}} \sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j) \right);$$

The proposed server aggregation is given by

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

A new server aggregation: average of gradient stream

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 $\mathrm{T}_{\tilde{x}_t}\mathcal{M}$, i.e.,
  $\Gamma_{x_{t,0}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,0}} \sum_{b\in\mathcal{B}_{t,0}^j} \mathrm{grad} f_j(x_{t,0}^j; \xi_{t,0,b}^j) \right), \ldots, \Gamma_{x_{t,1}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,K-1}} \sum_{b\in\mathcal{B}_{t,K-1}^j} \mathrm{grad} f_j(x_{t,K-1}^j; \xi_{t,K-1,b}^j) \right),$

- Add these transported together to get to $\zeta_{t,K}^j$:
  $$\zeta_{t,K}^j = \sum_{k=0}^{K-1} \alpha_{t,k} \Gamma_{x_{t,k}^j}^{\tilde{x}_t} \left(\frac{1}{B_{t,k}} \sum_{b\in\mathcal{B}_{t,k}^j} \mathrm{grad} f_j(x_{t,k}^j; \xi_{t,k,b}^j) \right);$$

The proposed server aggregation is given by

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

The proposed aggregation is another generalization of the Euclidean aggregation.

Algorithm: Riemannian Federated Learning Averaging Gradient Stream

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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Set $\zeta_{t,0}^j \leftarrow 0_{\tilde{x}_t}$;
6. **for** $k = 1, 2, \ldots, K$ **do**
7. Agent j randomly samples an i.i.d. mini-batch $\mathcal{B}_{t,k-1}^j$ of size $B_{t,k-1}$;
8. Set $\eta_{t,k-1}^j \leftarrow -\frac{\alpha_{t,k-1}}{B_{t,k-1}} \sum_{b \in \mathcal{B}_{t,k-1}^j} \mathrm{grad} f_j(x_{t,k-1}^j; \xi_{t,k-1,b}^j)$;
9. Set $x_{t,k}^j \leftarrow \mathrm{R}_{x_{t,k-1}^j}(\eta_{k-1}^j)$, and $\zeta_{t,k}^j \leftarrow \zeta_{t,k-1}^j + \Gamma_{x_{t,k-1}^j}^{\tilde{x}_t}(\eta_{t,k-1}^j)$
10. **end for**
11. Sent $\zeta_{t,K}^j$ to the server;
12. **end for**
13. Server aggregates the received local parameter difference $\{\zeta_{t,K}^j\}_{j \in \mathcal{S}_t}$ by averaging

$$\tilde{x}_{t+1} \leftarrow \mathrm{R}_{\tilde{x}_t}\left(-\sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \zeta_{t,K}^j\right);$$

14. **end for**

Algorithm: Riemannian Federated Learning Averaging Gradient Stream

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 $\tilde{x}_t$ to all agents in $\mathcal{S}_t$, i.e., $x_{t,0}^j \leftarrow \tilde{x}_t$;
4. **for** $j \in \mathcal{S}_t$ in parallel **do**
5. Set $\zeta_{t,0}^j \leftarrow 0_{\tilde{x}_t}$;
6. **for** $k = 1, 2, \ldots, K$ **do**
7. Agent j randomly samples an i.i.d. mini-batch $B_{t,k-1}^j$ of size $B_{t,k-1}$;
8. Set $\eta_{t,k-1}^j \leftarrow -\frac{\alpha_{t,k-1}}{B_{t,k-1}} \sum_{b \in \mathcal{B}_{t,k-1}^j} \mathrm{grad} f_j(x_{t,k-1}^j; \xi_{t,k-1,b}^j)$;
9. Set $x_{t,k}^j \leftarrow \mathrm{R}_{x_{t,k-1}^j}(\eta_{k-1}^j)$, and $\zeta_{t,k}^j \leftarrow \zeta_{t,k-1}^j + \Gamma_{x_{t,k-1}^j}^{\tilde{x}_t}(\eta_{t,k-1}^j)$
10. **end for**
11. Sent $\zeta_{t,K}^j$ to the server;
12. **end for**
13. Server aggregates the received local parameter difference $\{\zeta_{t,K}^j\}_{j \in \mathcal{S}_t}$ by averaging

$$\tilde{x}_{t+1} \leftarrow \mathrm{R}_{\tilde{x}_t}\left(-\sum_{j \in \mathcal{S}_t} \frac{p_j}{\sum_{j \in \mathcal{S}_t} p_j} \zeta_{t,K}^j\right);$$

14. **end for**

- 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

1. Federated Learning Review

2. Riemannian Federated Learning Averaging Gradient Stream

3. Convergence Analysis

4. Numerical Experiments

5. Summary

Convergence Analysis

Assumptions:

- (Full Participation) Full agents participate in local updates at each communication round.
- (I.I.D. Data) Agent's data are subjected to an independently identical distribution.

Convergence Analysis

Assumptions:

- (Full Participation) Full agents participate in local updates at each communication round.
- (I.I.D. Data) Agent's data are subjected to an independently identical distribution.

We focus on expected risk minimization.

Assumptions:

- (Full Participation) Full agents participate in local updates at each communication round.
- (I.I.D. Data) Agent's data are subjected to an independently identical distribution.

We focus on expected risk minimization.

$$\min_{x\in\mathcal{M}} F(x) := \frac{1}{S}\sum_{i=1}^{S} \mathbb{E}_{\xi\sim\mathcal{D}_i}[f_i(x;\xi)]$$
$$= \mathbb{E}[f(x;\xi)]$$

- Each agent only has access to $f(x;\xi)$ and $\mathrm{grad} f(x;\xi)$.

Assumption 3.1

We assume that:

(1) $x^* = \arg\min_{x \in \mathcal{M}} F(x)$, *the outer iterates* $\{\tilde{x}_t\}_{t \geq 1}$ *and the inner iterates* $\{\{\{x_{t,k}^j\}_{j=1}^S\}_{k \geq 0}\}_{t \geq 1}$ *generated by FedAGS remain in a compact and connected subset* $\mathcal{W} \subseteq \mathcal{M}$;

(2) *the compact and connected subset* $\mathcal{W}$ *is totally retractive with respect to the retraction* R;

(3) *for each realization of* ξ, *the component* $f(\cdot; \xi)$ *are continuously differentiable;*

(4) *the vector transport* Γ *is isometric;*

(5) *the cost function* F *is* L*-retraction smooth and* L*-Lipchitz continuous differentiable with respect to* Γ *on* $\mathcal{W}$; *and*

(6) *the step sizes* $\alpha_{t,k}$ *are upper bounded, i.e., there exists* $A > 0$ *such that* $\alpha_{t,k} \leq A$ *for all* t *and* k.

Assumption 3.2

For any fixed parameter $x \in \mathcal{M}$, the Riemannian stochastic gradient $\mathrm{grad} f(x;\xi)$ is an unbiased estimator of the true gradient corresponding to the parameter x, i.e.,

$$\mathbb{E}_{\xi}[\mathrm{grad} f(x;\xi)] = \mathrm{grad} F(x)$$

Assumption 3.3

For any fixed parameter $x \in \mathcal{M}$, there exists a scalar $\sigma > 0$ such that for any mini-batch indices set $\mathcal{B}$ of the realizations of random variable ξ, the following holds

$$\mathbb{E}_{\mathcal{B}}\left[\left\|\frac{1}{B}\sum_{b\in\mathcal{B}}\mathrm{grad} f(x;\xi_b) - \mathrm{grad} F(x)\right\|^2\right] \leq \frac{\sigma^2}{B},$$

where B is the size of $\mathcal{B}$.

Assumption 3.2

For any fixed parameter $x \in \mathcal{M}$, the Riemannian stochastic gradient $\mathrm{grad} f(x;\xi)$ is an unbiased estimator of the true gradient corresponding to the parameter x, i.e.,

$$\mathbb{E}_{\xi}[\mathrm{grad} f(x;\xi)] = \mathrm{grad} F(x)$$

Assumption 3.3

For any fixed parameter $x \in \mathcal{M}$, there exists a scalar $\sigma > 0$ such that for any mini-batch indices set $\mathcal{B}$ of the realizations of random variable ξ, the following holds

$$\mathbb{E}_{\mathcal{B}}\left[\left\|\frac{1}{B}\sum_{b\in\mathcal{B}}\mathrm{grad} f(x;\xi_b) - \mathrm{grad} F(x)\right\|^2\right] \leq \frac{\sigma^2}{B},$$

where B is the size of $\mathcal{B}$.

Reasonability: all the assumptions have been used in exisitng Riemannian optimization or Federated learning algorithms.

Convergence Analysis: fixed step size

Theorem 1 (Nonconvex)

If we run RFedAGS with a fixed step size $\alpha_{t,k} = \bar{\alpha}$, a fixed batch size $B_{t,k} = \bar{B}$ such that the step size $\bar{\alpha}$ satisfies certain conditions. Then the resulting sequence of iterates $\{\tilde{x}_t\}_{t=1}^{T}$ satisfies

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathrm{grad}F(\tilde{x}_t)\|^2\right] \leq \frac{2(F(\tilde{x}_1)-F(x^*))}{T(K-1+\delta)\bar{\alpha}} + \frac{\bar{\alpha}K\sigma^2 L}{(K-1+\delta)}H(\bar{\alpha},K,S),$$

where $x^* \in \arg\min_{x\in\mathcal{M}} F(x)$.

Sublinearly converge to a neighborhood of the solution.

Convergence Analysis: fixed step size

Theorem 1 (Nonconvex)

If we run RFedAGS with a fixed step size $\alpha_{t,k} = \bar{\alpha}$, a fixed batch size $B_{t,k} = \bar{B}$ such that the step size $\bar{\alpha}$ satisfies certain conditions. Then the resulting sequence of iterates $\{\tilde{x}_t\}_{t=1}^{T}$ satisfies

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathrm{grad}F(\tilde{x}_t)\|^2\right] \leq \frac{2(F(\tilde{x}_1)-F(x^*))}{T(K-1+\delta)\bar{\alpha}} + \frac{\bar{\alpha}K\sigma^2 L}{(K-1+\delta)}H(\bar{\alpha},K,S),$$

where $x^* \in \arg\min_{x\in\mathcal{M}} F(x)$.

Suppose the fixed step size is chosen independent of K.

Then the larger K is, the faster the algorithm converges.

Theorem 2 (Riemannian Polyak-Łojasiewicz)

Under the same conditions as Theorem 1 together with assuming that the function F satisfies the Riemannian Polyak-Łojasiewicz (RPL) condition

$$F(x) - F(x^*) \leq \frac{1}{2\mu} \|\mathrm{grad} F(x)\|^2, \ \forall x \in \mathcal{W},$$

where $x^* = \arg\min_{x \in \mathcal{M}} F(x)$ and μ is a positive constant. Under certain conditions on $\bar{\alpha}$, we have

$$\mathbb{E}[F(\tilde{x}_T) - F(x^*)] \leq (1 - \mu\bar{\alpha}(K - 1 + \delta))^{T-1} \mathbb{E}[F(\tilde{x}_1) - F(x^*)] + \frac{K\bar{\alpha}\sigma^2 L}{2\mu\bar{B}(K - 1 + \delta)} H(\bar{\alpha}, K, S).$$

Linearly converge to a neighborhood of the solution.

Theorem 2 (Riemannian Polyak-Łojasiewicz)

Under the same conditions as Theorem 1 together with assuming that the function F satisfies the Riemannian Polyak-Łojasiewicz (RPL) condition

$$F(x) - F(x^*) \leq \frac{1}{2\mu}\|\mathrm{grad}F(x)\|^2, \ \forall x \in \mathcal{W},$$

where $x^* = \arg\min_{x\in\mathcal{M}} F(x)$ and μ is a positive constant. Under certain conditions on $\bar{\alpha}$, we have

$$\mathbb{E}[F(\tilde{x}_T) - F(x^*)] \leq (1 - \mu\bar{\alpha}(K - 1 + \delta))^{T-1}\mathbb{E}[F(\tilde{x}_1) - F(x^*)] + \frac{K\bar{\alpha}\sigma^2 L}{2\mu\bar{B}(K - 1 + \delta)}H(\bar{\alpha}, K, S).$$

Suppose the fixed step size is chosen independent of K.

Then the larger K is, the faster the algorithm converges.

Theorem 3 (Nonconvex)

If we run RFedAGS with decaying step sizes $\alpha_{t,k} = \bar{\alpha}_t$, and not fixed but bounded batch sizes $B_{t,k} = \bar{B}_t$ for outer iterations and $B_{\text{low}} \leq \bar{B}_t \leq B_{\text{up}}$ with B_{low} and B_{up} being positive integers, then the resulting sequence of iterates $\{\tilde{x}_t\}_{t=1}^T$ satisfies

$$\mathbb{E}\left[\sum_{t=1}^{T} \frac{\bar{\alpha}_t}{\sum_{t=1}^{T} \bar{\alpha}_t} \|\text{grad}F(\tilde{x}_t)\|^2\right] \leq \frac{2(F(\tilde{x}_1) - F(x^*))}{(K - 1 + \delta)\sum_{t=1}^{T} \bar{\alpha}_t} + \sum_{t=1}^{T} \frac{\bar{\alpha}_t^2 K \sigma^2 L}{(K - 1 + \delta)\bar{B}_t \sum_{t=1}^{T} \bar{\alpha}_t} H(\bar{\alpha}_t, K, S).$$

Further, if the step size $\bar{\alpha}_t$'s satisfy $\sum_{t=1}^{\infty} \bar{\alpha}_t = \infty$, and $\sum_{t=1}^{\infty} \bar{\alpha}_t^2 < \infty$, then the following holds

$$\lim_{T \to \infty} \mathbb{E}\left[\sum_{t=1}^{T} \frac{\alpha_t}{\sum_{t=1}^{T} \alpha_t} \|\text{grad}F(x_t)\|^2\right] = 0.$$

Suppose the decaying step size is chosen independent of K.

Then the larger K is, the faster the algorithm converges.

Theorem 3 (Nonconvex)

If we run RFedAGS with decaying step sizes $\alpha_{t,k} = \bar{\alpha}_t$, and not fixed but bounded batch sizes $B_{t,k} = \bar{B}_t$ for outer iterations and $B_{\text{low}} \leq \bar{B}_t \leq B_{\text{up}}$ with B_{low} and B_{up} being positive integers, then the resulting sequence of iterates $\{\tilde{x}_t\}_{t=1}^T$ satisfies

$$\mathbb{E}\left[\sum_{t=1}^{T} \frac{\bar{\alpha}_t}{\sum_{t=1}^{T} \bar{\alpha}_t} \|\text{grad}F(\tilde{x}_t)\|^2\right] \leq \frac{2(F(\tilde{x}_1) - F(x^*))}{(K-1+\delta)\sum_{t=1}^{T} \bar{\alpha}_t} + \sum_{t=1}^{T} \frac{\bar{\alpha}_t^2 K \sigma^2 L}{(K-1+\delta)\bar{B}_t \sum_{t=1}^{T} \bar{\alpha}_t} H(\bar{\alpha}_t, K, S).$$

Further, if the step size $\bar{\alpha}_t$'s satisfy $\sum_{t=1}^{\infty} \bar{\alpha}_t = \infty$, and $\sum_{t=1}^{\infty} \bar{\alpha}_t^2 < \infty$, then the following holds

$$\lim_{T \to \infty} \mathbb{E}\left[\sum_{t=1}^{T} \frac{\alpha_t}{\sum_{t=1}^{T} \alpha_t} \|\text{grad}F(x_t)\|^2\right] = 0.$$

If the decaying step size $\bar{\alpha}_t = \frac{\kappa}{(\gamma+t)^p}$ for constants $\kappa > 0, \gamma > 0$, and $p \in (1/2, 1]$, then

$$\mathbb{E}\left[\sum_{t=1}^{T} \frac{\bar{\alpha}_t}{\sum_{t=1}^{T} \bar{\alpha}_t} \|\text{grad}F(x_t)\|^2\right] = \begin{cases} \mathcal{O}\left(\frac{1}{\ln(\gamma+T)}\right) & p = 1, \\ \mathcal{O}\left(\frac{1}{(\gamma+T)^{1-p}}\right) & p \in (1/2, 1), \end{cases}$$

which converges sublinearly.

Theorem 4 (RPL)

Under the same conditions as Theorem 2 except for that the step size sequence and the batch size sequence satisfy

$$\alpha_{t,k} = \bar{\alpha}_t = \frac{\kappa}{\gamma + t} \text{ for some } \gamma > 0 \text{ and } \kappa \text{ satisfying certain assumptions}$$

$$\text{and } B_{t,k} = \bar{B}_t \in [B_{\text{low}}, B_{\text{up}}].$$

Then for all $t \in \{1, 2, \ldots, T-1\}$, *the expected optimality gap is bounded by*

$$\mathbb{E}[F(\tilde{x}_t) - F(x^*)] \leq \frac{\nu}{\gamma + t},$$

where $\nu = \max\{\frac{\kappa^2 K^2 \sigma^2 L}{SB_{\text{low}}(\kappa\mu(K-1+\delta)-1)}, \frac{\kappa^3(2K-1)K(K-1)\sigma^2 L^2 M}{3\gamma B_{\text{low}}(\kappa\mu(K-1+\delta)-1)}, (\gamma+1)(F(\tilde{x}_1) - F(x^*))\}$.

Theorem 4 (RPL)

Under the same conditions as Theorem 2 except for that the step size sequence and the batch size sequence satisfy

$$\alpha_{t,k} = \bar{\alpha}_t = \frac{\kappa}{\gamma + t} \text{ for some } \gamma > 0 \text{ and } \kappa \text{ satisfying certain assumptions}$$

$$\text{and } B_{t,k} = \bar{B}_t \in [B_{\text{low}}, B_{\text{up}}].$$

Then for all $t \in \{1, 2, \ldots, T-1\}$, *the expected optimality gap is bounded by*

$$\mathbb{E}[F(\tilde{x}_t) - F(x^*)] \leq \frac{\nu}{\gamma + t},$$

where $\nu = \max\{\frac{\kappa^2 K^2 \sigma^2 L}{SB_{\text{low}}(\kappa\mu(K-1+\delta)-1)}, \frac{\kappa^3(2K-1)K(K-1)\sigma^2 L^2 M}{3\gamma B_{\text{low}}(\kappa\mu(K-1+\delta)-1)}, (\gamma+1)(F(\tilde{x}_1) - F(x^*))\}$.

Convergence rate is improved from $O\left(\frac{1}{\ln(t)}\right)$ to $O\left(\frac{1}{t}\right)$.

All the above theorems rely on the below conditions for step sizes.

Theorem 5

We run RFedAGS with a fixed step size $\alpha_{t,k} = \bar{\alpha}_t$ and a fixed batch size $B_{t,k} = \bar{B}_t$ within parallel steps.

- *If $K = 1$ with step sizes $\bar{\alpha}_t$ satisfying*

$$2 - \delta \geq L\bar{\alpha}_t; \tag{3.1}$$

- *or $K > 1$ with step sizes $\bar{\alpha}_t$ satisfying*

$$\begin{cases} 1 \geq L^2\bar{\alpha}_t^2 M(K+1)(K-2) + \bar{\alpha}_t LK, \\ 1 - \delta \geq 2L^2\bar{\alpha}_t^2 M, \end{cases} \tag{3.2}$$

where $\delta \in (0,1)$ is some constant, then it holds that

$$\mathbb{E}_t[F(\tilde{x}_{t+1})] - F(\tilde{x}_t) \leq -\frac{\bar{\alpha}_t(K-1+\delta)}{2}\|\mathrm{grad}F(\tilde{x}_t)\|^2 + \frac{K\bar{\alpha}_t^2\sigma^2 L}{2\bar{B}_t}H(\bar{\alpha}_t, K, S),$$

where $H(\bar{\alpha}_t, K, S) = \frac{\bar{\alpha}_t(2K-1)(K-1)ML}{3} + \frac{K}{S}$, and the expectations above are taken over the randomness at the t-th outer iteration conditioned on $\tilde{x}_t$.

- Inspired from Euclidean results in [ZC18];
- M is a constant related to the manifold and retraction ($M = 1$ for Euclidean).

[ZC18] F. Zhou and G. Cong. On the convergence properties of a K-step averaging stochastic gradient descent algorithm for nonconvex optimization. International Joint Conference on Artificial Intelligence, 2018.

All the above theorems rely on the below conditions for step sizes.

Theorem 5

We run RFedAGS with a fixed step size $\alpha_{t,k} = \bar{\alpha}_t$ and a fixed batch size $B_{t,k} = \bar{B}_t$ within parallel steps.

- *If $K = 1$ with step sizes $\bar{\alpha}_t$ satisfying*

$$2 - \delta \geq L\bar{\alpha}_t; \tag{3.1}$$

- *or $K > 1$ with step sizes $\bar{\alpha}_t$ satisfying*

$$\begin{cases} 1 \geq L^2\bar{\alpha}_t^2 M(K+1)(K-2) + \bar{\alpha}_t LK, \\ 1 - \delta \geq 2L^2\bar{\alpha}_t^2 M, \end{cases} \tag{3.2}$$

where $\delta \in (0, 1)$ is some constant, then it holds that

$$\mathbb{E}_t[F(\tilde{x}_{t+1})] - F(\tilde{x}_t) \leq -\frac{\bar{\alpha}_t(K - 1 + \delta)}{2}\|\mathrm{grad}F(\tilde{x}_t)\|^2 + \frac{K\bar{\alpha}_t^2\sigma^2 L}{2\bar{B}_t}H(\bar{\alpha}_t, K, S),$$

where $H(\bar{\alpha}_t, K, S) = \frac{\bar{\alpha}_t(2K-1)(K-1)ML}{3} + \frac{K}{S}$, and the expectations above are taken over the randomness at the t-th outer iteration conditioned on $\tilde{x}_t$.

- $\frac{2}{\sqrt{K^2+4M(K+1)(K-2)}+K} \geq \sqrt{\frac{1-\delta}{2M}} \Rightarrow$ the second inequality in (3.2) takes effect;
- $M \geq \frac{(1-\delta)K^2}{2(1-(1-\delta)(K+1)(K-2))} \Rightarrow$ theoretical upper bound of $\bar{\alpha}_t$ independent of K.

Optimal choice for the number of local updates

Multiple inner iterations bring benefits.

Assume that step size α_t is chosen independent of K.

- The larger K is, the faster algorithms converge in the sense of the first terms in the upper bounds;
- The larger K is, the smaller the upper bounds (involving both terms) are if T is not too large.

Optimal choice for the number of local updates

Theorem 6 (Fixed step size)

We run RFedAGS with a fixed batch size $B_{t,k} = \bar{B}$ and a fixed step size $\alpha_{t,k} = \bar{\alpha}$ satisfying Conditions (3.1) *and* (3.2)*. Under the same conditions as Theorem 1, if the number of outer iterations T satisfies*

$$(F(x_1) - F(x^*)) > \frac{\alpha^2 TL\sigma^2}{SB} + \frac{\alpha^3\sigma^2 L^2 TM}{B},$$

then the optimal choice of K, the number of inner iterations, is greater than 1*, but not infinite.*

Theorem 7 (decaying step size)

We run RFedAGS with batch sizes $B_{t,k} = \bar{B}_t$ and decaying step sizes $\alpha_{t,k} = \bar{\alpha}_t$ such that $\bar{\alpha}_1$ satisfying Conditions (3.1) *and* (3.2)*. Under the same conditions as Theorem 3, if the number of outer iterations T satisfies*

$$(F(x_1) - F(x^*)) > \sigma^2 L \sum_{t=1}^{T} \frac{\alpha_t^2}{B_t}\left(\alpha_t ML + \frac{2}{S}\right),$$

then the optimal choice K is greater than 1*, but not infinite.*

Outline

1. Federated Learning Review

2. Riemannian Federated Learning Averaging Gradient Stream

3. Convergence Analysis

4. Numerical Experiments

5. Summary

Numerical Experiments

- The experiments conducted on the empirical risk minimization
- I.I.D. data and full agent participation

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

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

- $\min_{x \in \mathrm{S}^d} F(x) := -\frac{1}{S} \sum_{i=1}^{S} \frac{1}{N} \sum_{j=1}^{N} x^T (z_{i,j} z_{i,j}^T) x$ 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 $i = 1, \ldots, S$:
  - 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 $U_i \in \mathbb{R}^{N \times (d+1)}$ and $V \in \mathbb{R}^{(d+1) \times (d+1)}$ being two orthonormal matrix
  - Set $z_{i,j} = Z_i(j,:)^T$

Computing Fréchet mean over SPD manifolds (CFMSPD)

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

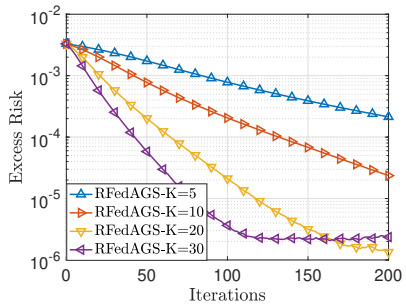
- $\min_{X \in \mathrm{St}(p,d)} F(X) = \frac{1}{S} \sum_{i=1}^{S} \sum_{j=1}^{N} \mathrm{trace}(X^T A_{i,j} X H)$ with $\mathrm{St}(p,d) = \{X \in \mathbb{R}^{d \times p} : X^T X = I_p\}$
- $H = \mathrm{diag}\{p, p-1, \ldots, 1\}$
- The objective locally satisfies the RPL condition
- Synthetize the 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 normal distribution

Three simulation experiments

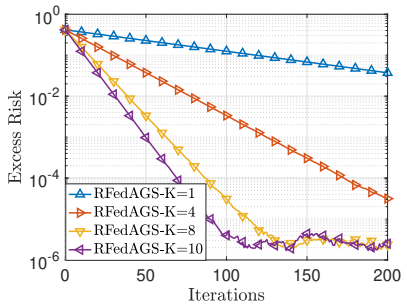
Table 1: The parameters of the three problems and RFedAGS. Notation $a.b_k$ denotes a number $a.b \times 10^k$ and the dash "$-$" means that the parameter does not exist in the problem.

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

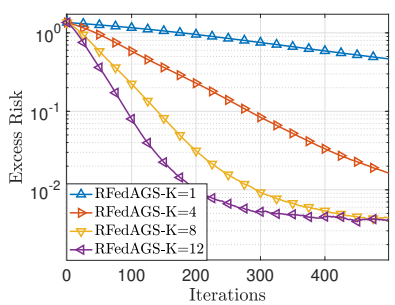
Three simulation experiments



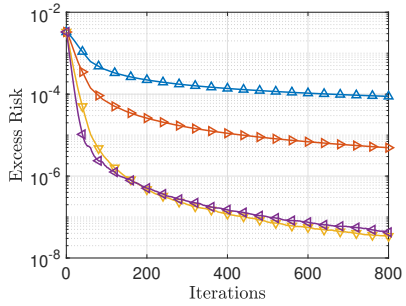
(a) CPESph, fixed



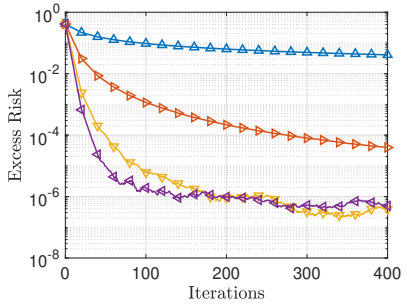
(b) CFMSPD, fixed



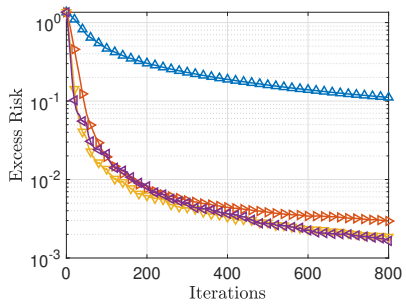
(c) MBCSti, fixed



(d) CPESph, decaying



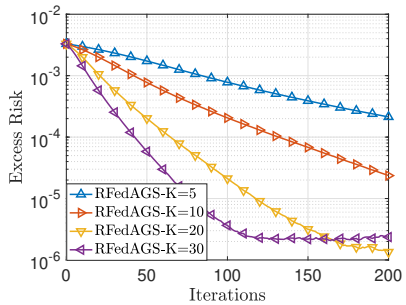
(e) CFMSPD, decaying



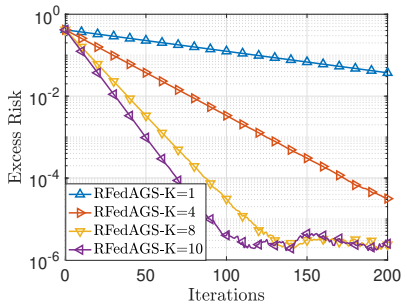
(f) MBCSti, decaying

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).

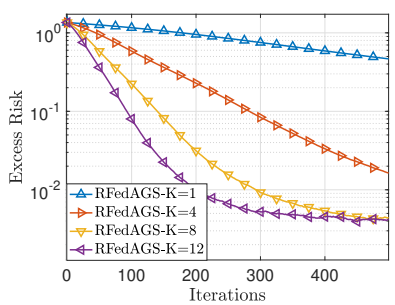
Three simulation experiments



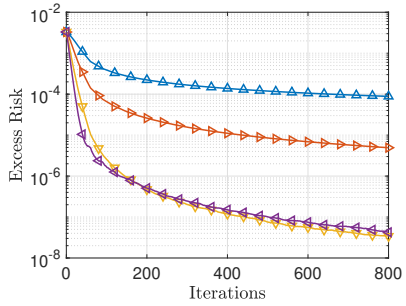
(a) CPESph, fixed



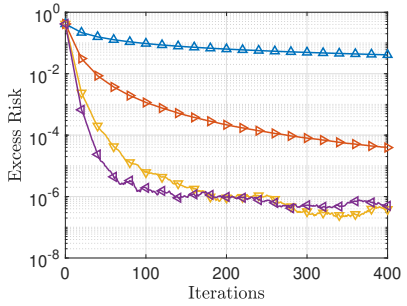
(b) CFMSPD, fixed



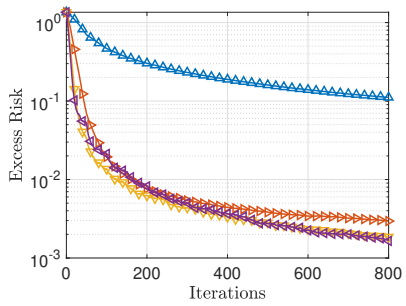
(c) MBCSti, fixed



(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 algorithm converges;
- Too large K may enlarge the accuracy for large T.

Comparison with existing methods

Principal component analysis (PCA) on the Stiefel manifold

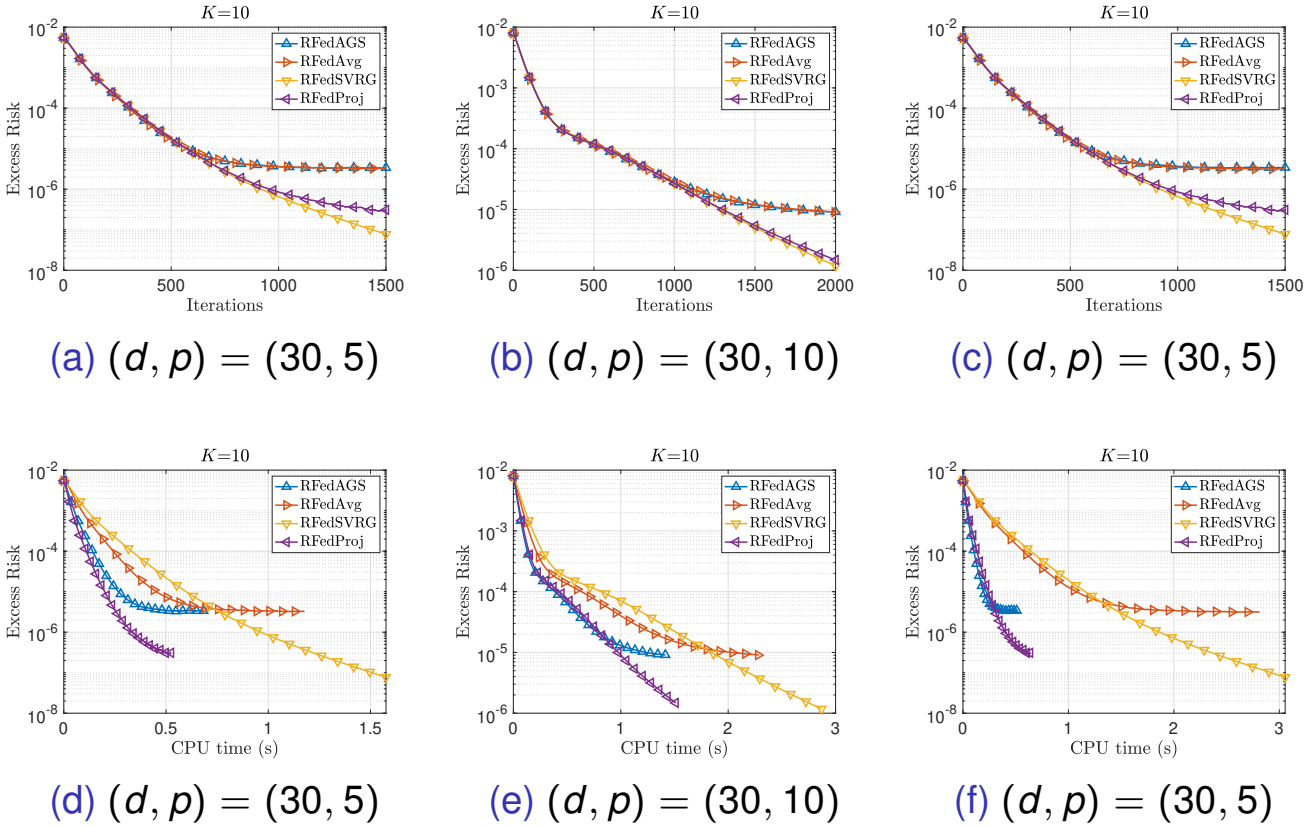
$$\min_{X \in \mathrm{St}(d,p)} F(X) = -\frac{1}{S}\sum_{i=1}^{S}\left(\frac{1}{N}\sum_{j=1}^{N}\mathrm{trace}(X^T(A_{i,j}A_{i,j}^T)X)\right)$$

- The samples generated identically to those in CPESph;
- Compared to RFedAvg [LM23], RFedSVRG [LM23], RFedProj [Zha+24];
- $(S, N) = (40, 80)$, $\alpha = 0.8$, $B = 40$, $K = 10$, and $(d, p) = (30, 5), (30, 10)$.

[LM23] J. Li and S. Ma. Federated Learning on Riemannian Manifolds. Applied Set-Valued Analysis and Optimization, 2023.

[Zha+24] 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.

Comparison with existing methods



(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 3: First two 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.

Summary

For more numerical experiments and 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;
- Proposed a new server aggregation;
- Convergence analysis;
- Numerical experiments;

Thank you

Thank you for your attention!

References I

S. Bonnabel. “Stochastic Gradient Descent on Riemannian Manifolds”. In: *IEEE Transactions on Automatic Control* 58.9 (2013), pp. 2217–2229. DOI: 10.1109/TAC.2013.2254619.

Zhenwei Huang et al. *Federated Learning on Riemannian Manifolds with Differential Privacy*. 2024. arXiv: 2404.10029 [cs.LG].

Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic Controlled Averaging for Federated Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. ICML’20. JMLR, 2020.

Jiaxiang Li and Shiqian Ma. “Federated Learning on Riemannian Manifolds”. In: *Applied Set-Valued Analysis and Optimization* 5.2 (2023).

References II

Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, 20–22 Apr 2017, pp. 1273–1282. URL: https://proceedings.mlr.press/v54/mcmahan17a.html.

Sashank J. Reddi et al. “Adaptive Federated Optimization”. In: *International Conference on Learning Representations*. 2021.

Fan Zhou and Guojing Cong. “On the convergence properties of a K-step averaging stochastic gradient descent algorithm for nonconvex optimization”. In: *International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence. 2018.

Jiaojiao Zhang et al. *Nonconvex Federated Learning on Compact Smooth Submanifolds With Heterogeneous Data*. 2024. arXiv: 2406.08465 [cs.LG]. URL: https://arxiv.org/abs/2406.08465.