FedSPDnet: Geometry-Aware Federated Deep Learning with SPDnet

Thibault Pautrel[1], Florent Bouchard[1], Ammar Mian[3], Guillaume Ginolhac[3]
[1] CentraleSupélec, L2S, France
[2]Université Savoie Mont-Blanc, LISTIC, France

***Abstract*—We introduce two federated learning frameworks for the classical SPDnet model operating on symmetric positive definite (SPD) matrices with Stiefel-constrained parameters. Unlike standard Euclidean averaging, which violates orthogonality, our approach preserves geometric structure through two efficient aggregation strategies: ProjAvg, projecting arithmetic means onto the Stiefel manifold, and RLAvg, approximating tangent-space averaging via retractions and liftings. Both methods are computationally efficient, independent of the optimizer, and enable scalable federated learning for signal processing applications whose features are SPD matrices. Simulations on EEG motor imagery benchmarks show that FedSPDnet outperforms federated EEGnet in F1 score and robustness to federation and partial participation, while using fewer parameters per communication round.**

***Index Terms*—Federated learning, Riemannian manifold, symmetric positive definite matrices**

I. Introduction

Federated learning (FL) enables collaborative model training without centralizing raw data, typically by iteratively averaging parameters on a central server as in FedAvg [1]–[3]. Subsequent refinements such as FedProx [3], [4] and SCAFFOLD [5] improve robustness under data heterogeneity and client drift, but remain limited to Euclidean parameter spaces and do not extend to non-Euclidean geometries.

In parallel, the last decade has seen a surge of interest in Riemannian optimization, which generalizes classical optimization to parameters living on manifolds [6], [7]. Such techniques are crucial for learning problems with geometric constraints, including low-rank matrix factorization, dimensionality reduction, and orthogonal parameterizations. In deep learning, Riemannian models have proven particularly effective when operating on structured data such as symmetric positive definite (SPD) covariance matrices, leading to architectures like SPDnet [8]. SPDnet integrates bilinear mappings on the Stiefel manifold with nonlinear eigenvalue rectification, ensuring that intermediate representations remain SPD while reducing dimensionality in a geometry-preserving way. This architecture has proven its effectiveness in several applications such as micro-Doppler radar [9], electroencephalography [10], and ground-penetrating radar [11].

Some works [12]–[15] have proposed adapting federated learning architectures to Riemannian manifolds, but these studies are fairly general and do not concern the SPDnet network. In this paper, we therefore propose new federated architectures that assume that all clients use the SPDnet network. To develop a manifold-based federated learning model, the main challenge lies in aggregating parameters: while local training on clients leverages optimization methods adapted to the manifold, the server-side averaging step must also be designed to obtain a result that belongs to the manifold. The optimal solution is therefore to use a Riemannian mean [7], which allows to obtain the barycenter of a set of parameters belonging to the specific manifold. Unfortunately, this step is tricky for SPDnet. This is because SPDnet network parameters are mainly orthogonal matrices that belong to the Stiefel manifold. However, it is well known [16] that calculating the Riemannian mean in this manifold is computationally expensive and therefore impossible to use for large-scale federated learning.

Inspired by the geometric tools from [16], we propose two lightweight, geometry-preserving aggregation strategies for SPDnet: (i) ProjAvg, which computes Euclidean averages of local weights and maps them back onto the Stiefel manifold via polar decomposition, and (ii) RLAvg, which approximates tangent-space Riemannian averaging using retractions and liftings. Both methods are computationally efficient, simple to implement, and guarantee that Stiefel constraints are preserved throughout training. By combining strong geometric motivations with practical algorithmic design, our work proposes a scalable deployment of SPDnet in federated learning applications. In particular, the proposed architecture is more robust than the one proposed in [14] because it does not depend on the local optimizer used in the clients. Numerical experiments on real EEG data show that the performance of our federated models matches that of a centralized model for a small number of clients.

II. Background

This section presents the key elements that are needed to develop our proposed federated learning method adapted to the SPDnet architecture. Since SPDnet weights are orthogonal, we first introduce Riemannian geometry and optimization on the Stiefel manifold. Then, we provide details on the SPDnet architecture.

A. Riemannian geometry and optimization on Stiefel

The Stiefel manifold [6], [7] is defined as $\mathrm{St}_{p,k} = \{\boldsymbol{W} \in \mathbb{R}^{p\times k} : \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}_k\}$. At $\boldsymbol{W} \in \mathrm{St}_{p,k}$, the tangent space is $T_{\boldsymbol{W}}\mathrm{St}_{p,k} = \{\boldsymbol{X} \in \mathbb{R}^{p\times k} : \boldsymbol{W}^\top \boldsymbol{X} + \boldsymbol{X}^\top \boldsymbol{W} = \boldsymbol{0}\}$. We endow it with the Euclidean metric: for $\boldsymbol{X}, \boldsymbol{Y} \in T_{\boldsymbol{W}}\mathrm{St}_{p,k}$, $\langle \boldsymbol{X}, \boldsymbol{Y}\rangle_{\boldsymbol{W}} = \mathrm{tr}(\boldsymbol{X}^\top \boldsymbol{Y})$. From there, in the context of federated learning, we need geometrical tools on $\mathrm{St}_{p,k}$ to: *(i)* perform Riemannian optimization, and *(ii)* aggregate points on $\mathrm{St}_{p,k}$.

Concerning the optimization of a cost function, one needs the Riemannian gradient to obtain a descent direction on the tangent space and a retraction to get a new iterate on the manifold from it. Given a cost function $f : \mathrm{St}_{p,k} \to \mathbb{R}$, the Riemannian gradient $\mathrm{grad} f(\boldsymbol{W})$ at $\boldsymbol{W}$ is the unique tangent vector such that, $\forall \boldsymbol{X} \in T_{\boldsymbol{W}}\mathrm{St}_{p,k}$, $\langle \mathrm{grad} f(\boldsymbol{W}), \boldsymbol{X}\rangle_{\boldsymbol{W}} = \mathrm{D} f(\boldsymbol{W})[\boldsymbol{X}]$, where $\mathrm{D} f(\boldsymbol{W})[\boldsymbol{X}]$ denotes the directional derivative of f at $\boldsymbol{W}$ in direction $\boldsymbol{X}$. For $\mathrm{St}_{p,k}$, it can be obtained from the Euclidean gradient $\nabla f(\boldsymbol{W})$ through $\mathrm{grad} f(\boldsymbol{W}) = P_{\boldsymbol{W}}(\nabla f(\boldsymbol{W}))$, where $P_{\boldsymbol{W}} : \mathbb{R}^{p\times k} \to T_{\boldsymbol{W}}\mathrm{St}_{p,k}$ is the orthogonal projection [6], [7]

$$P_{\boldsymbol{W}}(\boldsymbol{X}) = \boldsymbol{X} - \frac{1}{2}\boldsymbol{W}(\boldsymbol{W}^\top \boldsymbol{X} + \boldsymbol{X}^\top \boldsymbol{W}). \tag{1}$$

On Stiefel, an advantageous retraction $R_{\boldsymbol{W}} : T_{\boldsymbol{W}}\mathrm{St}_{p,k} \to \mathrm{St}_{p,k}$ at $\boldsymbol{W}$ is [6], [7]

$$R_{\boldsymbol{W}}(\boldsymbol{X}) = \mathcal{P}(\boldsymbol{W} + \boldsymbol{X}) = \mathrm{uf}(\boldsymbol{W} + \boldsymbol{X}), \tag{2}$$

where $\mathcal{P} : \mathbb{R}^{p\times k} \to \mathrm{St}_{p,k}$ is the projection on Stiefel obtained by taking the orthogonal factor of the polar decomposition.

To aggregate points on a Riemannian manifold, the natural solution is to exploit the Riemannian mean. To be able to compute it, one needs three elements: the Riemannian exponential and logarithm mappings, and the Riemannian distance [6], [7]. These three objects rely on geodesics, which generalize the concept of straight lines on the manifold. Unfortunately, for $\mathrm{St}_{p,k}$, the Riemannian logarithm and distance are unknown. Hence, the Riemannian mean is not available and another solution needs to be found to aggregate points. It will be the contributions of Section III-B.

B. *SPDnet architecture*

SPDnet [8] is a lightweight deep learning architecture (see Figure 1) operating on SPD (covariance) matrices in a SPD-preserving way while reducing dimension thanks to **BiRe** blocks. Such blocks combine a **BiMap** layer, performing bilinear projection for dimensionality reduction: $\bar{\boldsymbol{\Sigma}}_\ell = \boldsymbol{W}_\ell^\top \boldsymbol{\Sigma}_{\ell-1} \boldsymbol{W}_\ell$, $\boldsymbol{W}_\ell \in \mathrm{St}_{d_{\ell-1},d_\ell}$, and a **ReEig** layer inducing non-linearity by eigenvalues rectification: $\boldsymbol{\Sigma}_\ell = \boldsymbol{U}\max(\varepsilon \boldsymbol{I}_{d_\ell}, \boldsymbol{\Lambda})\boldsymbol{U}^\top$, $\bar{\boldsymbol{\Sigma}}_\ell = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$. After L **BiRe** blocks, the representation $\boldsymbol{\Sigma}_{L-1}$ is mapped from the SPD manifold to the Euclidean space through the **LogEig** layer $\boldsymbol{\Sigma}_L = \boldsymbol{U}\log(\boldsymbol{\Lambda})\boldsymbol{U}^\top$, $\boldsymbol{\Sigma}_{L-1} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ and finally fed to a softmax classifier via half-vectorization, $\hat{y} = \mathrm{softmax}(\boldsymbol{\xi}\mathrm{vech}(\boldsymbol{\Sigma}_L) + \boldsymbol{\beta})$, $\boldsymbol{\xi} \in \mathbb{R}^{K\times d_L(d_L+1)/2}$, $\boldsymbol{\beta} \in \mathbb{R}^K$, where K is the number of classes. The learnable parameters are $\theta = (\boldsymbol{W}_1, \dots, \boldsymbol{W}_L, \boldsymbol{\xi}, \boldsymbol{\beta})$.

III. Federated Learning with SPDnet Architecture

In this section, we give the main contribution of the paper: a new federated learning framework adapted to the SPDnet architecture, called FedSPDnet. The main challenge lies in the aggregation step performed in the central server, which is delicate for the SPDnet model. Indeed, despite the works proposed to develop federated learning on manifolds [12]–[15], they are unfortunately not appropriate in the context of SPDnet due to the geometric structure of the parameters (the weights belong to the Stiefel manifold). As stated in the background, this is because the Riemannian mean of parameters living on the Stiefel manifold is either untractable or computationally expensive [7]. Therefore, this strategy is impossible to use for large-scale federated learning.

In this paper, we propose two new strategies to aggregate weight matrices in Stiefel. The first is inspired by [12] but unlike computing the true Riemannian mean, the efficient mean on Stiefel from [16] is exploited. Our second strategy is inspired by [15], which avoids having to compute a mean on the manifold by averaging local (transported) Riemannian gradients instead. However, this approach is limited to using a specific Riemannian stochastic gradient descent for the local optimizer on every client. Our proposed method is more general and allows to exploit any optimizer (e.g. SGD, Adam, ...) for each client, which works well with SPDnet. FedSPDnet is summarized in Algorithm 1.

A. *Proposed model*

Let $\mathcal{C} = \{1, \dots, N\}$ denote the set of all clients. The client i has the training dataset

$$\mathcal{D}^{(i)} = \left\{(\boldsymbol{\Sigma}_j^{(i)}, y_j^{(i)}) \in \mathcal{S}_{d_0}^{++} \times [\![1, K]\!]\right\}_{j=1}^{|\mathcal{D}^{(i)}|}, \tag{3}$$

where $y_j^{(i)}$ is the class label associated with the covariance $\boldsymbol{\Sigma}_j^{(i)}$ and $|\mathcal{D}^{(i)}|$ is the cardinal of the dataset $\mathcal{D}^{(i)}$. Given SPDnet learnable parameters $\theta = (\boldsymbol{W}_1, \dots, \boldsymbol{W}_L, \boldsymbol{\xi}, \boldsymbol{\beta})$, with, $\forall \ell \in [\![1, L]\!]$, BiMap weights $\boldsymbol{W}_\ell \in \mathrm{St}_{d_{\ell-1},d_\ell}$, and softmax parameters $\boldsymbol{\xi} \in \mathbb{R}^{K\times d_L(d_L+1)/2}$ and $\boldsymbol{\beta} \in \mathbb{R}^K$, the empirical risk based on cross-entropy [17] on client i is

$$F_i(\theta) = \frac{-1}{|\mathcal{D}^{(i)}|} \sum_{j=1}^{|\mathcal{D}^{(i)}|} \sum_{k=1}^{K} \delta_{y_j^{(i)}=k} \log\left([f_\theta(\boldsymbol{\Sigma}_j^{(i)})]_k\right), \tag{4}$$

where $\delta_{y_j^{(i)}=k}$ is the Kronecker delta and $f_\theta(\boldsymbol{\Sigma}_j^{(i)})$ denotes the forward pass of $\boldsymbol{\Sigma}_j^{(i)}$ through SPDnet.

On the server, the objective is to minimize the global empirical risk over all clients, *i.e.*, solve

$$\underset{\theta}{\mathrm{argmin}} \quad \frac{1}{N}\sum_{i=1}^{N} F_i(\theta). \tag{5}$$

To find a solution collaboratively, T client-server communication rounds are performed. At each round t, a subset $\mathcal{S}_t \subset \mathcal{C}$ of M clients is selected uniformly at random.

The first step of the round t consists in optimizing every local SPDnet model on the selected clients. For each client $i \in \mathcal{S}_t$, the global parameters $\theta_t = (\boldsymbol{W}_{1,t}, \dots, \boldsymbol{W}_{L,t}, \boldsymbol{\xi}_t, \boldsymbol{\beta}_t)$ are set to the local SPDnet architecture. Then, E_i local epochs of standard SPDnet training are performed. All parameters are optimized with Adam [18]: Stiefel weights $\boldsymbol{W}_{\ell,t}$, $\ell \in [\![1, L]\!]$,

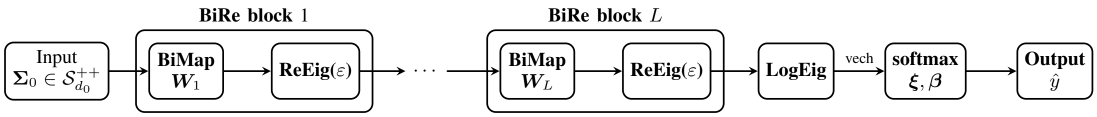

Figure 1. General SPDnet architecture [8]: a chain of L stacked **BiRe** (**BiMap** + **ReEig**) blocks, followed by **LogEig** mapping to Euclidean space and a fully connected softmax classifier. Learnable parameters are Stiefel weights $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$ and Euclidean weights $\boldsymbol{\xi}$, and bias $\boldsymbol{\beta}$.

are handled via tangent-space local trivialization [19], while softmax parameters $\boldsymbol{\xi}_t, \boldsymbol{\beta}_t$ are updated in the standard Euclidean way. After E_i local epochs, client i returns updated parameters $\theta_t^{(i)}$ to the server.

From there, the second step, which is the key issue in this work, is to aggregate the set of parameters $\{\theta_t^{(i)}\}_{i\in\mathcal{S}_t}$ from every selected client to obtain the new global parameters θ_{t+1}. For the Euclidean softmax parameters, the usual averaging technique from [1] can be employed, *i.e.*,

$$\boldsymbol{\xi}_{t+1} = \frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\boldsymbol{\xi}_t^{(i)}, \quad \boldsymbol{\beta}_{t+1} = \frac{1}{|\mathcal{S}_t|}\sum_{i\in S_t}\boldsymbol{\beta}_t^{(i)}. \tag{6}$$

However, this standard procedure cannot be applied to the Stiefel weights. This issue is overcome in the next subsection.

B. Orthogonality-preserving aggregation

In this subsection, we propose two strategies to aggregate weights $\{\boldsymbol{W}_{\ell,t}^{(i)}\}_{i\in\mathcal{S}_t}$ in the Stiefel manifold $\mathrm{St}_{d_{\ell-1},d_\ell}$. The first one is a straightforward adaptation of the usual federated learning averaging method [1]. It relies on computing a suitable mean on the Stiefel manifold. As discussed in Section II-A, the Riemannian mean on Stiefel is computationally intractable. Following [16], we instead project the arithmetic mean of the local weights onto the manifold. Concretely, given weights $\{\boldsymbol{W}_{\ell,t}^{(i)}\}_{i\in\mathcal{S}_t}$, the aggregated weight reads

$$\boldsymbol{W}_{\ell,t+1} = \mathcal{P}\left(\frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\boldsymbol{W}_{\ell,t}^{(i)}\right), \tag{7}$$

where the projection $\mathcal{P}$ is defined in (2). As explained in [16], this projection is the one that best suits the geometry of the Stiefel manifold. In the following, the corresponding FedSPDnet method is denoted ProjAvg. Our second strategy is inspired by [15]. In this latter work, authors avoid having to average parameters on the manifold by noticing that the usual federated learning average used to get the new global parameter can be rewritten as the sum of the previous global parameter and an average of some gradients of the local empirical risks F_i. They then adapt this Euclidean formula to the Riemannian case to obtain a new federated learning method. This allows them to only have to compute an average in the tangent space of the previous global parameters, which is fairly simple, before going back on the manifold through a retraction to get the new global parameters. While this strategy is very appealing, it still suffers from a fundamental limitation. Indeed, it requires the user to employ a specific stochastic gradient descent method for the local optimization procedure on the clients, which might not be adapted. To overcome this issue, we adopt the same strategy with a different rewriting of the Euclidean average. If the weights are Euclidean, we get the same update formula as in (6), which can be rewritten as

$$\boldsymbol{W}_{\ell,t+1} = \frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\boldsymbol{W}_{\ell,t}^{(i)} = \boldsymbol{W}_{\ell,t} + \frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}(\boldsymbol{W}_{\ell,t}^{(i)} - \boldsymbol{W}_{\ell,t}). \tag{8}$$

One can then notice that we have

$$\boldsymbol{W}_{\ell,t+1} = \exp_{\boldsymbol{W}_{\ell,t}}\left(\frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}\log_{\boldsymbol{W}_{\ell,t}}(\boldsymbol{W}_{\ell,t}^{(i)})\right), \tag{9}$$

where $\exp_{\boldsymbol{W}}(\boldsymbol{X}) = \boldsymbol{W}+\boldsymbol{X}$ and $\log_{\boldsymbol{W}}(\bar{\boldsymbol{W}}) = \bar{\boldsymbol{W}}-\boldsymbol{W}$ denote the Euclidean exponential and logarithm mappings, respectively. Unfortunately, we cannot apply this formula directly on the Stiefel manifold because the Riemannian logarithm is unknown and the Riemannian exponential is numerically expensive and might not be advantageous [7]. Instead, as in [16], we exploit some approximation of these tools. The Riemannian exponential and logarithm can be replaced with a retraction R and a lifting L, respectively. For Stiefel, as discussed in [16], the best choices appear to be the retraction (2) and the lifting

$$L_{\boldsymbol{W}}(\bar{\boldsymbol{W}}) = P_{\boldsymbol{W}}(\bar{\boldsymbol{W}} - \boldsymbol{W}), \tag{10}$$

where P is defined in (1). Finally, the aggregation formula is

$$\boldsymbol{W}_{\ell,t+1} = R_{\boldsymbol{W}_{\ell,t}}\left(\frac{1}{|\mathcal{S}_t|}\sum_{i\in\mathcal{S}_t}L_{\boldsymbol{W}_{\ell,t}}(\boldsymbol{W}_{\ell,t}^{(i)})\right). \tag{11}$$

In the following, the corresponding FedSPDnet method is denoted RLAvg. The resulting algorithm is provided in Algorithm 1.

IV. Numerical experiments

We evaluate FedSPDnet on EEG-based motor imagery classification, where spatial covariance matrices are effective descriptors [20], [21] and multi-site data confidentiality naturally motivates federated learning. Unlike traditional BCI pipelines [22], [23] that rely on within-subject calibration, we target population-level models that generalize across subjects without per-user data, reflecting realistic federated scenarios. As a Euclidean baseline, we consider EEGnet [24], a compact convolutional network operating on raw signals, federated with standard FedAvg [1].

A. Experimental setup

We use two motor imagery datasets from the MOABB benchmark [25]: **Weibo2014** (10 subjects, 60 channels, sampling frequency $f_s = 200\,\mathrm{Hz}$, 7 classes, $\sim$80 trials/class) and **PhysionetMI** (106 subjects, 64 channels, sampling frequency

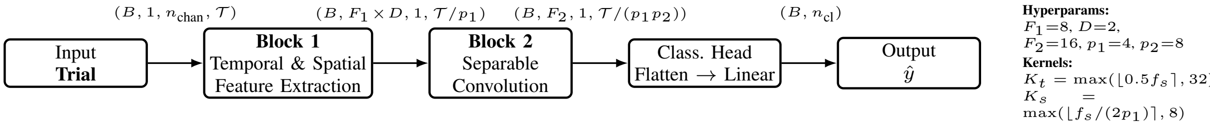

Figure 2. EEGnet overall pipeline.

Algorithm 1 FedSPDnet

Input: number of rounds T, number of clients per round M, number of local epochs E

Initialize randomly $\theta_0 = (\boldsymbol{W}_{1,0}, \ldots, \boldsymbol{W}_{L,0}, \boldsymbol{\xi}_0, \boldsymbol{\beta}_0)$
for $t = 0, \ldots, T-1$ **do**
    Randomly sample $S_t \subset \mathcal{C}$ with $|S_t| = M$
    for each client $i \in S_t$ **in parallel do**
        Initialize local SPDnet with θ_t
        Train SPDnet over E epochs to get $\theta_t^{(i)}$
        Send $\theta_t^{(i)}$ to the server
    end for
    Get $\boldsymbol{\xi}_{t+1}$ and $\boldsymbol{\beta}_{t+1}$ with (6)
    for $\ell = 1, \ldots, L$ **do**
        Compute $\boldsymbol{W}_{\ell,t+1}$ with either (7) or (11)
    end for
    $\theta_{t+1} = (\boldsymbol{W}_{1,t+1}, \ldots, \boldsymbol{W}_{L,t+1}, \boldsymbol{\xi}_{t+1}, \boldsymbol{\beta}_{t+1})$
end for

Output: θ_T

$f_s = 160$ Hz, 4 classes, ~23 trials/class). Three PhysionetMI subjects were excluded due to incomplete recordings. All signals are band-pass filtered in $[8, 32]$ Hz.

EEGnet [24] operates on z-score normalized raw trials $X_j \in \mathbb{R}^{n_{\text{chan}} \times \mathcal{T}}$ through temporal, depthwise spatial, and separable convolutions followed by a classification head (Figure 2).

SPDnet [8] operates on the sample covariance matrix (SCM) of each centered trial $\Sigma_j = \frac{1}{\mathcal{T}-1} \bar{X}_j \bar{X}_j^\top \in \mathcal{S}^{++}_{n_{\text{chan}}}$. A single BiRe block is used, with hidden dimension d and ReEig threshold ε chosen per dataset. Prior spectral analysis of the sample covariances sets ε to balance stability against discriminative information ($\varepsilon = 10^{-1}$ for Weibo2014, 10^{-2} for PhysionetMI), and a sweep over d retains the smallest value maximising validation F1 ($d = 22$ for Weibo2014, $d = 18$ for PhysionetMI), yielding models more compact than EEGnet.

Both architectures are trained with cross-entropy loss using Adam (through local tangent space parametrisation for SPDnet), initial learning rate 10^{-3}, batch size 64, and ReduceLROnPlateau scheduling (patience 20, factor 0.5). All results are reported over 10 independent runs.

B. Centralized baselines

As an upper bound on federated performance, we pool all subjects and perform a stratified split into training (75%), validation (10%), and test (15%) sets. Training runs for at most 300 epochs with early stopping (patience 75), retaining the best validation checkpoint [26]. Results (Table I) show comparable

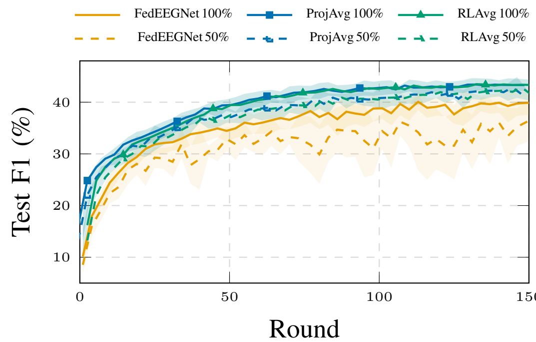

Figure 3. Convergence on Weibo2014 under full (solid) and half (dashed) participation. Shaded bands: ±1 std over 10 seeds. ProjAvg/RLAvg curves slightly offset horizontally.

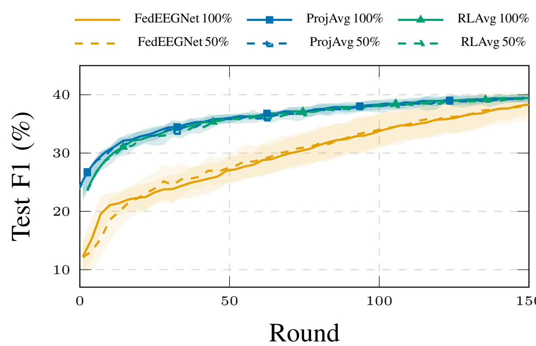

Figure 4. Convergence on PhysionetMI under full (solid) and half (dashed) participation. Shaded bands: ±1 std over 10 seeds. ProjAvg/RLAvg curves slightly offset horizontally.

centralized performance: SPDnet slightly outperforms EEGnet on Weibo2014 while EEGnet leads on PhysionetMI.

C. Federated learning experiments

We simulate a multi-institutional setting by assigning two subjects per client – mimicking small clinical sites – yielding $N = 5$ clients for Weibo2014 and $N = 53$ for PhysionetMI. Each client applies the same 75/10/15 stratified split to its local data. Training follows Algorithm 1 with $T = 150$ rounds and $E = 2$ local epochs, matching the centralized budget of 300 total epochs. We consider both full participation (all clients every round) and half participation ($M = \lfloor N/2 \rfloor$ clients).

For FedEEGnet, we apply FedAvg excluding batch normalization statistics from aggregation [27].

D. Results and analysis

a) Aggregation scheme equivalence: ProjAvg and RLAvg share the same per-round complexity of $\mathcal{O}(M n_{\text{chan}} d + n_{\text{chan}} d^2)$.

Table I
FINAL TEST F1 (%) FOR CENTRALIZED AND FEDERATED APPROACHES. MEAN ± STD OVER 10 SEEDS; BEST PER ROW IN **BOLD**.

Dataset		EEGnet *(FedAvg)*	SPDnet *(FedSPDnet)*
Weibo2014	*# params*	*5 303*	*4 715*
	Centr.	50.7 ± 1.5	$\mathbf{51.7 \pm 0.8}$
	Full	39.9 ± 2.4	$\mathbf{43.3 \pm 1.0}$
	Half	36.4 ± 4.1	$\mathbf{41.2 \pm 0.8}$
PhysionetMI	*# params*	*3 284*	*2 452*
	Centr.	$\mathbf{50.8 \pm 0.7}$	43.1 ± 0.5
	Full	38.3 ± 1.8	$\mathbf{39.5 \pm 0.6}$
	Half	38.0 ± 2.7	$\mathbf{39.5 \pm 0.9}$

As shown in Figures 3 and 4, their validation F1 curves overlap throughout all communication rounds on both datasets, under both full and partial participation. Since ProjAvg has slightly lower constant factors and does not require storing the previous global iterate, it is adopted as the default in Table I.

b) Federated performance: FedSPDnet consistently retains more of its centralized performance than FedEEGnet across both datasets and participation rates, despite using fewer parameters (Table I). This advantage holds even on PhysionetMI where SPDnet's centralized score is well below EEGnet's, suggesting that geometry-aware aggregation provides intrinsic regularisation absent in Euclidean averaging. FedSPDnet also converges faster, reaching its plateau in fewer communication rounds while FedEEGnet continues to oscillate or improve slowly (Figures 3 and 4), and exhibits lower variance across seeds.

V. Conclusion

We proposed FedSPDnet, a federated learning framework for SPDnet with two lightweight, optimizer-agnostic, geometry-preserving aggregation strategies (ProjAvg and RLAvg) that rely only on standard linear algebra. Both are empirically equivalent; ProjAvg is recommended for its simpler, storage-free implementation. On two EEG motor imagery benchmarks, FedSPDnet retains most of its centralized accuracy, degrades gracefully under partial participation and data heterogeneity, and converges faster than federated EEGnet while communicating fewer parameters per round.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 5693–5700.

[3] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[4] X. Yuan and P. Li, "On convergence of fedprox: Local dissimilarity invariant bounds, non-smoothness and beyond," *Advances in Neural Information Processing Systems*, vol. 35, pp. 10 752–10 765, 2022.

[5] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*. PMLR, 2020, pp. 5132–5143.

[6] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

[7] N. Boumal, *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[8] Z. Huang and L. Van Gool, "A riemannian network for spd matrix learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.

[9] D. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord, "Riemannian batch normalization for spd neural networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[10] R. J. Kobler, J. ichiro Hirayama, Q. Zhao, and M. Kawanabe, "SPD domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG," in *Neurips*, 2022.

[11] D. Jafuno, A. Mian, G. Ginolhac, and N. Stelzenmuller, "Classification of buried objects from ground penetrating radar images by using second order deep learning models," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2025.

[12] J. Li and S. Ma, "Federated learning on riemannian manifolds," *arXiv preprint arXiv:2206.05668*, 2022.

[13] J. Zhang, J. Hu, A. M.-C. So, and M. Johansson, "Nonconvex federated learning on compact smooth submanifolds with heterogeneous data," *Advances in Neural Information Processing Systems*, vol. 37, pp. 109 817–109 844, 2024.

[14] Z. Huang, W. Huang, P. Jawanpuria, and B. Mishra, "Federated learning on riemannian manifolds with differential privacy," *arXiv preprint arXiv:2404.10029*, 2024.

[15] ——, "Riemannian federated learning via averaging gradient stream," *arXiv preprint arXiv:2409.07223*, 2024.

[16] F. Bouchard, N. Laurent, S. Said, and N. Le Bihan, "Beyond R-barycenters: An effective averaging method on Stiefel and Grassmann manifolds," *IEEE Signal Processing Letters*, vol. 32, pp. 1950–1954, 2025.

[17] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[18] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] P. Ablin, S. Vary, B. Gao, and P.-A. Absil, "Infeasible deterministic, stochastic, and variance-reduction algorithms for optimization under orthogonality constraints," *Journal of Machine Learning Research*, vol. 25, no. 389, pp. 1–38, 2024.

[20] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass brain–computer interface classification by riemannian geometry," *IEEE transactions on biomedical engineering*, vol. 59, no. 4, pp. 920–928, 2012.

[21] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition?" in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2070–2078.

[22] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update," *Journal of neural engineering*, vol. 15, no. 3, p. 031005, 2018.

[23] I. Carrara, B. Aristimunha, M.-C. Corsi, R. Y. de Camargo, S. Chevallier, and T. Papadopoulo, "Geometric neural network based on phase space for bci-eeg decoding," *Journal of Neural Engineering*, vol. 22, no. 1, p. 016049, 2025.

[24] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces," *Journal of neural engineering*, vol. 15, no. 5, p. 056013, 2018.

[25] S. Chevallier, I. Carrara, B. Aristimunha, P. Guetschel, S. Sedlar, B. Lopes, S. Velut, S. Khazem, and T. Moreau, "The largest eeg-based bci reproducibility study for open science: the moabb benchmark," *arXiv preprint arXiv:2404.15319*, 2024.

[26] L. Prechelt, "Early stopping—but when?" in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[27] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," *arXiv preprint arXiv:2102.07623*, 2021.