
Siegel Neural Networks

Xuan Son Nguyen Aymeric Histace Nistor Grozavu
ETIS, UMR 8051, CY Cergy Paris University, ENSEA, CNRS
{xuan-son.nguyen, aymeric.histace}@ensea.fr
nistor.grozavu@cyu.fr

This supplemental material provides the experimental details, key concepts, basic facts, definitions, and mathematical proofs for the theoretical results presented in our paper. Appendix 1 gives more details on our experiments. In Appendix 2, we discuss an extension and social impacts of our methods. Appendix 3 presents definitions and basic facts used in our paper. Mathematical proofs of our theoretical results are detailed in Appendix 4.

1 Experimental Details

1.1 Radar Clutter Classification

1.1.1 Datasets and Experimental Settings

We use the method in [10] to create our datasets. For each time series of length N , we simulate the first q temporal elements using the following equation:

$$z = r^{\frac{1}{2}}x,$$

where r is a Block-Toeplitz HPD matrix [55] and x is a standard complex Gaussian random vector whose dimension is equal to the dimension of the time series to simulate times the length of the time series. We then simulate the $N - q$ remaining elements using the following equation:

$$u_n + \sum_{j=1}^q c_j u_{n-j} = v_n,$$

where $u_n \in \mathbb{C}^m$ is the vector of signals at time n , $c_j \in \mathbb{C}^{m \times m}$, $j = 1, \dots, q$ are the prediction coefficients (AR parameters), and $v_n \in \mathbb{C}^m$ is the prediction error at time n which is assumed to be a multidimensional Gaussian random variable. Time series of the same class are generated from the same matrix r in the above equation.

For each dataset, we generate the same number of time series per class, half of which is used for training and the remaining samples are used for testing. Tab. 4 provides the settings for simulating our datasets.

1.1.2 Optimization and Hyperparameters

For SPDNet¹ and SPDNetBN², we use the architectures in [45, 54] with three Bimap layers. All the Bimap layers output SPD matrices having the same size as input matrices.

MLR-AI and GyroSpd++ are implemented by following closely [60, 61]. We experiment with the architecture of GyroSpd++ in [61] which uses affine-invariant Riemannian metrics for the convolutional layer and Log-Euclidean metrics for the MLR layer. The convolutional layer of GyroSpd++ outputs an SPD matrix of the same size as the input SPD matrix.

¹<https://github.com/zhiwu-huang/SPDNet>.

²<https://papers.nips.cc/paper/2019/hash/6e69ebbfad976d4637bb4b39de261bf7-Abstract.html>.

Parameter	Dimension	Length	Number of classes	Number of time series per class	Order	Size
Dataset 1	3	20	600	6	3	3600
Dataset 2	4	50	100	20	2	2000
Dataset 3	5	50	80	20	2	1600
Dataset 4	6	50	50	10	2	500
Dataset 5	10	200	200	10	5	2000
Dataset 6	12	200	200	10	4	2000
Dataset 7	14	200	200	10	3	2000
Dataset 8	16	200	200	10	2	2000

Table 4: Parameter settings for simulating our datasets.

For kNN, we use the Kähler distance [55] given by

$$d_{BT}^2(w_1, w_2) = q \left\| \log \left(p_{0,1}^{-\frac{1}{2}} p_{0,2} p_{0,1}^{-\frac{1}{2}} \right) \right\|^2 + \sum_{j=1}^{q-1} \frac{q-j}{4} \text{Tr} \left(\log^2 \left(\frac{I_m + C_j^{\frac{1}{2}}}{I_m - C_j^{\frac{1}{2}}} \right) \right),$$

where $\text{Tr}(\cdot)$ is the trace operator, $w_1 = (p_{0,1}, w_{1,1}, \dots, w_{q-1,1})$, $w_2 = (p_{0,2}, w_{1,2}, \dots, w_{q-1,2}) \in \mathbb{H}_m^+ \times \mathbb{SD}_m^{q-1}$, and $C_j, j = 1, \dots, q-1$ are given by

$$C_j = (w_{j,2} - w_{j,1})(I_m - w_{j,1}^H w_{j,2})^{-1} (w_{j,2}^H - w_{j,1}^H)(I_m - w_{j,1} w_{j,2}^H)^{-1}.$$

The best value of k in kNN is found by using 10-fold cross-validation with the training set.

To parameterize a time series as $(p_0, w_1, \dots, w_{q-1}) \in \mathbb{H}_m^+ \times \mathbb{SD}_m^{q-1}$, where $p_0 \in \mathbb{H}_m^+$ and $w_1, \dots, w_{q-1} \in \mathbb{SD}_m$, we use Algorithm 1 proposed in [10].

Algorithm 1: Parameterize a time series as a point in $\mathbb{H}_m^+ \times \mathbb{SD}_m^{q-1}$

Input: A vector sequence $u_0, \dots, u_{p-1} \in \mathbb{C}^m$

- 1 $f_{0,k} = b_{0,k} = u_k, k = 0, \dots, p-1$
- 2 $p_0 = \frac{1}{p} \sum_{k=0}^{p-1} u_k u_k^H$
- 3 **for** $i \leftarrow 1$ **to** $q-1$ **do**
- 4 $r_{i-1}^f = \sum_{k=i}^{p-1} f_{i-1,k} f_{i-1,k}^H$
- 5 $r_{i-1}^b = \sum_{k=i}^{p-1} b_{i-1,k-1} b_{i-1,k-1}^H$
- 6 $r_{i-1}^{fb} = \sum_{k=i}^{p-1} f_{i-1,k} b_{i-1,k-1}^H$
- 7 $w_i = -(r_{i-1}^f)^{-\frac{1}{2}} r_{i-1}^{fb} \left((r_{i-1}^b)^{-\frac{1}{2}} \right)^H$
- 8 $\begin{cases} f_{i,k} = f_{i-1,k} + w_i b_{i-1,k-1} & k = i, \dots, p-1 \\ b_{i,k} = b_{i-1,k-1} + w_i^H f_{i-1,k} & k = i, \dots, p-1 \end{cases}$
- 9 **end**

Output: $(p_0, w_1, \dots, w_{q-1}) \in \mathbb{H}_m^+ \times \mathbb{SD}_m^{q-1}$

The input data of our networks belong to product spaces $\text{Sym}_3^+ \times \mathbb{SH}_3^2$, $\text{Sym}_4^+ \times \mathbb{SH}_4$, $\text{Sym}_5^+ \times \mathbb{SH}_5$, and $\text{Sym}_6^+ \times \mathbb{SH}_6$ on datasets 1, 2, 3, and 4, respectively.

To convert a real matrix u to a symmetric matrix, we set $u = u + u^T$. To convert a matrix $v \in \text{Sym}_m$ to an SPD matrix, we use a method similar to [21, 28]. We compute an eigendecomposition of v as $v = kdk^{-1}$ where $k \in O_m$ and d is a $m \times m$ diagonal matrix, and set $v = kd_\epsilon k^{-1}$ where

$$(d_\epsilon)_{ii} = \begin{cases} d_{ii} & \text{if } d_{ii} > \epsilon \\ \epsilon & \text{otherwise} \end{cases}$$

In our experiments, the value of ϵ is set to $1e-4$. We apply the same method to the real and imaginary parts of the coordinates z_1, \dots, z_{q-1} .

Method	Dataset 1 (3, 600, 3600)	Dataset 2 (4, 100, 2000)	Dataset 3 (5, 80, 1600)	Dataset 4 (6, 50, 500)
SiegelNet-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ (Ours)	72.16 \pm 0.11	91.05 \pm 0.10	85.12 \pm 0.18	75.64 \pm 0.05
SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ (Ours)	80.94\pm0.14	96.50\pm0.12	91.00\pm0.18	85.60\pm0.06

Table 5: Effectiveness of the AFC layer on the radar clutter classification task.

The AFC (DFC) layer is performed on each coordinate z_1, \dots, z_{q-1} of the input data, while the coordinate \tilde{p}_0 remains unchanged. One can also perform an FC layer [54, 61] on this coordinate to improve performance. The parameters in the MLR layers (i.e., $a_j, p_j, j = 1, \dots, L$) of our networks belong either to Siegel spaces or to SPD manifolds:

- A parameter in \mathbb{SH}_m is represented by two matrices $u \in \text{Sym}_m$ and $v \in \text{Sym}_m^+$. To learn parameter u , we learn a triangular matrix $a \in \mathbb{R}^{m \times m}$ with $\frac{m(m+1)}{2}$ parameters, and compute u as $u = a + a^T$. To learn parameter v , we learn a triangular matrix $b \in \mathbb{R}^{m \times m}$ with $\frac{m(m+1)}{2}$ parameters and use the matrix exponential to obtain v as $v = \exp(b + b^T)$. We then use the map $\phi(\cdot)$ given in Section 2.1 to compute w_j and h_j .
- For each parameter $p_j \in \text{Sym}_m^+$, we learn three parameters $k_{j,1}, k_{j,2} \in O_m$ and $u_j \in \mathbb{R}^m$ and set $h_j = k_{j,1} \exp(\text{diag}(u_j)) k_{j,2}$, where $\text{diag}(u_j)$ is the diagonal matrix with diagonal entries u_j . To learn a parameter $k \in O_m$, we learn a triangular matrix $a \in \mathbb{R}^{m \times m}$ with $\frac{m(m+1)}{2}$ parameters and use the matrix exponential to obtain k as $k = \exp(a - a^T)$. For parameters $a_j, j = 1, \dots, L \in \text{Sym}_m^+$, since $\log(w_j w_j^T)$ are symmetric matrices, we can use the same method as above to learn these matrices instead of parameters a_j . Note that when $x_j \in \text{Sym}_m^+$, we have $g_j g_j^T = x_j$. This simplifies the computation for the term $\log(h_j^{-1} g_j g_j^T h_j^{-T})$ in the expression of the point-to-hyperplane distance.

For the MLR layer of our networks, the probability of class l is computed as

$$p(y = l|x) \propto \exp \left(\frac{\left| \sum_{j=1}^L \langle \log(h_{j,l}^{-1} g_j g_j^T h_{j,l}^{-T}), \log(w_{j,l} w_{j,l}^T) \rangle \right|}{\sqrt{\sum_{j=1}^L \left\| \log(w_{j,l} w_{j,l}^T) \right\|^2}} \right),$$

where $x = (x_1, \dots, x_L) \in X, x_j = g_j K_j \in X_j, g_j \in G_j, j = 1, \dots, L$, and $h_{j,l}, w_{j,l} \in G_j, j = 1, \dots, L$ are the parameters associated with class $l, l = 1, \dots, M$.

Our networks are implemented in the Pytorch framework. All networks are trained using cross-entropy loss and Adadelta optimizer for 2000 epochs. The learning rate is set to $1e-2$. The batch size is set to 25. Results are averaged over 10 evaluations for each model. All experiments are performed using an Intel Core i7-9700 CPU 3.00 GHz.

1.1.3 Complexity Analysis

Below we discuss the memory and time complexities of the AFC, DFC, and ProductMLR layers for one training sample and one iteration:

- The AFC layer has memory complexity $O(m(m+1))$ and time complexity $O(m^3)$.
- The DFC layer has memory complexity $O\left(\frac{m_2(2m+m_2+1)}{2}\right)$ and time complexity $O(m_2 m^2 + m_2^2 m)$.
- The ProductMLR layer has memory complexity $O\left(\frac{m(3m+5)}{2} + \frac{4(q-1)m(m+1)}{2}\right)$ and time complexity $O(Mqm^3)$.

1.1.4 Ablation Study and More Results

Tab. 5 shows the effectiveness of the AFC layer on radar clutter classification. In terms of mean accuracy, SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ outperforms SiegelNet-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ by 8.78%,

Method	Dataset 1 (3, 600, 3600)	Dataset 2 (4, 100, 2000)	Dataset 3 (5, 80, 1600)	Dataset 4 (6, 50, 500)
SiegelNet-SPD-QMLR $_{\text{Sym}_m^+}$	64.02 \pm 0.12	42.18 \pm 0.11	45.23 \pm 0.16	66.41 \pm 0.06
SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$	80.94\pm0.14	96.50\pm0.12	91.00\pm0.18	85.60\pm0.06

Table 6: Impact of Siegel features on radar clutter classification.

Method	Dataset 5 (10, 200, 2000) $\text{Sym}_{10}^+ \times \mathbb{SH}_{10}^4$	Dataset 6 (12, 200, 2000) $\text{Sym}_{12}^+ \times \mathbb{SH}_{12}^3$	Dataset 7 (14, 200, 2000) $\text{Sym}_{14}^+ \times \mathbb{SH}_{14}^2$	Dataset 8 (16, 200, 2000) $\text{Sym}_{16}^+ \times \mathbb{SH}_{16}$
kNN [11]	88.00 \pm 0.0	73.40 \pm 0.0	85.40 \pm 0.0	87.10 \pm 0.0
SPDNet [21]	21.60 \pm 0.14	27.74 \pm 0.12	58.42 \pm 0.13	80.12 \pm 0.16
SPDNetBN [9]	21.95 \pm 0.12	28.82 \pm 0.09	59.95 \pm 0.09	80.75 \pm 0.11
MLR-AI [31]	25.84 \pm 0.16	32.01 \pm 0.11	65.18 \pm 0.15	84.12 \pm 0.14
GyroSpd++ [32]	29.16 \pm 0.17	30.18 \pm 0.15	62.14 \pm 0.16	85.28 \pm 0.18
SiegelNet-DFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ (Ours)	33.94 \pm 0.16	43.35 \pm 0.12	68.32 \pm 0.11	86.34 \pm 0.14
SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ (Ours)	93.72\pm0.13	79.61\pm0.10	90.06\pm0.10	94.36\pm0.12

Table 7: Results (mean accuracy \pm standard deviation) computed over 10 runs for radar clutter classification. The tuple (m, M, s) below each dataset indicates the signal dimension m , the number of classes M , and the size of the dataset s . The product space to which input data of our networks belong is also shown for each dataset.

5.45%, 5.87%, and 9.95% on datasets 1, 2, 3, and 4, respectively. These results demonstrate that the AFC layer brings significant improvements in mean accuracy.

To investigate the impact of features on Siegel spaces encoded by the coordinates z_1, \dots, z_{q-1} , we remove them from the input data and evaluate SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$. The resulting network consists of a MLR layer built on Sym_m^+ using the point-to-hyperplane distance in Theorem 3.8. Results in Tab. 6 clearly show that Siegel features have significant impact on the performance of SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$.

Tab. 7 reports results of our networks for high-dimensional time series. In this experiment, we fix the numbers of classes and of time series in each dataset while varying the order of the AR model for simulating the data. The sizes of the parameter b in the DFC layer are set to 10×6 , 12×8 , 14×9 , and 16×10 on datasets 5, 6, 7, and 8, respectively. It can be observed that SiegelNet-AFC-QMLR $_{\text{Sym}_m^+ \times \mathbb{SH}_m^{q-1}}$ remains effective across different orders and signal dimensions.

1.2 Node Classification

1.2.1 Datasets and Experimental Settings

Tab. 8 shows the statistics of Glass, Iris, and Zoo datasets. We conduct 10 random evaluations, in each of which 20% of the sequences are randomly selected for training, and the rest are used for testing. Note that this experimental setting is more challenging than the one in [12].

1.2.2 Optimization and Hyperparameters

We use the same method in Appendix 1.1.2 to parameterize a point $x \in \mathbb{SH}_m$. For the LogEig classifier, each node embedding $x = u + iv$ is transformed to the following matrix:

$$\log \left(\begin{bmatrix} v + uv^{-1}u & uv^{-1} \\ v^{-1}u & v^{-1} \end{bmatrix} \right),$$

which is then fed to a linear layer for classification.

For the QMLR layer, the probability of class l is computed as

$$p(y = l|x) \propto \exp \left(\frac{|\langle \log(\phi(p_l)^{-1}\phi(x)\phi(x)^T\phi(p_l)^{-T}), \log(\phi(a_l)\phi(a_l)^T) \rangle|}{\|\log(\phi(a_l)\phi(a_l)^T)\|} \right),$$

where $p_l, a_l \in \mathbb{SH}_m$ are the parameters associated with class $l, l = 1, \dots, M$.

Dataset	# Nodes	# Features	# Classes
Glass	214	9	6
Iris	150	4	3
Zoo	101	16	7

Table 8: Statistics of the datasets for node classification.

For the VMLR layer, the probability of class l is computed as

$$p(y = l|x) \propto \exp(\langle d_\Delta(x, p_l), a_{\xi, l} \rangle),$$

where $p_l \in \mathbb{S}\mathbb{H}_m$ and $a_{\xi, l} \in \mathbb{R}^m$ are the parameters associated with class $l, l = 1, \dots, M$. In our implementation, $a_{\xi, l}, l = 1, \dots, M$ are unit vectors whose elements are sorted in descending order (see Appendix 4.4).

The computation of the vector-valued distance in the VMLR layer is detailed in Appendices 3.1.4 and 3.2.2. To compute the eigenvalues of the cross-ratio $R(x, y)$, we compute those of $R(iI_m, \phi^{-1}(x)[y])$ where $\phi(\cdot)$ is the map presented in Section 2.1. They are obtained via the Takagi factorization [47] of $\varphi(\phi^{-1}(x)[y])$ where $\varphi(\cdot)$ is the map presented in Section 2.1. Given a complex symmetric matrix $u = a + ib \in \mathbb{C}^{m \times m}$, the Takagi factorization computes a real diagonal matrix d and a complex unitary matrix k such that $u = \bar{k}dk^H$. We construct a matrix $v \in \text{Sym}_{2m}$ as

$$v = \begin{bmatrix} a & b \\ b & -a \end{bmatrix}.$$

Let $v = k'd'k'^{-1}$ be an eigendecomposition of v , where $k' = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix} \in \mathbb{R}^{2m \times 2m}$ and d' is a $2m \times 2m$ diagonal matrix (the diagonal entries are sorted in descending order). Then $k = k_2 - ik_4$ and $d = \text{diag}(d'_{mm}, \dots, d'_{2m-12m-1})$.

To compute the inverse of a complex matrix, we use the method in [51]. Given a complex matrix $u = a + ib \in \mathbb{C}^{m \times m}$, the real and imaginary parts of the inverse $v = c + id$ of u are computed as

$$c = (br + a)^{-1}, \quad d = -rc,$$

where $r = a^{-1}b$.

To compute the point-to-hyperplane distance in the BMLR layer, we need to compute the map $H : G \rightarrow \mathfrak{a}$ determined by $g_1 = k_1 \exp H(g_1)n_1$ with $g_1 \in G$, $k_1 \in K$, $n_1 \in N$, and \mathfrak{a} is the Lie algebra of A (please refer to Proposition 4.9 and Corollary 4.10 in [33]). We recall here that any connected noncompact semisimple Lie group with finite center G can be decomposed into subgroups K , A , and N , where K is maximal compact, A is maximal abelian, and N is maximal nilpotent (the decomposition in question is called Iwasawa decomposition). The map $H(\cdot)$ can

be computed by Cholesky factorization [64, 66] as follows. Let $g = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in \text{Sp}_{2m}$ and let

$g^T g = \begin{bmatrix} a_1 & b_1 \\ b_1^T & d_1 \end{bmatrix}$. Let $a_1 = q^T r q$ be the Cholesky factorization of the positive definite matrix a_1 , where q is a unit upper triangular matrix and r is a diagonal matrix with positive diagonal entries. Then $H(g) = \begin{bmatrix} \frac{1}{2} \log(r) & 0 \\ 0 & -\frac{1}{2} \log(r) \end{bmatrix}$.

Our networks are implemented in the Pytorch framework. All networks are trained using cross-entropy loss and Adadelata optimizer for 2000 epochs. The learning rate is set to $1e - 2$. We use a batch size of 25 for all the datasets. The embedding dimension is set to 6. The value of k in kNN is set to 5. Results are averaged over 10 evaluations for each model. We use a Intel Core i7-9700 CPU 3.00 GHz for all experiments.

1.2.3 Complexity Analysis

Here we provide the memory and time complexities of the QMLR and VMLR layers for one training sample and one iteration:

Dataset	HDM05			XView60		
	x-channel	y-channel	z-channel	x-channel	y-channel	z-channel
SPDNet [21]	38.21±0.34	52.66±0.36	39.25±0.26	64.91±0.48	59.26±0.40	47.36±0.41
SPDNetBN [9]	40.74±0.30	56.82±0.33	42.48±0.25	68.74±0.46	61.97±0.42	50.61±0.38
MLR-AI [31]	41.32±0.32	59.54±0.35	45.17±0.24	69.62±0.50	63.87±0.45	50.58±0.43
GyroSpd++ [32]	41.09±0.37	57.32±0.39	43.65±0.30	68.83±0.51	62.12±0.48	49.94±0.46
SiegelNet-AFC-QMLR _{Sym_m⁺ × S^H_m}	42.17±0.85	66.91±0.69	44.92±0.44	72.57±0.73	66.17±0.58	51.78±0.48

Table 9: Results (mean accuracy \pm standard deviation) computed over 5 runs for action recognition on HDM05 and NTU-60 datasets. XView60 corresponds to the cross-view setting of NTU-60.

- The QMLR layer has memory complexity $O(2Mm(m+1))$ and time complexity $O(Mm^3)$.
- The VMLR layer has memory complexity $O(Mm(m+2))$ and time complexity $O(Mm^3)$.
- The BMLR layer has memory complexity $O(Mm(m+2))$ and time complexity $O(Mm^3)$.

1.3 Human Action Recognition

1.3.1 Datasets and Experimental Settings

We conduct action recognition experiments on HDM05 [59], NTU-60 [65], and SBU Interaction [68].

HDM05 It has 2337 sequences of 3D skeleton data with 130 classes. Each frame contains the 3D coordinates of 31 body joints. We use all the action classes and follow the experimental protocol in [53] in which 2 subjects are used for training and the remaining 3 subjects are used for testing.

NTU-60 It has 56880 sequences of 3D skeleton data with 60 classes. Each frame contains the 3D coordinates of 25 or 50 body joints. We use the mutual (interaction) actions for classification (11 classes). For the cross-view experimental protocol [65], training data come from the camera views 2 and 3, and testing data come from the camera view 1.

SBU Interaction It is an interaction action dataset which contains 282 sequences in 8 action classes created from 7 subjects. Each action is performed by two subjects where each subject has 15 joints. We follow the experimental protocol based on 5-fold cross validation with the provided training/testing splits [68].

For HDM05 and NTU-60 datasets, we consider a challenging setting in which the input data contain only one of the three coordinates (channels) of human joints. For SBU dataset, we experiment with a challenging setting in which we use only the x -coordinates of the right elbow (joint 4), right hand (joint 5), left elbow (joint 7), and left hand (joint 8) of the first subject for classification. The method in Section 5.1 is used to compute the input data which belong to products of an SPD space and a Siegel space. We focus on comparisons of our networks and SPD neural networks.

1.3.2 Optimization and Hyperparameters

We use the same method in Appendix 1.1.2 for optimizing parameters. All networks are trained using cross-entropy loss and Adadelta optimizer for 2000 epochs. The learning rate is set to $1e-2$. We use a batch size of 32 for HDM05 and SBU datasets, and a batch size of 256 for NTU-60. Results are computed over 5 runs for each model. All experiments are performed using machines with an Intel Core i7-9700 CPU 3.00 GHz.

1.3.3 Results

Tab. 9 shows the results of our method and some state-of-the-art SPD neural networks on HDM05 and NTU-60 datasets. It can be seen that our method surpasses its competitors in most cases. It is also noted that SiegelNet-AFC-QMLR_{Sym_m⁺ × S^H_m} using only the x -coordinates of human joints is superior to SPDNetBN using all the joint coordinates on HDM05 dataset (62.54% [30]).

Method	kNN [11]	SPDNet [21]	SPDNet-BN [9]	MLR-AI [31]	GyroSpd++ [32]	SiegelNet-AFC-QMLR $_{\text{Sym}^+ \times \text{SH}_m}$
	46.59 \pm 5.85	39.10 \pm 3.77	43.03 \pm 4.73	45.18 \pm 4.94	46.01 \pm 4.28	56.33\pm4.78

Table 10: Results (mean accuracy \pm standard deviation) computed over 5 runs for action recognition on SBU dataset.

Tab. 10 shows the results of kNN and the competing methods in Tab. 9 on SBU dataset. The value of k in kNN is set to 7. Again, the results demonstrate the superiority of our network compared to its competitors for human action recognition in challenging settings.

1.4 Riemannian Generative Modeling

To our knowledge, we are not aware of any works in this context that target Siegel spaces. Since we do not aim to propose new generative methods on Siegel spaces, we consider SPD spaces and apply the method in [48] based on flow matching to solve the task. Note that our methods are directly applicable to SPD spaces. We use BCIC-IV-2a [46] and MAMEM-SSVEP-II [62] datasets for our experiments.

1.4.1 Datasets and Experimental Settings

BCIC-IV-2a It consists of electroencephalography (EEG) data captured from 9 subjects. The cue-based BCI paradigm consists of 4 different motor imagery tasks, namely the imagination of movement of the left hand (class 1), right hand (class 2), both feet (class 3), and tongue (class 4). Two sessions on different days are recorded for each subject. Each session is comprised of 6 runs separated by short breaks. One run consists of 48 trials (12 trials for each of the 4 possible classes), yielding a total of 288 trials per session.

MAMEM-SSVEP-II It consists of EEG data with 256 channels captured from 11 subjects executing a SSVEP-based experimental protocol. Five different frequencies (6.66, 7.50, 8.57, 10.00 and 12.00Hz) are used for the visual stimulation, and the EGI 300 Geodesic EEG System (GES 300), using a 256-channel HydroCel Geodesic Sensor Net (HCGSN) and a sampling rate of 250 Hz is used to capture the signals.

We create two datasets of SPD matrices by computing a covariance matrix from each signal of the datasets. For BCIC-IV-2a, the session 1 data of subject 1 is used as the training set whose 1/8 is used as the validation set. The session 2 data of subject 1 is used as the test set [46]. For MAMEM-SSVEP-II, the first 4 sessions of subject 1 are used as the training set whose 1/4 is used as the validation set. The fifth session of subject 1 is used as the test set [62].

1.4.2 Optimization and Hyperparameters

We use the official code³ of the method in [48]. Vector fields are parameterized as neural networks in the ambient space and are projected onto the tangent space at every point. They are normalized by the inverse of square root of the metric tensor to cancel out the effect of the metric tensor on the Riemannian norm [48]. We use the original vector field (a standard multilayer perceptron) and replace its first convolutional layer with the $\text{VMLR}_{\text{Sym}^+}$ and $\text{QMLR}_{\text{Sym}^+}$ layers, as well as the MLR-LE and MRL-AI layers [60]. This is inspired by [58] which uses a Poincaré MLR layer as the first layer (called geodesic distance layer) of the vector field to inform it about the geometry of the considered manifold. The vector-valued distance is computed as in [57]. We use 1000 Euler steps with a projection after every step for evaluation. For likelihood computation, we use the checkpoint that gives the best negative log-likelihood (NLL) on the validation set to compute the NLL on the test set [48]. The number of hidden units and the number of layers of the vector field are set to 32 and 4, respectively. All networks are trained using AdamW optimizer with model exponential moving average. The learning rate and weight decay are set to $2e-4$ and 0.999, respectively. The number of epochs and the batch size are set to 100 and 128, respectively. Results are computed over 5 runs for each model. All experiments are performed using machines with an Intel Core i7-9700 CPU 3.00 GHz.

³<https://github.com/facebookresearch/riemannian-fm>.

Method	BCIC-IV-2a 22×22	MAMEM-SSVEP-II 8×8
RFM [48]	-225.23 \pm 3.42	-169.28 \pm 1.24
RFM-MLR-LE [31]	-231.64 \pm 6.19	-170.37 \pm 3.18
RFM-MLR-AI [31]	-238.59 \pm 6.38	-170.92 \pm 4.05
RFM-QMLR $_{\text{Sym}_m^+}$ (Ours)	-252.94 \pm 8.53	-173.06\pm7.34
RFM-VMLR $_{\text{Sym}_m^+}$ (Ours)	-260.81\pm6.55	-172.53 \pm 3.62

Table 11: Test NLL (mean \pm standard deviation) computed over 5 runs for generative modeling on SPD manifolds. The number of function evaluations is set to 1000 for all the models. The size of SPD matrices in each dataset is shown below the dataset.

1.4.3 Results

Tab. 11 reports estimates of NLL on the two datasets. It can be observed that our proposed MLR layers significantly improve RFM. We also see that these layers achieve the best performance on both the datasets.

2 Discussion

2.1 Extension of Our Approach

While our work focuses on Siegel spaces, it can be extended to other families of Riemannian manifolds, e.g., Grassmann manifolds. The Grassmann manifold $\text{Gr}_{m,p}$ [43, 44, 50] is defined as the set of all p -dimensional subspaces of the Euclidean space \mathbb{R}^m , i.e.,

$$\text{Gr}_{m,p} = \{\mathcal{U} \in \mathbb{R}^m | \mathcal{U} \text{ is a subspace, } \dim(\mathcal{U}) = p\}.$$

This set can be identified with the set of orthogonal rank- p projectors

$$\text{Gr}_{m,p} = \{x \in \mathbb{R}^{m \times m} | x = x^T, x^2 = x, \text{rank}(x) = p\}.$$

Grassmann manifolds belong to symmetric spaces of compact type and have been encountered in many machine learning applications [23]. Using the quotient structure [44] of Grassmann manifolds $\text{Gr}_{m,p} \cong \text{St}_{m,p} / \text{O}_p$, our method in Section 3.1 can be seamlessly applied to these manifolds.

2.2 Social Impact

Our work aims to advance the field of geometric deep learning. We do not expect any negative societal impact of our work. It leverages the representation power of Siegel spaces which appear in many fields such as graph learning [28], radar signal processing [3, 4, 9, 10], information fusion [67], and theoretical physics [63]. Thus it can benefit applications in these fields by making them more effective and efficient.

3 Definitions and Basic Facts

In this section, we recap several definitions and basic facts related to RSS. For greater mathematical detail and in-depth discussion, we refer the interested reader to [2, 7, 19].

3.1 Symmetric Spaces of Noncompact Type

Definition 3.1. A (globally) symmetric space X is a connected Riemannian manifold with an isometry $\sigma_x : X \rightarrow X$ for every point $x \in X$ such that $\sigma_x(x) = x$ and the differential of σ_x at x is direction-reversing, i.e., $D_x \sigma_x = -\text{id}_{T_x X}$ where $T_x X$ denotes the tangent space of X at x .

Products of symmetric spaces are symmetric spaces. Any irreducible symmetric space X admits a decomposition

$$X \cong \mathbb{E}^m \times X_+ \times X_-,$$

where \mathbb{E}^m is of Euclidean type, X_+ is of compact type, and X_- is of noncompact type. Symmetric spaces of Euclidean type are flat and are isometric to some \mathbb{R}^m . A symmetric space of compact type has non-vanishing sectional curvature ≥ 0 and is compact. A symmetric space of noncompact type has non-vanishing sectional curvature ≤ 0 and is noncompact.

There is a correspondence between symmetric spaces of noncompact type and connected noncompact semisimple Lie groups with finite center, given as follows. Let G be a connected noncompact semisimple Lie group with finite center, and let K be a maximal compact subgroup of G . Then the space X defined by

$$X := G/K := \{x = gK | g \in G\},$$

when endowed with a G -invariant Riemannian metric, forms a symmetric space of noncompact type.

In the following, we denote by X a symmetric space of noncompact type.

3.1.1 The Killing Form

The Killing form is an important and natural bilinear form on a Lie algebra. To define the Killing form, we need to define the adjoint representations of a Lie group and of its Lie algebra.

Definition 3.2. Let G be a connected Lie group and let \mathfrak{g} be its Lie algebra. Let $h \in G$ and let $I(h)$ be the map defined by

$$\begin{aligned} I(h) : G &\rightarrow G, \\ g &\mapsto hgh^{-1}, \end{aligned}$$

which is an isomorphism of Lie groups. Denote by $\mathrm{GL}(\mathfrak{g})$ the Lie group of all bijective linear maps on \mathfrak{g} , by $\mathrm{Ad}(h)$ the differential $D_e(I(h))$ of $I(h)$ at the identity $e \in G$. The adjoint representation of G is defined by the following map:

$$\begin{aligned} \mathrm{Ad}_G : G &\rightarrow \mathrm{GL}(\mathfrak{g}), \\ h &\mapsto \mathrm{Ad}(h). \end{aligned}$$

The map $\mathrm{Ad}_G(\cdot)$ is a Lie group homomorphism. Let $\mathfrak{gl}(\mathfrak{g})$ be the Lie algebra of $\mathrm{GL}(\mathfrak{g})$. The adjoint representation $\mathrm{ad} : \mathfrak{g} \rightarrow \mathfrak{gl}(\mathfrak{g})$ of \mathfrak{g} is defined as the differential of the map $\mathrm{Ad}_G(\cdot)$ at the identity $e \in G$.

Definition 3.3. The Killing form B of a Lie algebra \mathfrak{g} is the symmetric bilinear form on \mathfrak{g} defined by

$$\begin{aligned} B : \mathfrak{g} \times \mathfrak{g} &\rightarrow \mathbb{R}, \\ (x, y) &\mapsto \mathrm{Tr}(\mathrm{ad}(x) \circ \mathrm{ad}(y)). \end{aligned}$$

The Killing form has the following properties [19].

Proposition 3.4. For any Lie algebra automorphism $\nu : \mathfrak{g} \rightarrow \mathfrak{g}$ we have

$$B(\nu(x), \nu(y)) = B(x, y),$$

where $x, y \in \mathfrak{g}$.

3.1.2 Weyl Group

A Cartan decomposition of a noncompact semisimple Lie algebra \mathfrak{g} is a vector space direct sum decomposition $\mathfrak{g} = \mathfrak{l} \oplus \mathfrak{p}$ such that the Killing form of \mathfrak{g} is negative definite on \mathfrak{l} and positive definite on \mathfrak{p} . Let \mathfrak{a} be a maximal abelian subspace of \mathfrak{p} .

Denote by M the centralizer, and by M' the normalizer of \mathfrak{a} in K , i.e.

$$\begin{aligned} M &= \{k \in K : \mathrm{Ad}(k)h = h \text{ for all } h \in \mathfrak{a}\}, \\ M' &= \{k \in K : \mathrm{Ad}(k)h \in \mathfrak{a} \text{ for all } h \in \mathfrak{a}\} \end{aligned}$$

Definition 3.5. The Weyl group of X is the factor group M'/M .

For the real symplectic group $G := \mathrm{Sp}_{2m}$, the Lie algebra \mathfrak{a} , the centralizer, and the normalizer of \mathfrak{a} in K are given by

$$\mathfrak{a} = \left\{ \begin{bmatrix} a & \mathbf{0} \\ \mathbf{0} & -a \end{bmatrix} : a = \mathrm{diag}(a_1, \dots, a_m), a_1, \dots, a_m \in \mathbb{R} \right\},$$

$$M = \left\{ \begin{bmatrix} a & \mathbf{0} \\ \mathbf{0} & a \end{bmatrix} : a = \text{diag}(a_1, \dots, a_m), a_1, \dots, a_m = \pm 1 \right\},$$

$$M' = \left\{ \begin{bmatrix} a & b \\ -b & a \end{bmatrix} : a + b \text{ is a matrix with each row or column having exactly one non-zero entry of } \pm 1 \right\}.$$

Thus the Weyl group $W = M'/M$ contains all permutations of entries of a in $\begin{bmatrix} a & \mathbf{0} \\ \mathbf{0} & -a \end{bmatrix} \in \mathfrak{a}$ as well as all possible changes of sign.

3.1.3 Weyl Chamber

Weyl chambers play an important role in the description of the geometry of X and that of the boundary at infinity ∂X of X . They can be defined in \mathfrak{a} , X , and ∂X as follows.

Definition 3.6. Let $x \in \mathfrak{p}$ and let $C_{\mathfrak{g}}(x) = \{y \in \mathfrak{g} : [y, x] = 0\}$ be the centralizer of x in \mathfrak{g} , where $[\cdot, \cdot]$ denotes the Lie bracket. The vector x is called regular if $C_{\mathfrak{g}}(x) \cap \mathfrak{p}$ is maximal abelian, and singular otherwise. An open Weyl chamber \mathfrak{a}^+ in \mathfrak{a} is a connected component of the set of regular vectors in \mathfrak{a} . An open Weyl chamber in X is a set of the form $g \exp(\mathfrak{a}^+)[o]$, where $g \in G$. A Weyl chamber at infinity is a subset of ∂X and is the boundary at infinity of the closure of a Weyl chamber in X .

Note that the domain of the vector-valued distance function used for constructing the point-to-hyperplane distance in Section 3.2 is a Weyl chamber in \mathfrak{a} .

3.1.4 Vector-valued Distance

A general procedure for computing the vector-valued distance on a symmetric space is given in Algorithm 2 which follows the construction in [24, 25]. Detailed steps for computing the vector-valued distance on Siegel spaces are discussed in Appendix 3.2.2.

Algorithm 2: Compute the vector-valued distance on a symmetric space

Input: $x, y \in X$, the model flat F_{mod} , the Weyl chamber Δ .

- 1 Compute the projections of x, y into F_{mod} , i.e., find $g \in G$ such that $a = g[x]$, $b = g[y]$ and $a, b \in F_{\text{mod}}$.
- 2 Compute the translation t from a to b .
- 3 Identify the Weyl group element w such that $w[t] = v \in \Delta$.

Output: The vector-valued distance $v \in \Delta$.

3.2 Siegel Spaces

3.2.1 Riemannian Metric

Consider the cross-ratio $R(\cdot, \cdot)$ (see Section 2.1) given as

$$R(x, y) = (x - y)(x - \bar{y})^{-1}(\bar{x} - \bar{y})(\bar{x} - y)^{-1},$$

where $x, y \in \mathbb{S}\mathbb{H}_m$. If $R(x, y)$ is viewed as a function of y , then the second differential of $R(x, y)$ at the point $y = x$ is given by

$$d^2 R(x, y) = \frac{1}{2} dx v^{-1} d\bar{x} v^{-1},$$

where $x = u + iv$. Hence

$$d^2 \text{Tr}(R(x, y)) = \text{Tr}(d^2 R(x, y)) = \frac{1}{2} \text{Tr}(v^{-1} dx v^{-1} d\bar{x}),$$

where the first equality is due to the commutativity of the differential and trace operators.

One can show that for any $g \in \text{Sp}_{2m}$, $R(x, y)$ and $R(g[x], g[y])$ have the same eigenvalues [37]. Thus $\text{Tr}(R(x, y)) = \text{Tr}(R(g[x], g[y]))$ and therefore $\text{Tr}(v^{-1} dx v^{-1} d\bar{x})$ is invariant under the group action of Sp_{2m} on $\mathbb{S}\mathbb{H}_m$. This motivates the use of the differential form in Eq. (1) as the Riemannian metric for the Siegel upper space model.

3.2.2 Vector-valued Distance on Siegel Spaces

Given two points $x, y \in \mathbb{SH}_m$, the computation of the vector-valued distance between x and y follows the general procedure given in Algorithm 2 and is detailed below.

Identify a maximal flat of \mathbb{SH}_m A maximal abelian subspace of \mathfrak{p} is given by

$$\mathfrak{a} = \left\{ \begin{bmatrix} a & \mathbf{0} \\ \mathbf{0} & -a \end{bmatrix} : a = \text{diag}(a_1, \dots, a_m), a_1, \dots, a_m \in \mathbb{R} \right\}.$$

Thus the model flat F_{mod} of \mathbb{SH}_m can be given by

$$F_{\text{mod}} = \{g[iI_m] : g \in \exp(\mathfrak{a})\} = \{i \text{diag}(a_1^2, \dots, a_m^2) : a_i \neq 0\}.$$

Compute the projections of two points on F_{mod} It has been shown [37] that there exists a symplectic transformation mapping x and y into iI_m and iT where $T = \text{diag}(t_1, \dots, t_m)$, $t_k = \frac{1+\sqrt{r_k}}{1-\sqrt{r_k}}$, r_1, \dots, r_m are the eigenvalues of the cross-ratio $R(x, y)$. Thus the projections of x and y on the model flat F_{mod} are iI_m and iT , respectively.

Compute the translation between the projections The translation between iI_m and iT is determined by viewing I_m and T as points on Sym_m^+ . Thus the vector-valued distance is given by $[v_1, \dots, v_m]$, where (v_1, \dots, v_m) is a permutation of $(\log(t_1), \dots, \log(t_m))$ and $v_1 \geq \dots \geq v_m$.

3.3 SPD Manifolds

The SPD manifold Sym_m^+ is defined as

$$\text{Sym}_m^+ = \{x \in \text{Sym}_m : v^T x v > 0 \text{ for all } v \in \mathbb{R}^m, v \neq 0\}.$$

Quotient Structure Let GL_m be the general linear group. The group GL_m acts transitively on Sym_m^+ as follows: For all $g \in \text{GL}_m$ and all $x \in \text{Sym}_m^+$,

$$g[x] = gxg^T.$$

The stabilizer of $I_m \in \text{Sym}_m^+$ is O_m . Thus Sym_m^+ can be identified as

$$\text{Sym}_m^+ \cong \text{GL}_m / O_m.$$

Riemannian Distance The Riemannian distance induced by the affine-invariant metric [35] between two points $x, y \in \text{Sym}_m^+$ is given by

$$d(x, y) = \|\log(x^{-\frac{1}{2}}yx^{-\frac{1}{2}})\|.$$

3.4 Angles

Here we review different notions of angle used in our paper and some proofs of our theoretical results.

Definition 3.7 (Riemannian angles [7]). Let $\delta(t)$ and $\delta'(t)$ be two continuously differentiable curves such that $\delta(0) = \delta'(0)$, then the Riemannian angle between them at $\delta(0)$ is the unique $\alpha \in [0, \pi]$ such that

$$\cos \alpha = \frac{\langle u, u' \rangle_{\delta(0)}}{\|u\|_{\delta(0)} \|u'\|_{\delta(0)}},$$

where u and u' are the velocity vectors at time $t = 0$ of $\delta(t)$ and $\delta'(t)$, respectively, $\langle \cdot, \cdot \rangle_{\delta(0)}$ denotes the Riemannian metric computed at $\delta(0)$, and $\|\cdot\|_{\delta(0)}$ denotes the norm induced by the Riemannian metric.

Definition 3.8 (Gyroangles [31]). Let X be a symmetric space, and let \oplus and \ominus be the binary and inverse operations defined on X . Let p, q , and r be three distinct points on X . The gyrocosine of the measure of the gyroangle $\alpha \in [0, \pi]$ between $\ominus p \oplus q$ and $\ominus p \oplus r$ is given by the equation

$$\cos \alpha = \frac{\langle \ominus p \oplus q, \ominus p \oplus r \rangle}{\|\ominus p \oplus q\| \|\ominus p \oplus r\|}.$$

The gyroangle α is denoted by $\alpha = \angle qpr$.

Definition 3.9 (Comparison angles [7]). A comparison triangle in E^2 for a triple of points (p, q, r) in X is a triangle in the Euclidean plane with vertices $\bar{p}, \bar{q}, \bar{r}$ such that $d(p, q) = d(\bar{p}, \bar{q})$, $d(q, r) = d(\bar{q}, \bar{r})$, and $d(p, r) = d(\bar{p}, \bar{r})$. Such a triangle is unique up to isometry, and shall be denoted $\bar{\Delta}(p, q, r)$. The interior angle of $\bar{\Delta}(p, q, r)$ at \bar{p} is called the comparison angle between q and r at p and is denoted $\bar{\angle}_p(q, r)$. The comparison angle is well-defined provided q and r are both distinct from p .

Definition 3.10 (Angle between two geodesic paths [7]). Let $\delta : [0, a] \rightarrow X$ and $\delta' : [0, a'] \rightarrow X$ be two geodesic paths with $\delta(0) = \delta'(0)$. Given $t \in (0, a]$ and $t' \in (0, a']$, we consider the comparison triangle $\bar{\Delta}(\delta(0), \delta(t), \delta'(t'))$, and the comparison angle $\bar{\angle}_{\delta(0)}(\delta(t), \delta'(t'))$. The (Alexandrov) angle or the upper angle between the geodesic paths δ and δ' is the number $\angle_{\delta, \delta'} \in [0, \pi]$ defined by:

$$\angle_{\delta, \delta'} := \lim_{t, t' \rightarrow 0} \sup \bar{\angle}_{\delta(0)}(\delta(t), \delta'(t')) = \lim_{\varepsilon \rightarrow 0} \sup_{0 < t, t' < \varepsilon} \bar{\angle}_{\delta(0)}(\delta(t), \delta'(t')).$$

Definition 3.11 (Angle between a geodesic segment and a geodesic ray [7]). Let X be a symmetric space. Given $x, y \in X$ and $\xi, \xi' \in \partial X$, we shall use the symbol $\angle_x(\xi, \xi')$ to denote the angle at x between the unique geodesic rays which issue from x and lie in the classes ξ and ξ' , respectively, and we write $\angle_x(y, \xi)$ to denote the angle at x between the geodesic segment $[x, y]$ and the geodesic ray which issues from x and is in the class ξ .

Definition 3.12 (Angle between two points on the boundary at infinity [7]). Let X be a symmetric space. The angle $\angle(\xi, \xi')$ between $\xi, \xi' \in \partial X$ is defined to be:

$$\angle(\xi, \xi') = \sup_{x \in X} \angle_x(\xi, \xi').$$

4 Mathematical Proofs

4.1 Proof of Proposition 3.3

Proof. Assuming that the Riemannian metric is induced by the Killing form (see Appendix 3.1.1). To prove the first property, we need a result from [56].

Lemma 4.1. Let $\mu : G \rightarrow \mathfrak{a}^+$ be the map defined by $g = k \exp(\mu(g))k'$ where $g \in G$ and $k, k' \in K$. Let $\rho : X \rightarrow \mathfrak{a}^+$ be the map sending $x = g[o] \in X$ to $\mu(g)$, where $g \in G$. For all $x, x' \in X$,

$$\|\rho(x) - \rho(x')\| \leq d(x, x'),$$

where $\|\cdot\|$ is the Euclidean norm induced by $\langle \cdot, \cdot \rangle$.

Moreover, if $x, x' \in \exp(\mathfrak{a}^+)[o]$, then $d(x, x') = \|\rho(x) - \rho(x')\|$.

We also need to prove the following result.

Lemma 4.2. Let $g, h \in G$. If there exists some $k \in K$ such that $g = k \exp(\mu(g))k_1$ and $h = k \exp(\mu(h))k_2$ where $k_1, k_2 \in K$, then

$$\langle \log(gg^T), \log(hh^T) \rangle = 4\langle \mu(g), \mu(h) \rangle.$$

Proof. We have

$$\begin{aligned} gg^T &= k \exp(\mu(g))k_1 k_1^T \exp(\mu(g))k^T \\ &= k \exp(2\mu(g))k^T, \end{aligned}$$

and $hh^T = k \exp(2\mu(h))k^T$. Hence

$$\begin{aligned} \langle \log(gg^T), \log(hh^T) \rangle &= \langle \log(k \exp(2\mu(g))k^T), \log(k \exp(2\mu(h))k^T) \rangle \\ &= 4\langle k\mu(g)k^T, k\mu(h)k^T \rangle \\ &= 4\langle \mu(g), \mu(h) \rangle, \end{aligned}$$

where the last equality is due to the fact that K is a subgroup of the group of orthogonal matrices. \square

Let $x = gK, y = hK$ where $g, h \in G$. Since the distance $d(\cdot, \cdot)$ is G -invariant, we have

$$\begin{aligned} d(x, y) &= d(g^{-1}[x], g^{-1}[y]) \\ &= d(o, g^{-1}h[o]). \end{aligned}$$

Let $g^{-1}h = kak'$ where $a \in \exp(\overline{\mathfrak{a}^+})$, $k, k' \in K$. Then

$$\begin{aligned} d(o, g^{-1}h[o]) &= d(k^{-1}[o], k^{-1}kak'[o]) \\ &= d(o, a[o]) \end{aligned}$$

By Lemma 4.1,

$$\begin{aligned} d(o, a[o]) &= \|\rho(o) - \rho(a[o])\| \\ &= \|\mu(a)\| \\ &= \|\mu(g^{-1}h)\|. \end{aligned}$$

Note that

$$\begin{aligned} \|\ominus x \oplus y\|_{\mathbb{S}} &= \|g^{-1}hK\|_{\mathbb{S}} \\ &= \sqrt{\langle g^{-1}hK, g^{-1}hK \rangle_{\mathbb{S}}} \\ &= \sqrt{\langle \log(g^{-1}h(g^{-1}h)^T), \log(g^{-1}h(g^{-1}h)^T) \rangle}. \end{aligned}$$

By Lemma 4.2, we get

$$\begin{aligned} \|\ominus x \oplus y\|_{\mathbb{S}} &\propto \sqrt{\langle \mu(g^{-1}h), \mu(g^{-1}h) \rangle} \\ &= \|\mu(g^{-1}h)\|. \end{aligned}$$

Therefore

$$\|\ominus x \oplus y\|_{\mathbb{S}} \propto d(x, y).$$

To prove the second property, note that

$$\begin{aligned} \langle k[x], k[y] \rangle_{\mathbb{S}} &= \langle kgK, khK \rangle_{\mathbb{S}} \\ &= \langle \log(kgg^T k^T), \log(khh^T k^T) \rangle \\ &= \langle k \log(gg^T) k^T, k \log(hh^T) k^T \rangle. \end{aligned}$$

Since K is a subgroup of the group of orthogonal matrices, we have

$$\begin{aligned} \langle k[x], k[y] \rangle_{\mathbb{S}} &= \langle \log(gg^T), \log(hh^T) \rangle \\ &= \langle gK, hK \rangle_{\mathbb{S}} \\ &= \langle x, y \rangle_{\mathbb{S}}. \end{aligned}$$

□

4.2 Proof of Theorem 3.5

Proof. Let $p = hK, q = lK, a = wK \in X$ where $h, l, w \in G$, and let $\mathcal{H}_{a,p}$ be a hyperplane as given in Definition 3.4. Then

$$\langle \ominus p \oplus q, a \rangle_{\mathbb{S}} = \langle \log(h^{-1}l^T h^{-T}), \log(w^T) \rangle.$$

Thus for any $q = lK \in \mathcal{H}_{a,p}, l \in G$, we have

$$\langle \log(h^{-1}l^T h^{-T}), \log(w^T) \rangle = 0.$$

Let $x = gK \in X, g \in G$. By the definition of the point-to-hyperplane distance,

$$\bar{d}(x, \mathcal{H}_{a,p}) = \sin(\angle xp\bar{q})d(x, p),$$

where \bar{q} is the solution of the following optimization problem:

$$\begin{aligned}\bar{q} &= \arg \max_{q \in \mathcal{H}_{w,p} \setminus \{p\}} \left(\frac{\langle \ominus p \oplus q, \ominus p \oplus x \rangle_{\mathbb{S}}}{\| \ominus p \oplus q \|_{\mathbb{S}} \| \ominus p \oplus x \|_{\mathbb{S}}} \right) \\ &= \arg \max_{q \in \mathcal{H}_{w,p} \setminus \{p\}} \left(\frac{\langle h^{-1}lK, h^{-1}gK \rangle_{\mathbb{S}}}{\| h^{-1}lK \|_{\mathbb{S}} \| h^{-1}gK \|_{\mathbb{S}}} \right) \\ &= \arg \max_{q \in \mathcal{H}_{w,p} \setminus \{p\}} \left(\frac{\langle \log(h^{-1}ll^T h^{-T}), \log(h^{-1}gg^T h^{-T}) \rangle}{\| \log(h^{-1}ll^T h^{-T}) \| \| \log(h^{-1}gg^T h^{-T}) \|} \right)\end{aligned}$$

subject to the constraint

$$\langle \log(h^{-1}ll^T h^{-T}), \log(ww^T) \rangle = 0. \quad (3)$$

This amounts to finding the minimum angle between the vector $\log(h^{-1}gg^T h^{-T})$ and the Euclidean hyperplane described by Eq. (3). This problem has a closed form solution [52] and the distance $\bar{d}(x, \mathcal{H}_{a,p})$ can be obtained as

$$\bar{d}(x, \mathcal{H}_{a,p}) = \frac{|\langle \log(h^{-1}gg^T h^{-T}), \log(ww^T) \rangle|}{\| \log(ww^T) \|}.$$

Note that $h = \phi(p)$, $g = \phi(x)$, and $w = \phi(a)$. Hence

$$\bar{d}(x, \mathcal{H}_{a,p}) = \frac{|\langle \log(\phi(p)^{-1}\phi(x)\phi(x)^T\phi(p)^{-T}), \log(\phi(a)\phi(a)^T) \rangle|}{\| \log(\phi(a)\phi(a)^T) \|}.$$

□

4.3 Proof of Theorem 3.8

Proof. Let $q = (q_1, \dots, q_L)$, and let $q_j = l_j K_j$, $l_j \in G_j$, $j = 1, \dots, L$. Then

$$\begin{aligned}\langle \ominus p \oplus q, a \rangle_{\mathbb{S}} &= \sum_{j=1}^L \langle \ominus p_j \oplus q_j, a_j \rangle_{\mathbb{S}} \\ &= \sum_{j=1}^L \langle h_j^{-1}l_j K_j, w_j K_j \rangle_{\mathbb{S}} \\ &= \sum_{j=1}^L \langle \log(h_j^{-1}l_j l_j^T h_j^{-T}), \log(w_j w_j^T) \rangle \\ &= \langle \mathcal{C}(\{\log(h_j^{-1}l_j l_j^T h_j^{-T})\}_{j=1}^L), \mathcal{C}(\{\log(w_j w_j^T)\}_{j=1}^L) \rangle,\end{aligned}$$

where the operation $\mathcal{C}(\{u_j\}_{j=1}^L)$ flattens u_j and concatenates the resulting vectors for given matrices u_j , $j = 1, \dots, L$. Similarly, we have

$$\begin{aligned}\langle \ominus p \oplus q, \ominus p \oplus x \rangle_{\mathbb{S}} &= \sum_{j=1}^L \langle \ominus p_j \oplus q_j, \ominus p_j \oplus x_j \rangle_{\mathbb{S}} \\ &= \sum_{j=1}^L \langle h_j^{-1}l_j K_j, h_j^{-1}g_j K_j \rangle_{\mathbb{S}} \\ &= \sum_{j=1}^L \langle \log(h_j^{-1}l_j l_j^T h_j^{-T}), \log(h_j^{-1}g_j g_j^T h_j^{-T}) \rangle \\ &= \langle \mathcal{C}(\{\log(h_j^{-1}l_j l_j^T h_j^{-T})\}_{j=1}^L), \mathcal{C}(\{\log(h_j^{-1}g_j g_j^T h_j^{-T})\}_{j=1}^L) \rangle.\end{aligned}$$

From the above equation, we deduce that

$$\| \ominus p \oplus q \|_{\mathbb{S}} = \| \mathcal{C}(\{\log(h_j^{-1}l_j l_j^T h_j^{-T})\}_{j=1}^L) \|,$$

and

$$\|\ominus p \oplus x\|_{\mathbb{S}} = \|\mathcal{C}(\{\log(h_j^{-1} g_j g_j^T h_j^{-T})\}_{j=1}^L)\|.$$

The distance $\bar{d}(x, \mathcal{H}_{a,p})$ between point x and hyperplane $\mathcal{H}_{a,p}$ is given by

$$\bar{d}(x, \mathcal{H}_{a,p}) = \sin(\angle xp\bar{q})d(x, p),$$

where \bar{q} is the solution of the following optimization problem:

$$\bar{q} = \arg \max_{q \in \mathcal{H}_{a,p} \setminus \{p\}} \left(\frac{\langle \mathcal{C}(\{\log(h_j^{-1} l_j l_j^T h_j^{-T})\}_{j=1}^L), \mathcal{C}(\{\log(h_j^{-1} g_j g_j^T h_j^{-T})\}_{j=1}^L) \rangle}{\|\mathcal{C}(\{\log(h_j^{-1} l_j l_j^T h_j^{-T})\}_{j=1}^L)\| \|\mathcal{C}(\{\log(h_j^{-1} g_j g_j^T h_j^{-T})\}_{j=1}^L)\|} \right)$$

subject to the constraint

$$\langle \mathcal{C}(\{\log(h_j^{-1} l_j l_j^T h_j^{-T})\}_{j=1}^L), \mathcal{C}(\{\log(w_j w_j^T)\}_{j=1}^L) \rangle = 0. \quad (4)$$

The above problem is equivalent to the one of finding the minimum angle between the vector $\mathcal{C}(\{\log(h_j^{-1} g_j g_j^T h_j^{-T})\}_{j=1}^L)$ and the Euclidean hyperplane described by Eq. (4). Therefore, the distance $\bar{d}(x, \mathcal{H}_{a,p})$ can be obtained as

$$\bar{d}(x, \mathcal{H}_{a,p}) = \frac{|\langle \mathcal{C}(\{\log(h_j^{-1} g_j g_j^T h_j^{-T})\}_{j=1}^L), \mathcal{C}(\{\log(w_j w_j^T)\}_{j=1}^L) \rangle|}{\|\mathcal{C}(\{\log(w_j w_j^T)\}_{j=1}^L)\|}.$$

Some simple manipulations lead to

$$\bar{d}(x, \mathcal{H}_{a,p}) = \frac{\left| \sum_{j=1}^L \langle \log(h_j^{-1} g_j g_j^T h_j^{-T}), \log(w_j w_j^T) \rangle \right|}{\sqrt{\sum_{j=1}^L \|\log(w_j w_j^T)\|^2}}.$$

□

4.4 Proof of Proposition 3.11

Proof. Following [33], we will assume that the Riemannian metric is induced by the Killing form (see Appendix 3.1.1) and $\|a_\xi\| = 1$.

Let $\xi, \xi' \in \partial X$. Then the function $(\xi, \xi') \mapsto \angle(\xi, \xi')$ (see Definition 3.12) defines a metric on ∂X called the angular metric [7]. The Tits metric on ∂X is the length metric associated to the angular metric. The Tits metric is denoted d_T . The length space $(\partial X, d_T)$ is called the Tits boundary [7] of X , and is denoted $\partial_T X$. The natural G -action on X by isometries extends to a continuous action on $\partial_T X$. Let Δ_{sph} be the boundary at infinity of Δ . Then there exists a natural identification $\partial_T X/G \cong \Delta_{sph}$ [24, 25]. One can thus define a Δ_{sph} -direction map θ as the natural projection

$$\theta : \partial_T X \rightarrow \Delta_{sph} \cong \partial_T X/G,$$

which is obtained by dividing out the G -action [24, 25].

Let $\bar{\theta} = \theta(\xi)$. Then we have

$$d(x, p) \cos \angle_x(p, \xi) \leq d(x, p) \max_{\theta(\xi') = \bar{\theta}} \cos \angle_x(p, \xi'). \quad (5)$$

Let $x = gK, g \in G$. Then for any $k \in K$, since $\theta(gk[\xi]) = \bar{\theta}$, we have

$$d(x, p) \max_{\theta(\xi') = \bar{\theta}} \cos \angle_x(p, \xi') \geq d(x, p) \max_{k \in K} \cos \angle_x(p, gk[\xi]).$$

For any $h \in G$, there exists $g' \in G$ such that $h = gg'$. Also, we have $g'[\xi] = k[\xi]$ for some $k \in K$ [49]. Thus $h[\xi] = gk[\xi]$ and we have

$$d(x, p) \max_{\theta(\xi') = \bar{\theta}} \cos \angle_x(p, \xi') \leq d(x, p) \max_{k \in K} \cos \angle_x(p, gk[\xi]).$$

We thus deduce that

$$\begin{aligned}
d(x, p) \max_{\theta(\xi')=\bar{\theta}} \cos \angle_x(p, \xi') &= d(x, p) \max_{k \in K} \cos \angle_x(p, gk[\xi]) \\
&= d(x, p) \max_{k \in K} \cos \angle_o(g^{-1}[p], k[\xi]) \\
&= \max_{k \in K} d(x, p) \cos \angle_o(g^{-1}[p], k[\xi]).
\end{aligned} \tag{6}$$

Denote by τ the natural projection $G \rightarrow G/K$, by $D_e\tau$ the differential of τ at the identity $e \in G$. Let $\delta_1(t)$ be the geodesic ray which issues from o and passes $g^{-1}[p]$. Then $\delta_1(t) = k_1 \exp(td_\Delta(x, p))K$ for some $k_1 \in K$. Since $\delta_1(t) = \exp(tk_1 d_\Delta(x, p)k_1^{-1})K$, the tangent vector of $\delta_1(t)$ at $t = 0$ is given as $D_e\tau(k_1 d_\Delta(x, p)k_1^{-1})$ [42]. Let $\delta_2(t)$ be the geodesic ray which issues from o and lies in the class $k[\xi]$. Then $\delta_2(t) = kk_2 \exp(ta_\xi)K$ for some $k_2 \in K$ and the tangent vector of $\delta_2(t)$ at $t = 0$ is given as $D_e\tau(kk_2 a_\xi k_2^{-1}k^{-1})$. By the definition of Riemannian angles (see Definition 3.7),

$$\begin{aligned}
\cos \angle_o(g^{-1}[p], k[\xi]) &= \frac{\langle D_e\tau(k_1 d_\Delta(x, p)k_1^{-1}), D_e\tau(kk_2 a_\xi k_2^{-1}k^{-1}) \rangle_o}{\|D_e\tau(k_1 d_\Delta(x, p)k_1^{-1})\|_o \|D_e\tau(kk_2 a_\xi k_2^{-1}k^{-1})\|_o} \\
&= \frac{\langle k_1 d_\Delta(x, p)k_1^{-1}, kk_2 a_\xi k_2^{-1}k^{-1} \rangle}{\|k_1 d_\Delta(x, p)k_1^{-1}\| \|kk_2 a_\xi k_2^{-1}k^{-1}\|} \\
&= \frac{\langle d_\Delta(x, p), k_1^{-1}kk_2 a_\xi k_2^{-1}k^{-1}k_1 \rangle}{\|d_\Delta(x, p)\| \|a_\xi\|} \\
&= \frac{\langle d_\Delta(x, p), k_1^{-1}kk_2 a_\xi k_2^{-1}k^{-1}k_1 \rangle}{d(x, p)}.
\end{aligned}$$

Hence

$$\begin{aligned}
\max_{k \in K} d(x, p) \cos \angle_o(g^{-1}[p], k[\xi]) &= \max_{k \in K} d(x, p) \frac{\langle d_\Delta(x, p), k_1^{-1}kk_2 a_\xi k_2^{-1}k^{-1}k_1 \rangle}{d(x, p)} \\
&= \max_{k \in K} \langle d_\Delta(x, p), k_1^{-1}kk_2 a_\xi k_2^{-1}k^{-1}k_1 \rangle \\
&= \max_{k \in K} \langle d_\Delta(x, p), ka_\xi k^{-1} \rangle.
\end{aligned}$$

Since $d_\Delta(x, p), a_\xi \in \Delta$, we have

$$\max_{k \in K} \langle d_\Delta(x, p), ka_\xi k^{-1} \rangle = \langle d_\Delta(x, p), a_\xi \rangle.$$

Therefore

$$\max_{k \in K} d(x, p) \cos \angle_o(g^{-1}[p], k[\xi]) = \langle d_\Delta(x, p), a_\xi \rangle. \tag{7}$$

Combining Eqs. (5), (6), and (7), we get

$$d(x, p) \cos \angle_x(p, \xi) \leq \langle d_\Delta(x, p), a_\xi \rangle.$$

□

References

- [42] F. Bartolucci, F. D. Mari, and M. Monti. Unitarization of the Horocyclic Radon Transform on Symmetric Spaces. *CoRR*, abs/2108.04338, 2021. [16](#)
- [43] E. Batzies, K. Hüper, L. Machado, and F. S. Leite. Geometric Mean and Geodesic Regression on Grassmannians. *Linear Algebra and its Applications*, 466:83–101, 2015. [8](#)
- [44] T. Bendokat, R. Zimmermann, and P. A. Absil. A Grassmann Manifold Handbook: Basic Geometry and Computational Aspects. *CoRR*, abs/2011.13699, 2020. [8](#)
- [45] D. A. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord. Riemannian Batch Normalization for SPD Neural Networks. In *NeurIPS*, pages 15463–15474, 2019. [1](#)
- [46] C. Brunner, R. Leeb, G. Müller-Putz, A. Schlögl, and G. Pfurtscheller. BCI Competition 2008-Graz data set A. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, 16:1–6, 2008. [7](#)

- [47] A. Bunse-Gerstner and W. B. Gragg. Singular Value Decompositions of Complex Symmetric Matrices. *Journal of Computational and Applied Mathematics*, 21(1):41–54, 1988. 5
- [48] R. T. Q. Chen and Y. Lipman. Flow Matching on General Geometries. In *ICLR*, 2024. 7, 8
- [49] P. Eberlein. *Geometry of Nonpositively Curved Manifolds*. Chicago Lectures in Mathematics. University of Chicago Press, Chicago, IL, 1996. 15
- [50] A. Edelman, T. A. Arias, and S. T. Smith. The Geometry of Algorithms with Orthogonality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998. 8
- [51] A. Falkenberg. Method to Calculate the Inverse of a Complex Matrix using Real Matrix Inversion. 2007. 5
- [52] O.-E. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic neural networks. In *NeurIPS*, pages 5350–5360, 2018. 14
- [53] M. Harandi, M. Salzmann, and R. Hartley. Dimensionality Reduction on SPD Manifolds: The Emergence of Geometry-Aware Methods. *TPAMI*, 40:48–62, 2018. 6
- [54] Z. Huang and L. V. Gool. A Riemannian Network for SPD Matrix Learning. In *AAAI*, pages 2036–2042, 2017. 1, 3
- [55] B. Jeuris and R. Vandebril. The Kähler Mean of Block-Toeplitz Matrices with Toeplitz Structured Blocks. *SIAM J. Matrix Anal. Appl.*, 37(3):1151–1175, 2016. 1, 2
- [56] F. Kassel. Proper Actions on Corank-one Reductive Homogeneous Spaces. *CoRR*, abs/0807.3980, 2009. 12
- [57] F. López, B. Pozzetti, S. Trettel, M. Strube, and A. Wienhard. Vector-valued Distance and Gyrocalculus on the Space of Symmetric Positive Definite Matrices. In *NeurIPS*, pages 18350–18366, 2021. 7
- [58] E. Mathieu and M. Nickel. Riemannian Continuous Normalizing Flows. In *NeurIPS*, 2020. 7
- [59] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation Mocap Database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007. 6
- [60] X. S. Nguyen and S. Yang. Building Neural Networks on Matrix Manifolds: A Gyrovector Space Approach. In *ICML*, pages 26031–26062, 2023. 1, 7
- [61] X. S. Nguyen, S. Yang, and A. Histace. Matrix Manifold Neural Networks++. In *ICLR*, 2024. 1, 3
- [62] S. Nikolopoulos. MAMEM EEG SSVEP Dataset II (256 channels, 11 subjects, 5 frequencies presented simultaneously). 2021. 7
- [63] T. Ohsawa. The Siegel Upper Half Space is a Marsden–Weinstein Quotient: Symplectic Reduction and Gaussian Wave Packets. *Letters in Mathematical Physics*, 105(9):1301–1320, 2015. 8
- [64] P. Sawyer. Computing the Iwasawa Decomposition of the Classical Lie Groups of Noncompact Type Using the QR Decomposition. *Linear Algebra and its Applications*, 493:573–579, 2016. 5
- [65] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In *CVPR*, pages 1010–1019, 2016. 6
- [66] T.-Y. Tam. Computing the Iwasawa Decomposition of a Symplectic Matrix by Cholesky Factorization. *Appl. Math. Lett.*, 19:1421–1424, 2006. 5
- [67] M. Tang, Y. Rong, J. Zhou, and X. R. Li. Information Geometric Approach to Multisensor Estimation Fusion. *IEEE Transactions on Signal Processing*, 67(2):279–292, 2019. 8
- [68] K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras. Two-person Interaction Detection Using Body-pose Features And Multiple Instance Learning. In *CVPRW*, pages 28–35, 2012. 6