RMLR: Extending Multinomial Logistic Regression into General Geometries

Ziheng Chen[1], **Yue Song**[2,*], **Rui Wang**[3], **Xiao-Jun Wu**[3], **Nicu Sebe**[1]
[1] University of Trento, [2] Caltech, [3] Jiangnan University
ziheng_ch@163.com, yuesong@caltech.edu

Abstract

Riemannian neural networks, which extend deep learning techniques to Riemannian spaces, have gained significant attention in machine learning. To better classify the manifold-valued features, researchers have started extending Euclidean multinomial logistic regression (MLR) into Riemannian manifolds. However, existing approaches suffer from limited applicability due to their strong reliance on specific geometric properties. This paper proposes a framework for designing Riemannian MLR over general geometries, referred to as RMLR. Our framework only requires minimal geometric properties, thus exhibiting broad applicability and enabling its use with a wide range of geometries. Specifically, we showcase our framework on the Symmetric Positive Definite (SPD) manifold and special orthogonal group $\mathrm{SO}(n)$, *i.e.,*the set of rotation matrices in $\mathbb{R}^n$. On the SPD manifold, we develop five families of SPD MLRs under five types of power-deformed metrics. On $\mathrm{SO}(n)$, we propose Lie MLR based on the popular bi-invariant metric. Extensive experiments on different Riemannian backbone networks validate the effectiveness of our framework. The code is available at https://github.com/GitZH-Chen/RMLR.

1 Introduction

In recent years, significant advancements have been achieved in Deep Neural Networks (DNNs), enabling them to effectively analyze complex patterns from various types of data, including images, videos, and speech [29, 38, 27, 66]. However, most existing models have primarily assumed the underlying data with a Euclidean structure. Recently, a growing body of research has emerged, recognizing that the latent spaces of many applications exhibit non-Euclidean geometries, such as Riemannian geometries [9]. Various frequently-encountered manifolds in machine learning have posed interesting challenges and opportunities, including special orthogonal groups $\mathrm{SO}(n)$ [67, 31], symmetric positive definite (SPD) [30, 10, 42, 73, 18, 19], Gaussian [14, 47], Grassmannian [32, 72] spherical [56], and hyperbolic manifolds [23]. These manifolds share an important Riemannian property — their Riemannian operators, including geodesics, exponential & logarithmic maps, and parallel transportation, often possess closed-form expressions. Leveraging these Riemannian operators, researchers have successfully generalized different types of DNNs into manifolds, dubbed *Riemannian neural networks*.

Although Riemannian networks demonstrated success in many applications, most approaches still rely on Euclidean spaces for classification, such as tangent spaces [30, 31, 10, 47, 69, 71, 48, 49, 37, 70, 15], ambient Euclidean spaces [68, 57, 58], or coordinate systems [12]. However, these strategies distort the intrinsic geometry of the manifold, undermining the effectiveness of Riemannian networks. Researchers have recently started directly developing Riemannian Multinomial Logistic Regression (RMLR) on manifolds. Inspired by the idea of hyperplane margin [39], Ganea et al. [23] developed a hyperbolic MLR in the Poincaré ball for Hyperbolic Neural Networks (HNNs). Motivated by HNNs, Nguyen and Yang [50] developed three kinds of gyro SPD MLRs based on three distinct gyro

*Corresponding author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

structures of the SPD manifold. In parallel, Chen et al. [16] proposed a framework for building SPD MLRs induced by the flat metrics on the SPD manifold. Nguyen et al. [51] proposed gyro MLRs for the Symmetric Positive Semi-definite (SPSD) manifold based on the product of gyro spaces. However, these classifiers often rely on specific Riemannian properties, limiting their generalizability to other geometries. For instance, the hyperbolic MLR [23] relies on the generalized law of sine, while the gyro MLRs [50, 51] rely on the gyro structures.

This paper presents a framework of RMLR over general geometries. In contrast to previous works, our framework only requires the explicit expression of the Riemannian logarithm, which is the minimal requirement in extending the Euclidean MLR into manifolds. Since this property is satisfied by many commonly encountered manifolds in machine learning, our framework can be broadly applied to various types of manifolds. Empirically, we showcase our framework on the SPD manifold and rotation matrices. On the SPD manifold, we systematically propose SPD MLRs under five families of power-deformed metrics. We also present a complete theoretical discussion on the geometric properties of these metrics. In the Lie group of $\mathrm{SO}(n)$, we propose Lie MLR based on the widely used bi-invariant metric to build the Lie MLR. Our work is the first to extend the Euclidean MLR into Lie groups. Besides, our framework incorporates several previous Riemannian MLRs, including gyro SPD MLRs in [50], SPD MLRs in [16], and gyro SPSD MLRs in [51].

Our SPD MLRs are validated on four SPD backbone networks, including SPDNet [30] on the radar and human action recognition tasks and TSMNet [37] on the electroencephalography (EEG) classification tasks for the Riemannian feedforward network, RResNet [36] on the human action recognition task for the Riemannian residual network, and SPDGCN [76] on the node classification for the Riemannian graph neural network. Our Lie MLR is validated on the classic LieNet [31] backbone for the human action recognition task. Compared with previous non-intrinsic classifiers, our MLRs achieve consistent performance gains. Especially, our SPD MLRs outperform the previous classifiers by **14.23%** on SPDNet and **13.72%** on RResNet for human action recognition, and **4.46%** on TSMNet for EEG inter-subject classification. Furthermore, our Lie MLR can improve both the training stability and performance. In summary, our **main theoretical contributions** are the following: **(a)** We develop a general framework for designing Riemannian MLR over general geometries, incorporating several previous Riemannian MLRs on different geometries. **(b)** We systematically propose 5 families of SPD MLRs based on different geometries of the SPD manifold. **(c)** We propose a novel Lie MLR for deep neural networks on $\mathrm{SO}(n)$.

Main theoretical results: We solve the Riemannian margin distance to the hyperplane in Thm. 3.2 and present our RMLR framework in Thm. 3.3. As shown in Tab. 1, our RMLR incorporates several existing MLRs on different geometries. Thm. 4.2 showcases our RMLR on the SPD manifold under five families of metrics summarized in Tab. 2. To remedy the numerical instability of BWM geometry on the SPD manifold, we also propose a backpropagation-friendly solver for the SPD MLR under BWM in App. F.2.2. Thm. 5.2 proposes the Lie MLR for the Lie group $\mathrm{SO}(n)$. Due to the page limits, we put all the proofs in App. H.

2 Preliminaries

This section provides a brief review of the basic geometries of SPD manifolds and special orthogonal groups. Detailed review and notations are left in Apps. B and B.1.

SPD manifolds: The set of $n \times n$ symmetric positive definite (SPD) matrices is an open submanifold of the Euclidean space $\mathcal{S}^n$ of symmetric matrices, referred to as the SPD manifold $\mathcal{S}^n_{++}$ [3]. There are five kinds of popular Riemannian metrics on $\mathcal{S}^n_{++}$: Affine-Invariant Metric (AIM) [52], Log-Euclidean Metric (LEM) [3], Power-Euclidean Metrics (PEM) [22], Log-Cholesky Metric (LCM) [41], and Bures-Wasserstein Metric (BWM) [5]. Note that, when power equals 1, the PEM is reduced to the Euclidean Metric (EM). Thanwerdas and Pennec [63] generalized AIM, LEM, and EM into two-parameters families of $\mathrm{O}(n)$-invariant metrics, *i.e.*,(α, β)-AIM, (α, β)-LEM, and (α, β)-EM, with $\min(\alpha, \alpha + n\beta) > 0$. We denote the metric tensor of (α, β)-AIM, (α, β)-LEM, (α, β)-EM, LCM, and BWM as $g^{(\alpha,\beta)\text{-AIM}}$, $g^{(\alpha,\beta)\text{-LEM}}$, $g^{(\alpha,\beta)\text{-EM}}$, g^{LCM}, and g^{BWM}, respectively.

Rotation matrices: The special orthogonal group $\mathrm{SO}(n)$ is the set of $n \times n$ orthogonal matrices with unit determinant, the elements of which are also referred to as rotation matrices. As shown in [25], $\mathrm{SO}(n)$ forms a Lie group. We adopt the widely used bi-invariant Riemannian metric [8].

3 Riemannian multinomial logistic regression

Inspired by [39], Ganea et al. [23], Nguyen and Yang [50], Chen et al. [16], Nguyen et al. [51] extended the Euclidean MLR into hyperbolic, SPD, and SPSD manifolds. However, these classifiers rely on specific Riemannian properties, such as the generalized law of sines, gyro structures, and flat metrics, which limits their generality. In this section, we first revisit several existing MLRs and then propose our Riemannian classifiers with minimal geometric requirements.

3.1 Revisiting existing multinomial logistic regressions

Given C classes, the Euclidean MLR computes the multinomial probability of each class:

$$\forall k \in \{1, \dots, C\}, \quad p(y = k \mid x) \propto \exp\left(\langle a_k, x\rangle - b_k\right), \tag{1}$$

where $b_k \in \mathbb{R}$, and $x, a_k \in \mathbb{R}^n \backslash \{\mathbf{0}\}$. As shown in [23], the Euclidean MLR can be reformulated by the margin distance to the hyperplane:

$$p(y = k \mid x) \propto \exp\left(\operatorname{sign}(\langle a_k, x - p_k\rangle)\|a_k\| d(x, H_{a_k, p_k})\right), \tag{2}$$

$$H_{a_k, p_k} = \{x \in \mathbb{R}^n : \langle a_k, x - p_k\rangle = 0\}, \tag{3}$$

where $\langle a_k, p_k\rangle = b_k$, and H_{a_k, p_k} is a hyperplane.

Eqs. (2) and (3) can be naturally extended into manifolds $\mathcal{M}$ by Riemannian operators:

$$p(y = k \mid S) \propto \exp\left(\operatorname{sign}(\langle \tilde{A}_k, \operatorname{Log}_{P_k}(S)\rangle_{P_k})\|\tilde{A}_k\|_{P_k} \tilde{d}(S, \tilde{H}_{\tilde{A}_k, P_k})\right), \tag{4}$$

$$\tilde{H}_{\tilde{A}_k, P_k} = \{S \in \mathcal{M} : g_{P_k}(\operatorname{Log}_{P_k} S, \tilde{A}_k) = 0\}, \tag{5}$$

where $P_k \in \mathcal{M}$, $\tilde{A}_k \in T_{P_k}\mathcal{M} \backslash \{\mathbf{0}\}$, g_{P_k} is the Riemannian metric at P_k, and Log_{P_k} is the Riemannian logarithm at P_k. The margin distance is defined as an infimum:

$$\tilde{d}(S, \tilde{H}_{\tilde{A}_k, P_k})) = \inf_{Q \in \tilde{H}_{\tilde{A}_k, P_k}} d(S, Q). \tag{6}$$

The MLRs proposed in [39, 23, 50, 16] can be viewed as different implementations of Eq. (4)-Eq. (6). To calculate the MLR in Eq. (4), one has to compute the associated Riemannian metrics, logarithmic maps, and margin distance. The associated Riemannian metrics and logarithmic maps often have closed-form expressions on the frequently-encounter manifolds in machine learning. However, the computation of the margin distance can be challenging. On the Poincaré ball of hyperbolic manifolds, the generalized law of sines simplifies the calculation of Eq. (6) [23]. However, the generalized law of sines is not universally guaranteed on other manifolds. Additionally, Chen et al. [16] developed a closed-form solution of margin distance on the SPD manifold under any metric pulled back from Euclidean spaces. For curved manifolds, solving Eq. (6) would become a non-convex optimization problem. To address this challenge, Nguyen and Yang [50] defined gyro structures on the SPD manifold and proposed a pseudo-gyrodistance to calculate the margin distance. Similarly, Nguyen et al. [51] proposed a pseudo-gyrodistance on the SPSD manifold based on the gyro product space. However, gyro structures do not necessarily exist in general geometries. *In summary, the aforementioned methods often rely on specific properties of their associated Riemannian metrics, which usually do not generalize to general geometries.*

3.2 Riemannian multinomial logistic regression

Recalling Eqs. (4) and (5), the least requirement of extending Euclidean MLR into manifolds is the well-definedness of $\operatorname{Log}_{P_k}(S)$ for each k. In this subsection, we will develop Riemannian MLR, which depends solely on the Riemannian logarithm, without additional requirements, such as gyro structures and generalized law of sines. In the following, we always assume the well-definedness of the Riemannian logarithm. We start by reformulating the Euclidean margin distance to the hyperplane from a trigonometry perspective and then present our Riemannian MLR.

As we discussed before, obtaining the margin distance of Eq. (6) could be challenging. Inspired by [50], we resort to the perspective of trigonometry to reinterpret Euclidean margin distance. In Euclidean space, the margin distance is equivalent to

$$d(x, H_{a,p})) = \sin(\angle xpy^*) d(x, p), \quad \text{with } y^* = \operatorname*{arg\,max}_{y \in H_{a,p} \backslash \{p\}} (\cos \angle xpy). \tag{7}$$

We extend Eq. (7) to manifolds by the Riemannian trigonometry and geodesic distance, the counterparts of Euclidean trigonometry and distance.

Definition 3.1 (Riemannian Margin Distance)**.** Let $\tilde{H}_{\tilde{A},P}$ be a Riemannian hyperplane defined in Eq. (5), and $S \in \mathcal{M}$. The Riemannian margin distance from S to $\tilde{H}_{\tilde{A},P}$ is defined as

$$d(S, \tilde{H}_{\tilde{A},P}) = \sin(\angle SPY^*) d(S, P), \tag{8}$$

where $d(S, P)$ is the geodesic distance, and $Y^* = \operatorname{argmax}(\cos \angle SPY)$ with $Y \in \tilde{H}_{\tilde{A},P} \backslash \{P\}$. The initial velocities of geodesics define $\cos \angle SPY$:

$$\cos \angle SPY = \frac{\langle \operatorname{Log}_P Y, \operatorname{Log}_P S \rangle_P}{\| \operatorname{Log}_P Y \|_P, \| \operatorname{Log}_P S \|_P}, \tag{9}$$

where $\langle \cdot, \cdot \rangle_P$ is the Riemannian metric at P, and $\| \cdot \|_P$ is the associated norm.

The Riemannian margin distance in Def. 3.1 has a closed-form expression.

Theorem 3.2. [↓] *The Riemannian margin distance defined in Def. 3.1 is given as*

$$d(S, \tilde{H}_{\tilde{A},P}) = \frac{|\langle \operatorname{Log}_P S, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P}. \tag{10}$$

Putting the Eq. (10) into Eq. (4), we can a closed-form expression for Riemannian MLR.

Theorem 3.3 (RMLR)**.** [↓] *Given a Riemannian manifold $\{\mathcal{M}, g\}$, the Riemannian MLR induced by g is*

$$p(y = k \mid S \in \mathcal{M}) \propto \exp\left(\langle \operatorname{Log}_{P_k} S, \tilde{A}_k \rangle_{P_k} \right), \tag{11}$$

where $P_k \in \mathcal{M}$, $\tilde{A}_k \in T_{P_k}\mathcal{M} \backslash \{\mathbf{0}\}$, and Log *is the Riemannian logarithm.*

$\tilde{A}_k$ in Eq. (11) can not be directly viewed as a Euclidean parameter, as $\tilde{A}_k \in T_{P_k}\mathcal{M}$ depends on P_k and P_k varies during the training. However, the tangent vector $\tilde{A}_k$ can be generated from a tangent space at a fixed point. Several tricks can be used, such as Riemannian parallel transportation [21], vector transportation [1], the differential of Lie group or gyrogroup translation [64, 65]. Following previous work [23, 16, 50], we focus on parallel transportation and Lie group translation:

$$\tilde{A}_k = \Gamma_{Q \to P_k} A_k, \tag{12}$$

$$\tilde{A}_k = L_{P_k \odot Q_\odot^{-1} *, Q} A_k, \tag{13}$$

where $Q \in \mathcal{M}$ is a fixed point, $A_k \in T_Q\mathcal{M} \backslash \{0\}$, Γ is the parallel transportation along geodesic connecting Q and P_k, and $L_{P_k \odot Q_\odot^{-1} *, Q}$ denotes the differential map at Q of left translation $L_{P_k \odot Q_\odot^{-1}}$ with $P_k \odot Q_\odot^{-1}$ denoting Lie group product and inverse. In this way, A_k lies in a fixed tangent space and, therefore, can be optimized by a Euclidean optimizer.

Remark 3.4. We make the following remarks w.r.t. our Riemannian MLR.

(a). The reformulation of Eq. (7) in gyro MLR [50, 51] and ours are different. Gyro MLR adopts gyro trigonometry and gyro distance to reformulate Eq. (7), while our method directly uses Riemannian trigonometry and geodesic distance.

(b). Compared with the MLRs on hyperbolic, SPD, or SPSD manifolds in [23, 50, 16, 51], our framework enjoys broader applicability, as our framework only requires the Riemannian logarithm. This property is commonly satisfied by most manifolds encountered in machine learning, such as the five metrics on SPD manifolds mentioned in Sec. 2, the invariant metric on $\mathrm{SO}(n)$ [8], and hyperbolic & spherical manifolds [11, 56]. Besides, several existing MLRs on different geometries are special cases of our Riemannian MLR, which are detailed in Tab. 1.

(c). The well-definedness of the Riemannian logarithm is a much weaker requirement compared to the existence of the gyro structure. The gyro structure not only requires the Riemannian logarithm but also implicitly requires geodesic completeness [50, Eqs. (1-2)]. For instance, on SPD manifolds, EM and BWM [63] are incomplete, undermining the well-definedness of gyro operations.

4 SPD multinomial logistic regressions

This section showcases our RMLR framework on the SPD manifold. We first systematically discuss the power-deformed geometries of SPD manifolds. Based on these metrics, we will develop five families of deformed SPD MLRs.

Table 1: Several MLRs on different geometries are special cases of our MLR.

MLR	Geometries	Requirements	Incorporated by Our MLR
Euclidean MLR (Eq. (1))	Euclidean geometry	N/A	✓(App. C)
Gyro SPD MLRs [50]	AIM, LEM & LCM on $\mathcal{S}^n_{++}$	Gyro structures	✓(Rem. 4.3)
Gyro SPSD MLRs [51]	SPSD product gyro spaces	Gyro structures	✓(App. D)
Flat SPD MLRs [16]	(α, β)-LEM & (θ)-LCM on $\mathcal{S}^n_{++}$	Pullback metrics from the Euclidean space	✓(Rem. 4.3)
Ours	General Geometries	Riemannian logarithm	N/A

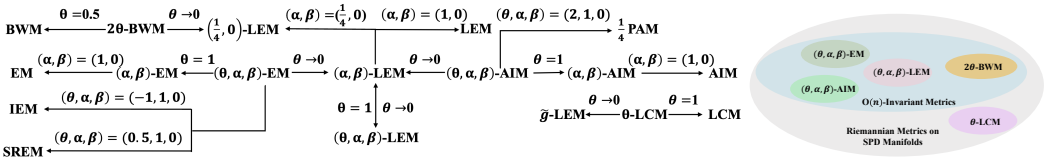

Figure 1: Illustration on the deformation (**left**) and Venn diagram (**right**) of metrics on SPD manifolds, where IEM, SREM, and $\frac{1}{4}$ PAM denotes Inverse Euclidean Metric, Square Root Euclidean Metric, and Polar Affine Metric scaled by $1/4$.

4.1 Deformed geometries of SPD manifolds

Table 2: Properties of deformed metrics on SPD manifolds ($\theta \neq 0$ and $\min(\alpha, \alpha + n\beta) > 0$).

Name	Properties
(θ, α, β)-LEM	Bi-Invariance, $\mathrm{O}(n)$-Invariance, Geodesically Completeness
(θ, α, β)-AIM	Lie Group Left-Invariance, $\mathrm{O}(n)$-Invariance, Geodesically Completeness
(θ, α, β)-EM	$\mathrm{O}(n)$-Invariance
θ-LCM	Lie Group Bi-Invariance, Geodesically Completeness
2θ-BWM	$\mathrm{O}(n)$-Invariance

As discussed in Sec. 2, there are five popular Riemannian metrics on SPD manifolds. These metrics can be all extended into power-deformed metrics. For a metric g on $\mathcal{S}^n_{++}$, the power-deformed metric is defined as

$$\tilde{g}_P(V, W) = \frac{1}{\theta^2} g_{P^\theta}\left((\phi_\theta)_{*,P}(V), (\phi_\theta)_{*,P}(W)\right), \forall P \in \mathcal{S}^n_{++}, V, W \in T_P\mathcal{S}^n_{++}, \tag{14}$$

where $\phi_\theta(P) = P^\theta$ is the matrix power, and $(\phi_\theta)_{*,P}$ is the differential map. The deformed metric $\tilde{g}$ can interpolate between a LEM-like metric ($\theta \to 0$) and g ($\theta = 1$) [61]. Previous work has extended (α, β)-AIM, (α, β)-LEM, LCM, and BWM into power-deformed metrics and (α, β)-LEM is proven to be invariant under the power deformation [17]. We denote these metrics as (θ, α, β)-AIM [59], (α, β)-LEM [17], 2θ-BWM [61], and θ-LCM [17], respectively. The deformation of these metrics is discussed in App. E.1. We further define the power-deformed metric of (α, β)-EM by Eq. (14), denoted as (θ, α, β)-EM. We have the following for the deformation of (θ, α, β)-EM.

Proposition 4.1. [↓] *(θ, α, β)-EM interpolates between (α, β)-LEM ($\theta \to 0$) and (α, β)-EM ($\theta = 1$).*

So far, all five popular Riemannian metrics on SPD manifolds have been generalized into power-deformed families of metrics. We summarize their associated properties in Tab. 2 and present their theoretical relation in Fig. 1. We leave technical details in App. E.2.

4.2 Five families of SPD multinomial logistic regressions

This subsection presents five families of specific SPD MLRs by our general framework in Thm. 3.3 and metrics discussed in Sec. 4.1. We focus on generating $\tilde{A}_k$ by parallel transportation from the identity matrix, except for 2θ-BWM. Since the parallel transportation under 2θ-BWM would undermine numerical stability (please refer to App. F.2.1 for more details), we resort to a newly

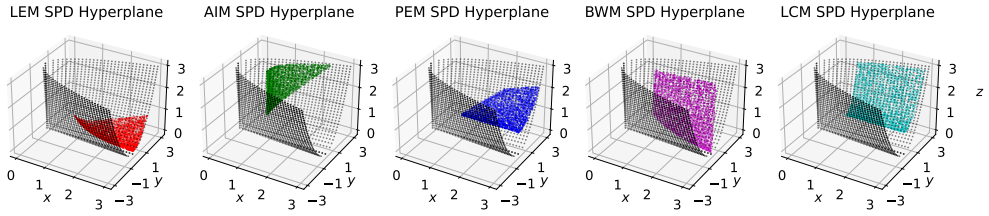

Figure 2: Conceptual illustration of SPD hyperplanes induced by five families of Riemannian metrics. The black dots denote the boundary of $\mathcal{S}_{++}^2$.

developed Lie group operation [62]:

$$S_1 \odot S_2 = L_1 S_2 L_1^T, \forall S_1, S_2 \in \mathcal{S}_{++}^n. \tag{15}$$

where $L_1 = \mathrm{Chol}(S_1)$ is the Cholesky decomposition.

Theorem 4.2 (SPD MLRs). [↓] *By abuse of notation, we omit the subscripts k of A_k and P_k. Given SPD feature S, the SPD MLRs, $p(y = k \mid S \in \mathcal{S}_{++}^n)$, are proportional to*

$$(\alpha, \beta)\text{-}\textit{LEM} : \exp\left[\langle \log(S) - \log(P), A\rangle^{(\alpha,\beta)}\right], \tag{16}$$

$$(\theta, \alpha, \beta)\text{-}\textit{AIM} : \exp\left[\frac{1}{\theta}\langle \log(P^{-\frac{\theta}{2}} S^{\theta} P^{-\frac{\theta}{2}}), A\rangle^{(\alpha,\beta)}\right], \tag{17}$$

$$(\theta, \alpha, \beta)\text{-}\textit{EM} : \exp\left[\frac{1}{\theta}\langle S^{\theta} - P^{\theta}, A\rangle^{(\alpha,\beta)}\right], \tag{18}$$

$$\theta\text{-}\textit{LCM} : \exp\left[\frac{1}{\theta}\langle \lfloor \tilde{K} \rfloor - \lfloor \tilde{L} \rfloor + \left[\mathrm{Dlog}(\mathbb{D}(\tilde{K})) - \mathrm{Dlog}(\mathbb{D}(\tilde{L}))\right], \lfloor A \rfloor + \frac{1}{2}\mathbb{D}(A)\rangle\right], \tag{19}$$

$$2\theta\text{-}\textit{BWM} : \exp\left[\frac{1}{4\theta}\langle (P^{2\theta} S^{2\theta})^{\frac{1}{2}} + (S^{2\theta} P^{2\theta})^{\frac{1}{2}} - 2P^{2\theta}, \mathcal{L}_{P^{2\theta}}(\bar{L} A \bar{L}^{\top})\rangle\right], \tag{20}$$

where $A \in T_I \mathcal{S}_{++}^n \backslash \{0\}$ is a symmetric matrix, $\log(\cdot)$ is the matrix logarithm, $\mathcal{L}_P(V)$ is the solution to the matrix linear system $\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$, known as the Lyapunov operator, $\mathrm{Dlog}(\cdot)$ is the diagonal element-wise logarithm, $\lfloor \cdot \rfloor$ is the strictly lower part of a square matrix, and $\mathbb{D}(\cdot)$ is a diagonal matrix with diagonal elements of a square matrix. Besides, $\log_{*,P}$ is the differential maps at P, $\tilde{K} = \mathrm{Chol}(S^{\theta})$, $\tilde{L} = \mathrm{Chol}(P^{\theta})$, and $\bar{L} = \mathrm{Chol}(P^{2\theta})$.

The Lyapunov operator in Eq. (20) requires the eigendecomposition. However, the backpropagation of eigendecomposition involves ${}^{1}/_{(\sigma_i - \sigma_j)}$ [34], undermining the numerical stability. Therefore, we propose a numerically stable backpropagation for the Lyapunov operator, detailed in App. F.2.2.

As 2×2 SPD matrices can be embedded into $\mathbb{R}^3$ as an open cone [74], we illustrate SPD hyperplanes induced by five families of metrics in Fig. 2.

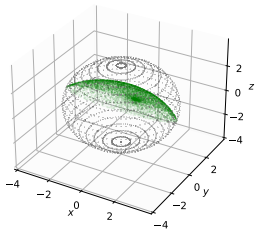

Figure 3: Conceptual illustration of a Lie hyperplane. Each pair of antipodal black dots corresponds to a rotation matrix with an Euler angle of π, while the green dots denote a Lie hyperplane.

Remark 4.3. Our SPD MLRs extend the existing SPD MLRs [50, 16]. The pseudo-gyrodistance to a SPD hyperplane in [50, Thms. 2.23-2.25] is incorporated by our Thm. 3.2, while the flat SPD MLRs under (α, β)-LEM and θ-LCM in [16, Cor. 4.1] are special cases of our Thm. 4.2. Furthermore, our approach extends the scope of prior work as neither [16] nor [50] explored SPD MLRs based on (θ, α, β)-EM and 2θ-BWM. The gyro operations in [50, Eq. (1)] implicitly requires geodesic completeness, whereas (θ, α, β)-EM and 2θ-BWM are incomplete. As neither (θ, α, β)-EM nor 2θ-BWM belong to pullback Euclidean metrics, the framework presented in [16] cannot be applied to these metrics. To the best of our knowledge, our work is the **first** to apply PEM and BWM to establish Riemannian neural networks, opening up new possibilities for utilizing these metrics in machine learning applications. Besides, neither Nguyen and Yang [50] nor Chen et al. [16] explore the deformed metrics for building SPD MLRs.

Table 3: Comparison of SPDNet with LogEig against SPD MLRs on the Radar dataset.

Architectures	LogEig MLR	(θ,α,β)-AIM	(θ,α,β)-EM		(α,β)-LEM		2θ-BWM		θ-LCM	
		(1,1,0)	(1,1,0)	(1,1,1/8)	(1,1,0)	(1,1,1)	(0.5)	(0.25)	(1)	(0.5)
2-Block	92.88±1.05	**94.53±0.95**	94.24±0.55	**94.93±0.60**	93.55±1.21	**95.64±0.83**	92.22±0.83	**94.99±0.47**	93.49±1.25	**94.59±0.82**
5-Block	93.47±0.45	**94.32±0.94**	**95.11±0.82**	95.01±0.84	94.60±0.70	**95.87±0.58**	93.69±0.66	**94.84±0.68**	93.93±0.98	**95.16±0.67**

Table 4: Comparison of SPDNet with LogEig against SPD MLRs on the HDM05 dataset.

Architectures	LogEig MLR	(θ,α,β)-AIM	(θ,α,β)-EM		(α,β)-LEM	2θ-BWM	θ-LCM	
		(1,1,0)	(1,1,0)	(0.5,1.0,1/30)	(1,1,0)	(0.5)	(1)	(0.5)
1-Block	57.42±1.31	58.07±0.64	66.32±0.63	**71.65±0.88**	56.97±0.61	**70.24±0.92**	63.84±1.31	**65.66±0.73**
2-Block	60.69±0.66	60.72±0.62	66.40±0.87	**70.56±0.39**	60.69±1.02	**70.46±0.71**	62.61±1.46	**65.79±0.63**
3-Block	60.76±0.80	61.14±0.94	66.70±1.26	**70.22±0.81**	60.28±0.91	**70.20±0.91**	62.33±2.15	**65.71±0.75**

Table 5: Inter-session experiments of TSMNet with different MLRs on the Hinss2021 dataset.

Classifiers	LogEig MLR	(θ,α,β)-AIM		(θ,α,β)-EM	(α,β)-LEM	2θ-BWM	θ-LCM	
		(1,1,0)	(0.5,1,0.05)	(1,1,0)	(1,1,0)	(0.5)	(1)	(1.5)
Balanced Acc.	53.83±9.77	53.36±9.92	**55.27±8.68**	**54.48±9.21**	53.51±10.02	**55.54±7.45**	55.71±8.57	**56.43±8.79**

Table 6: Inter-subject experiments of TSMNet with different MLRs on the Hinss2021 dataset.

Classifiers	LogEig MLR	(θ,α,β)-AIM		(θ,α,β)-EM		(α,β)-LEM	2θ-BWM		θ-LCM	
		(1,1,0)	(1.5,1,0)	(1,1,0)	(1.5,1,1/20)	(1,1,0)	(0.5)	(0.75)	(1)	(0.5)
Balanced Acc.	49.68±7.88	50.65±8.13	**51.15±7.83**	50.02±5.81	**51.38±5.77**	**51.41±7.98**	50.26±7.23	**51.67±8.73**	52.93±7.76	**54.14±8.36**

5 Lie multinomial logistic regression

This section introduces our Lie MLR on $\mathrm{SO}(n)$ based on the general RMLR framework in Thm. 3.3. The Riemannian metric on $\mathrm{SO}(n)$ is assumed to be the invariant metric in Tab. 13.

The two ways to generate $\tilde{A}_k$ in RMLR, *i.e.,*Eqs. (12) and (13), are equivalent on $\mathrm{SO}(n)$.

Lemma 5.1. [↓]

$$\Gamma_{Q\to P} = L_{PQ^{-1}*,Q}, \forall P, Q \in \mathrm{SO}(n). \tag{21}$$

Similar with SPD MLRs, we set $Q = I$. The Lie MLR on $\mathrm{SO}(n)$ is presented in the following.

Theorem 5.2. [↓] *The Lie MLR on* $\mathrm{SO}(n)$ *is given as*

$$p(y = k \mid R \in \mathrm{SO}(n)) \propto \langle \log(P_k^\top S), A_k \rangle, \tag{22}$$

where $P_k \in \mathrm{SO}(n)$ *and* $A_k \in \mathfrak{so}(n)$.

We refer to the Riemannian hyperplanes (Eq. (5)) on $\mathrm{SO}(n)$ as Lie hyperplanes. As $\mathrm{SO}(3)$ is homeomorphic to 3-dimensional real projective space $\mathbb{RP}^3$ [26], Fig. 3 illustrates Lie hyperplanes in the closed ball in $\mathbb{R}^3$ of radius π.

6 Experiments

We first validate our SPD MLRs on four SPD neural networks: SPDNet [30] and TSMNet [37] for Riemannian feedforward networks, RResNet [36] for Riemannian residual networks, and SPDGCN [76] for Riemannian graph neural networks. Then, we proceed with experiments of our Lie MLR under the classic LieNet architecture [31]. The classifier in all the above networks is the LogEig MLR (matrix logarithm + FC + softmax), a Euclidean MLR on the tangent space at the identity matrix. We substitute the original non-intrinsic LogEig MLR in each baseline model with our RMLRs. Notably, the gyro SPD MLRs [50] are special cases of our SPD MLRs under the standard AIM, LEM, and LCM ($(\theta,\alpha,\beta) = (1,1,0)$), while flat SPD MLRs [16] are incorporated by our SPD MLRs under (α,β)-LEM and θ-LCM. More implementation details are presented in App. G.

6.1 Experiments on the proposed SPD MLRs

In the following, we abbreviate *SPD MLR-metric* as *metric*. For instance, (θ,α,β)-AIM denotes the baseline endowed with the SPD MLR induced by (θ,α,β)-AIM and (1,1,0) as the value of (θ,α,β).

6.1.1 Experiments on the Riemannian feedforward network

We evaluate our SPD MLRs for Riemannian feedforward networks under the SPDNet and TSMNet backbones. Following [30, 10], on SPDNet, we use the Radar dataset [10] for radar recognition and the HDM05 dataset [44] for human action recognition. TSMNet [37] is one of the state-of-the-art methods for the EEG classification task. Following [37], we use the Hinss2021 [28] dataset. For each family of SPD MLRs, we report the SPD MLR induced from the standard metric ($\theta = 1, \alpha = 1, \beta = 0$), and the one induced from the deformed metric with best (θ, α, β). Besides, if the standard SPD MLR is already saturated, we only report the results of the standard one. Under each metric, We highlight the results in bold of our SPD MLR under the best hyperparameters. We visualize the results in App. G.1.6.

Radar: In line with [10], we evaluate our classifiers under two network architectures: 2-Block and 5-Block configurations. The 10-fold results (mean±std) are presented in Tab. 3. Note that the SPD MLR induced by standard AIM is saturated. Generally speaking, our SPD MLRs achieve superior performance against the vanilla LogEig MLR. Moreover, for most families of metrics, the associated SPD MLRs with proper (θ, α, β) outperform the standard SPD MLR, demonstrating the effectiveness of our parameterization. Besides, among all SPD MLRs, the ones induced by (α, β)-LEM achieve the best performance.

HDM05: Following [30], three architectures are adopted: 1-Block, 2-Block and 3-Block configurations. The 10-fold results (mean±std) are presented in Tab. 4. Note that the standard SPD MLRs under AIM, LEM, and BWM are already saturated on this dataset. As the Radar dataset, similar observations can be made on this dataset. Our SPD MLRs can bring consistent performance gain for SPDNet, and properly selected hyperparameters can bring further improvement. Particularly, among all the SPD MLRs, the ones based on the 2θ-BWM and (θ, α, β)-EM achieve the best performance. Compared to the vanilla LogEig MLR, **the highest performance improvement is 14.23%**, highlighting our approach's effectiveness. Notably, since 2θ-BWM and (θ, α, β)-EM are geodesically incomplete and not pulled back from a Euclidean space, the SPD MLR under these two metrics can not be derived by the framework of gyro or flat MLR. This contrast confirms the applicability of our theoretical framework to a broader range of geometries.

Hinss2021: The results (mean±std) of leaving 5% out cross-validation are reported in Tabs. 5 and 6. Once again, our intrinsic classifiers demonstrate improved performance compared to the LogEig MLR in both inter-session and inter-subject scenarios. Besides, the SPD MLRs based on θ-LCM achieve the best performance, **outperforming the vanilla classifier by 2.6% for inter-session and by 4.46% for inter-subject**. This finding highlights the versatility of our framework.

Table 7: Comparison of LogEig against SPD MLRs under the RResNet architecture.

Datasets	LogEigMLR	(θ, α, β)-AIM	(θ, α, β)-EM	(α, β)-LEM	2θ-BWM	θ-LCM
HDM05	58.17 ± 2.07	60.23 ± 1.26	**71.89 ± 0.60 (↑ 13.72)**	59.44 ± 0.87	69.85 ± 0.23	65.76 ± 0.96
NTU60	45.22 ± 1.23	48.94 ± 0.68	52.24 ± 1.25	46.99 ± 0.41	50.56 ± 0.59	**53.63 ± 0.95 (↑ 8.41)**

6.1.2 Experiments on the Riemannian residual network

Following [36], we use the HDM05 and NTU60 [55] datasets on the RResNet backbone. For the hyperparameter (θ, α, β) in our SPD MLRs, we borrow the best ones from Tab. 4. Tab. 7 reports the 10-fold and 5-fold results on the HDM05 and NTU datasets. The SPD MLRs still consistently outperform the vanilla LogEig MLR. Besides, similar to the SPD MLRs under the SPDNet backbone for action recognition (Tab. 4), the SPD MLR based on θ-LCM, 2θ-BWM, or (θ, α, β)-EM outperforms the vanilla LogEig MLR by a large margin. Especially, **the highest performance improvement is 13.72% and 8.4%** on these two datasets.

6.1.3 Experiments on the Riemannian graph network

We use SPDGCN [76] as the backbone network for the Riemannian graph network. Following [76], we use the Disease [2], Cora [54], and Pubmed [46] datasets for node classification. The 10-fold average and maximum results of the vanilla LogEig MLR against our SPD MLR with best (θ, α, β) are reported in Tab. 8. Similar to the previous results, our SPD MLRs outperform the LogEig MLR. Besides, the SPD MLR based on (α, β)-LEM generally achieves the best performance for SPDGCN.

6.1.4 Ablations of SPD MLRs on direct classification

For a more straightforward comparison, we compare LogEig against our SPD MLRs for direct classification. We adopt the Radar, HDM05, and Hinss2021 datasets. We follow the preprocessing of

Table 8: Comparison of LogEig against SPD MLRs under the SPDGCN architecture.

Classifiers	Disease		Cora		Pubmed	
	Mean±STD	Max	Mean±STD	Max	Mean±STD	Max
LogEig MLR	90.55 ± 4.83	96.85	78.04 ± 1.27	79.6	70.99 ± 5.12	77.6
(θ, α, β)-AIM	94.84 ± 2.27	98.43	79.79 ± 1.44	81.6	77.83 ± 1.08	**80**
(θ, α, β)-EM	90.87 ± 5.14	98.03	79.05 ± 1.23	81	78.16 ± 2.41	79.5
(α, β)-LEM	**96.33 ± 2.19**	**98.82**	**79.89 ± 0.99**	81.8	**78.16 ± 2.41**	79.5
2θ-BWM	91.93 ± 3.64	96.85	73.46 ± 2.18	77.7	73.22 ± 4.06	78.1
θ-LCM	93.01 ± 2.14	98.43	77.59 ± 1.20	80.1	74.46 ± 5.81	78.9

Table 9: Comparison of LogEig against SPD MLRs for direct classification.

Classifiers	Radar	HDM05	Hinss2021	
			Inster-session	Inster-subject
LogEig MLR	91.93 ± 1.30	48.43 ± 1.25	39.76 ± 7.60	44.66 ± 7.17
(θ, α, β)-AIM	95.21 ± 0.81	49.17 ± 1.08	41.14 ± 7.26	45.89 ± 6.52
(θ, α, β)-EM	92.25 ± 1.20	61.60 ± 0.69	**45.78 ± 8.51 (↑ 6.02)**	45.84 ± 4.75
(α, β)-LEM	95.09 ± 0.57	49.05 ± 0.91	40.88 ± 7.46	**46.02 ± 5.96 (↑ 1.36)**
2θ-BWM	94.89 ± 0.41	**66.77 ± 1.34 (↑ 18.34)**	44.84 ± 8.00	45.21 ± 7.44
θ-LCM	**95.67 ± 0.61 (↑ 3.74)**	58.66 ± 0.51	43.17 ± 6.21	45.10 ± 6.20

SPDNet and TSMNet to model features into the SPD manifold and directly use LogEig or our SPD MLRs for classification. The average results are presented in Tab. 9. The hyperparameters (θ, α, β) are borrowed from Tabs. 3 to 6. Our SPD MLRs consistently outperform the vanilla LogEig MLR. Particularly on the HDM05 dataset, **the highest performance improvement by our SPD MLRs is 18.34%**, surpassing the non-intrinsic LogEig MLR by a large margin. Ablations on model efficiency are also discussed in App. G.1.5.

6.2 Experiments on the proposed Lie MLR

Table 10: Results of LogEig MLR against Lie MLR under the LieNet architecture.

Classifiers	G3D		HDM05	
	Mean±STD	Max	Mean±STD	Max
LogEig MLR	87.91±0.90	89.73	76.92±1.27	79.11
Lie MLR	**89.13±1.7**	**92.12**	**78.24±1.03**	**80.25**

We apply our Lie MLR into the classic $\mathrm{SO}(n)$ network, *i.e.*,LieNet [31], where features are on the Lie group of $\mathrm{SO}(3) \times \cdots \times \mathrm{SO}(3)$. Following LieNet [31], we use G3D [6] and HDM05 [44] datasets. We also extend the Riemannian optimization package geoopt [4] into $\mathrm{SO}(3)$, allowing for Riemannian optimization. We find that Riemannian SGD performs best for LieNet. Tab. 10 presents the 10-fold average results of LieNet with or without Lie MLR. Note that on the HDM05 datasets, the LieNet might fail to converge, fluctuating between the validation accuracy of 70% - 75%. Therefore, we select 10-fold best performance out of 20-fold experiments. It can be observed that our Lie MLR can improve the performance of LieNet. Besides, our Lie MLR can also improve the training stability. On the HDM05 dataset, LieNet fails to converge in 8 out of 20 folds. However, when endowed with our Lie MLR, LieNet+LieMLR only encounters convergence failures in 2 folds.

7 Conclusions

This paper presents a novel and versatile framework for designing RMLR for general geometries, with a specific focus on SPD manifolds and $\mathrm{SO}(n)$. On the SPD manifold, we systematically explore five families of Riemannian metrics and utilize them to construct five families of deformed SPD MLRs. On $\mathrm{SO}(n)$, we develop the Lie MLR for classifying rotation matrices. Extensive experiments demonstrate the superiority of our intrinsic classifiers. We expect that our work could present a promising direction for designing intrinsic classifiers on diverse geometries.

Acknowledgments and Disclosure of Funding

This work was partly supported by the MUR PNRR project FAIR (PE00000013) funded by the NextGenerationEU, the EU Horizon project ELIAS (No. 101120237), a donation from Cisco, the National Natural Science Foundation of China (62306127), the Natural Science Foundation of Jiangsu Province (BK20231040), and the Fundamental Research Funds for the Central Universities (JUSRP124015). The authors also gratefully acknowledge the financial support from the China Scholarship Council (CSC).

References

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[2] Roy M Anderson and Robert M May. *Infectious diseases of humans: dynamics and control*. Oxford University Press, 1991.

[3] Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. *Fast and simple computations on tensors with Log-Euclidean metrics.* PhD thesis, INRIA, 2005.

[4] Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. *arXiv preprint arXiv:1810.00760*, 2018.

[5] Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the Bures-Wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2019.

[6] Victoria Bloom, Dimitrios Makris, and Vasileios Argyriou. G3D: A gaming action dataset and real time action recognition evaluation framework. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 7–12. IEEE, 2012.

[7] Silvere Bonnabel, Anne Collard, and Rodolphe Sepulchre. Rank-preserving geometric means of positive semi-definite matrices. *Linear Algebra and its Applications*, 438(8):3202–3216, 2013.

[8] Nicolas Boumal and P-A Absil. A discrete regression method on manifolds and its application to data on SO(n). *IFAC Proceedings Volumes*, 44(1):2284–2289, 2011.

[9] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34 (4):18–42, 2017.

[10] Daniel Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Riemannian batch normalization for SPD neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[11] James W Cannon, William J Floyd, Richard Kenyon, Walter R Parry, et al. Hyperbolic geometry. *Flavors of geometry*, 31(59-115):2, 1997.

[12] Rudrasis Chakraborty, Chun-Hao Yang, Xingjian Zhen, Monami Banerjee, Derek Archer, David Vaillancourt, Vikas Singh, and Baba Vemuri. A statistical recurrent model on the manifold of symmetric positive definite matrices. *Advances in Neural Information Processing Systems*, 31, 2018.

[13] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7354–7362, 2019.

[14] Ziheng Chen, Tianyang Xu, Xiao-Jun Wu, Rui Wang, and Josef Kittler. Hybrid Riemannian graph-embedding metric learning for image set classification. *IEEE Transactions on Big Data*, 2021.

[15] Ziheng Chen, Tianyang Xu, Xiao-Jun Wu, Rui Wang, Zhiwu Huang, and Josef Kittler. Riemannian local mechanism for SPD neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7104–7112, 2023.

[16] Ziheng Chen, Yue Song, Gaowen Liu, Ramana Rao Kompella, Xiaojun Wu, and Nicu Sebe. Riemannian multinomial logistics regression for SPD neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024.

[17] Ziheng Chen, Yue Song, Yunmei Liu, and Nicu Sebe. A Lie group approach to Riemannian batch normalization. In *The Twelfth International Conference on Learning Representations*, 2024.

[18] Ziheng Chen, Yue Song, Xiao-Jun Wu, Gaowen Liu, and Nicu Sebe. Understanding matrix function normalizations in covariance pooling through the lens of Riemannian geometry. *arXiv preprint arXiv:2407.10484*, 2024.

[19] Ziheng Chen, Yue Song, Xiao-Jun Wu, and Nicu Sebe. Product geometries on Cholesky manifolds with applications to SPD manifolds. *arXiv preprint arXiv:2407.02607*, 2024.

[20] Ziheng Chen, Yue Song, Tianyang Xu, Zhiwu Huang, Xiao-Jun Wu, and Nicu Sebe. Adaptive Log-Euclidean metrics for SPD matrix learning. *IEEE Transactions on Image Processing*, 2024.

[21] Manfredo Perdigao Do Carmo and J Flaherty Francis. *Riemannian Geometry*, volume 6. Springer, 1992.

[22] Ian L Dryden, Xavier Pennec, and Jean-Marc Peyrat. Power Euclidean metrics for covariance matrices with application to diffusion tensor imaging. *arXiv preprint arXiv:1009.3045*, 2010.

[23] Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[24] Alexandre Gramfort. MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, 7, 2013.

[25] Brian C Hall and Brian C Hall. *Lie groups, Lie algebras, and representations*. Springer, 2013.

[26] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International Journal of Computer Vision*, 103:267–305, 2013.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[28] Marcel F. Hinss, Ludovic Darmet, Bertille Somon, Emilie Jahanpour, Fabien Lotte, Simon Ladouce, and Raphaëlle N. Roy. An EEG dataset for cross-session mental workload estimation: Passive BCI competition of the Neuroergonomics Conference 2021, 2021.

[29] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997.

[30] Zhiwu Huang and Luc Van Gool. A Riemannian network for SPD matrix learning. In *Thirty-first AAAI Conference on Artificial Intelligence*, 2017.

[31] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. Deep learning on Lie groups for skeleton-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6099–6108, 2017.

[32] Zhiwu Huang, Jiqing Wu, and Luc Van Gool. Building deep networks on Grassmann manifolds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[33] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Matrix backpropagation for deep networks with structured layers. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2965–2973, 2015.

[34] Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. Training deep networks with structured layers by matrix backpropagation. *arXiv preprint arXiv:1509.07838*, 2015.

[35] Vinay Jayaram and Alexandre Barachant. MOABB: trustworthy algorithm benchmarking for BCIs. *Journal of Neural Engineering*, 15(6):066011, 2018.

[36] Isay Katsman, Eric Ming Chen, Sidhanth Holalkere, Anna Asch, Aaron Lou, Ser-Nam Lim, and Christopher De Sa. Riemannian residual neural networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[37] Reinmar Kobler, Jun-ichiro Hirayama, Qibin Zhao, and Motoaki Kawanabe. SPD domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG. *Advances in Neural Information Processing Systems*, 35:6219–6235, 2022.

[38] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 2012.

[39] Guy Lebanon and John Lafferty. Hyperplane margin classifiers on the multinomial manifold. In *Proceedings of the Twenty-first International Conference on Machine Learning*, page 66, 2004.

[40] John M Lee. *Introduction to smooth manifolds*. Springer, 2013.

[41] Zhenhua Lin. Riemannian geometry of symmetric positive definite matrices via Cholesky decomposition. *SIAM Journal on Matrix Analysis and Applications*, 40(4):1353–1370, 2019.

[42] Federico López, Beatrice Pozzetti, Steve Trettel, Michael Strube, and Anna Wienhard. Vector-valued distance and Gyrocalculus on the space of symmetric positive definite matrices. *Advances in Neural Information Processing Systems*, 34:18350–18366, 2021.

[43] Hà Quang Minh. Alpha Procrustes metrics between positive definite operators: a unifying formulation for the Bures-Wasserstein and Log-Euclidean/Log-Hilbert-Schmidt metrics. *Linear Algebra and its Applications*, 636:25–68, 2022.

[44] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Documentation mocap database HDM05. Technical report, Universität Bonn, 2007.

[45] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[46] Galileo Namata, Ben London, Lise Getoor, Bert Huang, and U Edu. Query-driven active surveying for collective classification. In *10th International Workshop on Mining and Learning with Graphs*, volume 8, page 1, 2012.

[47] Xuan Son Nguyen. Geomnet: A neural network based on Riemannian geometries of SPD matrix space and Cholesky space for 3D skeleton-based interaction recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 13379–13389, 2021.

[48] Xuan Son Nguyen. The Gyro-structure of some matrix manifolds. In *Advances in Neural Information Processing Systems*, volume 35, pages 26618–26630, 2022.

[49] Xuan Son Nguyen. A Gyrovector space approach for symmetric positive semi-definite matrix learning. In *Proceedings of the European Conference on Computer Vision*, pages 52–68, 2022.

[50] Xuan Son Nguyen and Shuo Yang. Building neural networks on matrix manifolds: A Gyrovector space approach. *arXiv preprint arXiv:2305.04560*, 2023.

[51] Xuan Son Nguyen, Shuo Yang, and Aymeric Histace. Matrix manifold neural networks++. In *The Twelfth International Conference on Learning Representations*, 2024.

[52] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A Riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, 2006.

[53] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3D deep learning with PyTorch3D. *arXiv preprint arXiv:2007.08501*, 2020.

[54] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[55] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+ D: A large scale dataset for 3D human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1010–1019, 2016.

[56] Ondrej Skopek, Octavian-Eugen Ganea, and Gary Bécigneul. Mixed-curvature variational autoencoders. *arXiv preprint arXiv:1911.08411*, 2019.

[57] Yue Song, Nicu Sebe, and Wei Wang. Why approximate matrix square root outperforms accurate SVD in global covariance pooling? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1115–1123, 2021.

[58] Yue Song, Nicu Sebe, and Wei Wang. On the eigenvalues of global covariance pooling for fine-grained visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3554–3566, 2022.

[59] Yann Thanwerdas and Xavier Pennec. Is affine-invariance well defined on SPD matrices? a principled continuum of metrics. In *Geometric Science of Information: 4th International Conference, GSI 2019, Toulouse, France, August 27–29, 2019, Proceedings 4*, pages 502–510. Springer, 2019.

[60] Yann Thanwerdas and Xavier Pennec. Exploration of balanced metrics on symmetric positive definite matrices. In *Geometric Science of Information: 4th International Conference, GSI 2019, Toulouse, France, August 27–29, 2019, Proceedings 4*, pages 484–493. Springer, 2019.

[61] Yann Thanwerdas and Xavier Pennec. The geometry of mixed-Euclidean metrics on symmetric positive definite matrices. *Differential Geometry and its Applications*, 81:101867, 2022.

[62] Yann Thanwerdas and Xavier Pennec. Theoretically and computationally convenient geometries on full-rank correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 43(4): 1851–1872, 2022.

[63] Yann Thanwerdas and Xavier Pennec. O (n)-invariant Riemannian metrics on SPD matrices. *Linear Algebra and its Applications*, 661:163–201, 2023.

[64] Loring W.. Tu. *An introduction to manifolds*. Springer, 2011.

[65] Abraham A Ungar. *Analytic hyperbolic geometry: Mathematical foundations and applications*. World Scientific, 2005.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[67] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3D skeletons as points in a Lie group. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–595, 2014.

[68] Qilong Wang, Jiangtao Xie, Wangmeng Zuo, Lei Zhang, and Peihua Li. Deep CNNs meet global covariance pooling: Better representation and generalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(8):2582–2597, 2020.

[69] Rui Wang, Xiao-Jun Wu, and Josef Kittler. SymNet: A simple symmetric positive definite manifold deep learning method for image set classification. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):2208–2222, 2021.

[70] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler. DreamNet: A deep Riemannian manifold network for SPD matrix learning. In *Proceedings of the Asian Conference on Computer Vision*, pages 3241–3257, 2022.

[71] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler. Learning a discriminative SPD manifold neural network for image set classification. *Neural networks*, 151:94–110, 2022.

[72] Rui Wang, Chen Hu, Ziheng Chen, Xiao-Jun Wu, and Xiaoning Song. A Grassmannian manifold self-attention network for signal classification. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 5099–5107, 2024.

[73] Rui Wang, Xiao-Jun Wu, Ziheng Chen, Cong Hu, and Josef Kittler. SPD manifold deep metric learning for image set classification. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[74] Or Yair, Mirela Ben-Chen, and Ronen Talmon. Parallel transport on the cone manifold of SPD matrices for domain adaptation. *IEEE Transactions on Signal Processing*, 67(7):1797–1811, 2019.

[75] Hongwei Yong, Jianqiang Huang, Deyu Meng, Xiansheng Hua, and Lei Zhang. Momentum batch normalization for deep learning with small batch size. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, pages 224–240. Springer, 2020.

[76] Wei Zhao, Federico Lopez, J Maxwell Riestenberg, Michael Strube, Diaaeldin Taha, and Steve Trettel. Modeling graphs beyond hyperbolic: Graph neural networks in symmetric positive definite matrices. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 122–139. Springer, 2023.

Appendix Contents

A Limitations and future avenues **17**

B Preliminaries **17**

B.1 Notations 17

B.2 Brief review of Riemannian geometry 17

B.3 Basic geometries of SPD manifolds 18

B.4 Basic geometry of rotation matrices 18

C RMLR as a natural extension of the Euclidean MLR **19**

D Gyro SPSD MLR as special cases of our RMLR **19**

E Theories on the deformed metrics **21**

E.1 Limiting cases of the deformed metrics 21

E.2 Proof of the properties of the deformed metrics (Tab. 2) 22

F Computational details on the SPD MLR under power-deformed BWM **23**

F.1 Matrix square roots in the SPD MLR under power-deformed BWM 23

F.2 Numerical stability of the SPD MLR under power-deformed BWM 23

F.2.1 Instability of parallel transportation under power-deformed BWM 23

F.2.2 Numerically stable methods for the SPD MLR under power-deformed BWM 23

G Implementation details and additional experiments **24**

G.1 Additional details and experiments on the SPD MLRs 24

G.1.1 Basic layers in SPDNet and TSMNet 24

G.1.2 Datasets and preprocessing 25

G.1.3 Implementation details 25

G.1.4 Hyper-parameters 26

G.1.5 Model efficiency 26

G.1.6 Visualization 27

G.2 Additional details and experiments on the Lie MLR 27

G.2.1 Basic layers in LieNet 27

G.2.2 Datasets and preprocessing 28

G.2.3 Implementation details 28

G.3 Hardware 28

H Proofs **28**

H.1 Proof of Thm. 3.2 28

H.2 Proof of Thm. 3.3 29

H.3 Proof of Prop. 4.1 29

H.4 Proof of Thm. 4.2 29

H.5 Proof of Lem. 5.1 32

H.6 Proof of Thm. 5.2 32

A Limitations and future avenues

Limitation: Recalling our RMLR in Eq. (11), our RMLR might be over-parameterized. In our RMLR, each class would require a Riemannian parameter P_k and Euclidean parameter A_k. Consequently, as the number of classes grows, the classification layer would become burdened with excessive parameters. We will address this problem in future work.

Future work: We highlight the advantage of our approach compared to existing methods that our framework only requires the Riemannian logarithm, which is commonly satisfied by various manifolds encountered in machine learning. Therefore, as a future avenue, our framework offers various possibilities for designing intrinsic classifiers for neural networks on other manifolds.

B Preliminaries

B.1 Notations

We briefly summarize the notations in Tab. 11 for better clarity.

Table 11: Summary of notations.

Notation	Explanation
$\{\mathcal{M}, g\}$ or abbreviated as $\mathcal{M}$	A Riemannian manifold
$T_P\mathcal{M}$	The tangent space at $P \in \mathcal{M}$
$g_P(\cdot,\cdot)$ or $\langle\cdot,\cdot\rangle_P$	The Riemannian metric at $P \in \mathcal{M}$
$\|\cdot\|_P$	The norm induced by $\langle\cdot,\cdot\rangle_P$ on $T_P\mathcal{M}$
Log_P	The Riemannian logarithm at P
$\Gamma_{P\to Q}$	The Riemannian parallel transportation along the geodesic connecting P and Q
$H_{a,p}$	The Euclidean hyperplane
$\tilde{H}_{\tilde{A},P}$	The Riemannian hyperplane
$\odot$	A Lie group operation
$\{\mathcal{M}, \odot\}$	A Lie group
$P_{\odot}^{-1}$	The group inverse of P under $\odot$
L_P	The Lie group left translation by $P \in \mathcal{M}$
$f_{*,P}$	The differential map of the smooth map f at $P \in \mathcal{M}$
f^*g	The pullback metric by f from g
$\mathcal{S}_{++}^n$	The SPD manifold
$\mathrm{SO}(n)$	The special orthogonal group
$\mathcal{S}^n$	The Euclidean space of symmetric matrices
$\mathcal{L}_+^n$	The Cholesky manifold, *i.e.*,the set of lower triangular matrices with positive diagonal elements
$\mathcal{L}^n$	The Euclidean space of $n \times n$ real lower triangular matrices
$\langle\cdot,\cdot\rangle$ or $\cdot:\cdot$	The standard Frobenius inner product
$\mathbf{ST}$	$\mathbf{ST} = \{(\alpha,\beta) \in \mathbb{R}^2 \mid \min(\alpha, \alpha+n\beta) > 0\}$
$\langle\cdot,\cdot\rangle^{(\alpha,\beta)}$	The $\mathrm{O}(n)$-invariant Euclidean inner product
$g^{(\alpha,\beta)\text{-LEM}}$	The Riemannian metric of (α,β)-LEM
$g^{(\alpha,\beta)\text{-AIM}}$	The Riemannian metric of (α,β)-AIM
$g^{(\alpha,\beta)\text{-EM}}$	The Riemannian metric of (α,β)-EM
g^{BWM}	The Riemannian metric of BWM
g^{LCM}	The Riemannian metric of LCM
$\log(\cdot)$	The matrix logarithm
$\mathrm{Chol}(\cdot)$	Cholesky decomposition
$\mathrm{Dlog}(\cdot)$	The diagonal element-wise logarithm
$\lfloor\cdot\rfloor$	The strictly lower triangular part of a square matrix
$\mathbb{D}(\cdot)$	A diagonal matrix with diagonal elements from a square matrix
$\mathcal{L}_P[\cdot]$	The Lyapunov operator
$(\cdot)^\theta$ or $\phi_\theta(\cdot)$	The matrix power
$\mathcal{Q}(\cdot)$	Return an orthogonal matrix by QR decomposition
$\mathrm{skew}(\cdot)$	$\mathrm{skew}(A) = \frac{A-A^\top}{2}$

B.2 Brief review of Riemannian geometry

Intuitively, manifolds are locally Euclidean spaces. Differentials are the generalization of derivatives in classic calculus. For more details on smooth manifolds, please refer to [64, 40]. Riemannian manifolds are the manifolds endowed with Riemannian metrics, which can be intuitively viewed as point-wise inner products.

Definition B.1 (Riemannian Manifolds)**.** A Riemannian metric on $\mathcal{M}$ is a smooth symmetric covariant 2-tensor field on $\mathcal{M}$, which is positive definite at every point. A Riemannian manifold is a pair $\{\mathcal{M}, g\}$, where $\mathcal{M}$ is a smooth manifold and g is a Riemannian metric.

W.l.o.g., we abbreviate $\{\mathcal{M}, g\}$ as $\mathcal{M}$. The Riemannian metric g induces various Riemannian operators, including the geodesic, exponential, and logarithmic maps, and parallel transportation. These

operators correspond to straight lines, vector addition, vector subtraction, and parallel displacement in Euclidean spaces, respectively [52, Tabel 1]. A plethora of discussions on Riemannian geometry can be found in [21].

When a manifold $\mathcal{M}$ is endowed with a smooth operation, it is referred to as a Lie group.

Definition B.2 (Lie Groups)**.** A manifold is a Lie group, if it forms a group with a group operation $\odot$ such that $m(x, y) \mapsto x \odot y$ and $i(x) \mapsto x_{\odot}^{-1}$ are both smooth, where $x_{\odot}^{-1}$ is the group inverse of x.

Lastly, we review the definition of pullback metric, a common technique in Riemannian geometry. This idea is a natural generalization of bijection from set theory.

Definition B.3 (Pullback Metrics)**.** Suppose $\mathcal{M}, \mathcal{N}$ are smooth manifolds, g is a Riemannian metric on $\mathcal{N}$, and $f : \mathcal{M} \to \mathcal{N}$ is smooth. Then the pullback of g by f is defined point-wisely,

$$(f^* g)_p(V_1, V_2) = g_{f(p)}(f_{*,p}(V_1), f_{*,p}(V_2)), \tag{23}$$

where $p \in \mathcal{M}$, $f_{*,p}(\cdot)$ is the differential map of f at p, and $V_i \in T_p\mathcal{M}$. If f^*g is positive definite, it is a Riemannian metric on $\mathcal{M}$, which is called the pullback metric defined by f.

Table 12: The associated Riemannian operators and properties of five basic metrics on SPD manifolds.

Metrics	$g_P(V, W)$	$\operatorname{Log}_P Q$	$\Gamma_{P\to Q}(V)$	Properties
(α, β)-LEM	$\langle \log_{*,P}(V), \log_{*,P}(W) \rangle^{(\alpha,\beta)}$	$(\log_{*,P})^{-1}[\log(Q) - \log(P)]$	$(\log_{*,Q})^{-1} \circ \log_{*,P}(V)$	$\mathrm{O}(n)$-Invariance, Geodesically Completeness
(α, β)-AIM	$\langle P^{-1}V, WP^{-1} \rangle^{(\alpha,\beta)}$	$P^{1/2} \log\left(P^{-1/2} Q P^{-1/2}\right) P^{1/2}$	$(QP^{-1})^{1/2} V (P^{-1}Q)^{1/2}$	Lie Group Left-Invariance, $\mathrm{O}(n)$-Invariance, Geodesically Completeness
(α, β)-EM	$\langle V, W \rangle^{(\alpha,\beta)}$	$Q - P$	V	$\mathrm{O}(n)$-Invariance
LCM	$\sum_{i>j} \tilde{V}_{ij} \tilde{W}_{ij} + \sum_{j=1}^{n} \tilde{V}_{jj} \tilde{W}_{jj} L_{jj}^{-2}$	$(\mathrm{Chol}^{-1})_{*,L}\left[\lfloor K \rfloor - \lfloor L \rfloor + \mathbb{D}(L) \operatorname{Dlog}(\mathbb{D}(L)^{-1}\mathbb{D}(K))\right]$	$(\mathrm{Chol}^{-1})_{*,K}\left[\lfloor \tilde{V} \rfloor + \mathbb{D}(K)\mathbb{D}(L)^{-1}\mathbb{D}(\tilde{V})\right]$	Lie Group Bi-Invariance, Geodesically Completeness
BWM	$\frac{1}{2}\langle \mathcal{L}_P[V], W \rangle$	$(PQ)^{1/2} + (QP)^{1/2} - 2P$	$U\left[\sqrt{\frac{\delta_i+\delta_j}{\sigma_i+\sigma_j}}\left[U^\top V U\right]_{ij}\right] U^\top$	$\mathrm{O}(n)$-Invariance

B.3 Basic geometries of SPD manifolds

Let $\mathcal{S}_{++}^n$ be the set of $n \times n$ symmetric positive definite (SPD) matrices. As shown in [3], $\mathcal{S}_{++}^n$ is an open submanifold of the Euclidean space $\mathcal{S}^n$ of symmetric matrices. There are five kinds of popular Riemannian metrics on $\mathcal{S}_{++}^n$: Affine-Invariant Metric (AIM) [52], Log-Euclidean Metric (LEM) [3], Power-Euclidean Metrics (PEM) [22], Log-Cholesky Metric (LCM) [41], and Bures-Wasserstein Metric (BWM) [5]. Note that, when power equals 1, the PEM is reduced to the Euclidean Metric (EM). The standard LEM, AIM, and EM have been generalized into parametrized families of metrics. We define $\mathbf{ST} = \{(\alpha, \beta) \in \mathbb{R}^2 \mid \min(\alpha, \alpha + n\beta) > 0\}$, and denote the $\mathrm{O}(n)$-invariant Euclidean metric on $\mathcal{S}^n$ [63] as

$$\langle V, W \rangle^{(\alpha,\beta)} = \alpha \langle V, W \rangle + \beta \operatorname{tr}(V) \operatorname{tr}(W), \tag{24}$$

where $(\alpha, \beta) \in \mathbf{ST}$, and $\langle \cdot, \cdot \rangle$ is the Frobenius inner product. By $\mathrm{O}(n)$-invariant Euclidean metric on $\mathcal{S}^n$, Thanwerdas and Pennec [63] generalized AIM, LEM, and EM into two-parameters families of $\mathrm{O}(n)$-invariant metrics, *i.e.*,(α, β)-AIM, (α, β)-LEM, and (α, β)-EM, with $(\alpha, \beta) \in \mathbf{ST}$. We denote the metric tensor of (α, β)-AIM, (α, β)-LEM, (α, β)-EM, LCM, and BWM as $g^{(\alpha,\beta)\text{-AIM}}$, $g^{(\alpha,\beta)\text{-LEM}}$, $g^{(\alpha,\beta)\text{-EM}}$, g^{LCM}, and g^{BWM}, respectively.

For any SPD points $P, Q \in \mathcal{S}_{++}^n$ and tangent vectors $V, W \in T_P\mathcal{S}_{++}^n$, we follow the notations in Tab. 11 and further denote $\tilde{V} = \mathrm{Chol}_{*,P}(V)$, $\tilde{W} = \mathrm{Chol}_{*,P}(W)$, $L = \operatorname{Chol} P$, and $K = \operatorname{Chol} Q$. For parallel transportation under the BWM, we only present the case where P, Q are commuting matrices, *i.e.*,$P = U\Sigma U^\top$ and $Q = U\Delta U^\top$. We summarize the associated Riemannian operators and properties in Tab. 12. Although there also exist other metrics on SPD manifolds [60, 61, 63], their lack of closed-form Riemannian operators makes them problematic to be applied in machine learning.

B.4 Basic geometry of rotation matrices

Table 13: The associated Riemannian operators on Rotation matrices.

Operators	$g_R(A_1, A_2)$	$\operatorname{Log}_R S$	$\Gamma_{R\to S}(A)$	Projection Map $\Pi_R(U)$	Retraction of $A \in T_R\mathrm{SO}(n)$ at R
Expression	$\langle A_1, A_2 \rangle$	$\log(R^\top S)$	A	$\operatorname{skew}(R^\top U)$	$Q = \mathcal{Q}(R + RA)$

We denote $R \in \mathrm{SO}(n)$, $A_1, A_2 \in T_R\mathrm{SO}(n)$, $U \in \mathbb{R}^{n\times n}$, $\mathrm{skew}(A) = \frac{A-A^\top}{2}$, and $\mathcal{Q}(\cdot)$ as the function return an orthogonal matrix by QR decomposition. There are two equivalent representations for the tangent vector on $\mathrm{SO}(n)$. In this paper, we use the Lie algebra representation, *i.e.*,$T_R\mathrm{SO}(n) \cong \mathfrak{so}(n)$ as the set of skew-symmetric matrices. We summarize all the necessary Riemannian ingredients for $\mathrm{SO}(n)$ in Tab. 13.

For the specific case of $R \in \mathrm{SO}(3)$, R can be represented by the Euler angle and axis [26, Sec. 3.2]:

$$\theta(R) = \arccos\left(\frac{\mathrm{tr}(R)-1}{2}\right), \tag{25}$$

$$\omega\left(R\right) = \frac{1}{2\sin\left(\theta\left(R\right)\right)}\begin{pmatrix} R(3,2)-R(2,3) \\ R(1,3)-R(3,1) \\ R(2,1)-R(1,2) \end{pmatrix}. \tag{26}$$

Besides, the matrix logarithm on $\mathrm{SO}(3)$ can be calculated without decomposition [45, Ex. A.14]:

$$\log(R) = \begin{cases} 0, & \text{if } \theta(R) = 0 \\ \frac{\theta(R)}{2\sin(\theta(R))}\left(R - R^T\right), & \text{otherwise} \end{cases}, \tag{27}$$

where θ is the Euler angle. Obviously, the matrix logarithm is related to the Euler angle and axis when $\theta \neq 0$.

C RMLR as a natural extension of the Euclidean MLR

Proposition C.1. *When $\mathcal{M} = \mathbb{R}^n$ is the standard Euclidean space, the RMLR defined in Thm. 3.3 becomes the Euclidean MLR in Eq. (1).*

Proof. On the standard Euclidean space $\mathbb{R}^n$, $\mathrm{Log}_y x = x - y, \forall x, y \in \mathbb{R}^n$. Besides, the differential maps of left translation and parallel transportation are the identity maps. Therefore, given $x, p_k \in \mathbb{R}^n$ and $a_k \in \mathbb{R}^n/\{0\} \cong T_0\mathbb{R}^n/\{0\}$, we have

$$p(y = k \mid x \in \mathbb{R}^n) \propto \exp(\langle \mathrm{Log}_{p_k} x, a_k\rangle_{p_k}), \tag{28}$$

$$\propto \exp(\langle x - p_k, a_k\rangle), \tag{29}$$

$$\propto \exp(\langle x, a_k\rangle - b_k), \tag{30}$$

where $b_k = \langle x, p_k\rangle$. □

D Gyro SPSD MLR as special cases of our RMLR

Gyro SPSD MLR [51] is derived by the product of the Grassmannian and SPD gyro spaces. This section will show that the gyro SPSD MLR is the special case of our RMLR on the product geometry of the SPSD manifold. We first review some necessary results about gyro SPSD MLR and then show the equivalence.

Following the notations in [51], we denote the Grassmannian with canonical metric under the projector and ONB perspective as $\mathrm{Gr}(p,n)$ and $\widetilde{\mathrm{Gr}}(p,n)$, respectively. The space of $n \times n$ SPSD matrices with a fixed rank p, denoted as $\mathcal{S}^+_{n,p}$, forms an SPSD manifold [7]. As shown in [7, 51], the SPSD manifold is a product space, *i.e.*,$\mathcal{S}^+_{n,p} \cong \widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}^p_{++}$. In other words, every $P \in \mathcal{S}^+_{n,p}$ can be decomposed as $P = U_P S_P U_P^\top$ with $U_p \in \widetilde{\mathrm{Gr}}(p,n)$ and $S_P \in \mathcal{S}^p_{++}$. We further denote $\mathcal{S}^{p,g}_{++}$ as the SPD manifold with metric g, where g could be AIM, LEM, and LCM. As shown in [51], the gyro space in $\mathcal{S}^+_{n,p}$ can be defined by the product of gyro spaces of $\widetilde{\mathrm{Gr}}(p,n)$ and $\mathcal{S}^{p,g}_{++}$. By this product structure, Nguyen et al. [51] proposed the SPSD Pseudo-gyrodistance to a hyperplane.

Definition D.1. (SPSD Hypergyroplanes [51]) Let $P, W \in \widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}^{n,g}_{++}$. Then hypergyroplanes in structure space $\widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}^{n,g}_{++}$ are defined as

$$H^{psd,g}_{W,P} = \left\{Q \in \widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}^{n,g}_{++} : \langle \ominus_{psd,g} P \oplus_{psd,g} Q, W\rangle^{psd,g} = 0\right\}. \tag{31}$$

where $\oplus_{psd,g}$ and $\langle,\rangle^{psd,g}$ are gyro addition and gyro inner product, which are defined in [51].

Theorem D.2. *(SPSD Pseudo-gyrodistance [51]) Let* $W = (U_W, S_W), P = (U_P, S_P), X = (U_X, S_X) \in \widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}_{++}^{n,g}$*, and* $\mathcal{H}_{A,P}^{psd,g}$ *be a hypergyroplane in structure space* $\widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}_{++}^{n,g}$*. Then the pseudo-gyrodistance from* X *to* $\mathcal{H}_{A,P}^{psd,g}$ *is given by*

$$\bar{d}\left(X, H_{W,P}^{psd,g}\right) = \frac{\left|\lambda\left\langle\left(\widetilde{\ominus}_{gr} U_P \widetilde{\oplus}_{gr} U_X\right)\left(\widetilde{\ominus}_{gr} U_P \widetilde{\oplus}_{gr} U_X\right)^T, U_W U_W^T\right\rangle^{gr} + \langle \ominus_g S_P \oplus_g S_X, S_W\rangle^g\right|}{\sqrt{\lambda\left(\left\|U_W U_W^T\right\|^{gr}\right)^2 + \left(\|S_W\|^g\right)^2}}, \tag{32}$$

where $\|.\|^{gr}$ *and* $\|.\|^g$ *are the gyro norms on the Grassmann and SPD [51], and* $\langle,\rangle^{gr}$ *and* $\langle,\rangle^g$ *are gyro inner products [51].* $\widetilde{\oplus}_{gr}$ *and* $\oplus_g$ *are gyro additions on* $\widetilde{\mathrm{Gr}}(p,n)$ *and* $\mathcal{S}_{++}^{p,g}$*.*

Denoting g^{gr} as the canonical metric on $\widetilde{\mathrm{Gr}}(p,n)$ and g as AIM, LEM, or LCM, we can prove that Thm. D.2 is the special case of our Thm. 3.2.

Theorem D.3. *Under the product metric* $g^{psd,g} = \lambda g^{gr} \times g$*, the Riemannian hyperplane in Eq. (5) on the SPSD manifold equals the SPSD hypergyroplane in Def. D.1. Similarly, the Riemannian margin distance in Thm. 3.2 on the SPSD manifold equals SPSD Pseudo-gyrodistance in Thm. D.2.*

Proof. Following the notations in Def. D.1 and Thm. D.2, we further denote $P = U_P S_P U_P^\top$, $Q = U_Q S_Q U_Q^\top$, $W = U_W S_W U_W^\top$, and $X = U_X S_X U_X^\top$ with $U_P, U_Q, U_W, U_X \in \widetilde{\mathrm{Gr}}(p,n)$ and $S_P, S_Q, S_W, S_X \in \mathcal{S}_{++}^{p,g}$. I_p is the $p \times p$ identity matrix. $\widetilde{I}_{p,n} = (I_p, 0)^\top$ is the gyro identity on $\widetilde{\mathrm{Gr}}(p,n)$. $\widetilde{\Gamma}^{gr}$, $\widetilde{\mathrm{Log}}^{gr}$, and $\langle,\rangle_{U_p}^{gr}$ are Riemannian parallel transport along a geodesic, logarithm and Riemannian metric at U_p on $\widetilde{\mathrm{Gr}}(p,n)$. Γ^g, Log^g, and $\langle,\rangle_{S_p}^g$ are Riemannian parallel transport along a geodesic, logarithm and Riemannian metric at S_p on $\mathcal{S}_{++}^{p,g}$. $\Gamma^{psd,g}$, $\mathrm{Log}^{psd,g}$, and $\langle,\rangle_X^{psd,g}$ are Riemannian parallel transport along a geodesic, logarithm and Riemannian metric at X on $\mathcal{S}_{n,p}^+ \cong \widetilde{\mathrm{Gr}}(p,n) \times \mathcal{S}_{++}^p$.

First, we show that the SPSD hypergyroplane equals our Riemannian hyperplane in Eq. (5). We have the following

$$\begin{aligned}
&\langle \ominus_{psd,g} P \oplus_{psd,g} Q, W\rangle^{psd,g} \\
&\overset{(1)}{=} \lambda\langle \ominus_{gr} U_P \oplus_{gr} U_Q, U_W\rangle^{gr} + \langle \ominus_g S_P \oplus_g S_Q, S_W\rangle^g \\
&\overset{(2)}{=} \lambda\langle \mathrm{Log}_{U_P}^{gr} U_Q, A_{U_W}\rangle_{U_P}^{gr} + \langle \mathrm{Log}_{S_P}^{gr} S_Q, A_{S_W}\rangle_{S_P}^g \\
&\overset{(3)}{=} \langle \mathrm{Log}_P^{psd,g} Q, \tilde{A}\rangle_P^{psd,g}
\end{aligned} \tag{33}$$

where $\tilde{A}_{U_W} = \widetilde{\Gamma}_{\widetilde{I}_{p,n} \to U_P}^{gr}\left(\widetilde{\mathrm{Log}}_{\widetilde{I}_{p,n}}^{gr}(U_W)\right)$, $\tilde{A}_{S_W} = \Gamma_{I_p \to S_P}^g\left(\mathrm{Log}_{I_p}^g(S_W)\right)$ and $\tilde{A} = (\tilde{A}_{U_W}, \tilde{A}_{S_W}) \in T_P\mathcal{S}_{n,p}^+ \cong T_{U_P}\widetilde{\mathrm{Gr}}(p,P) \otimes T_{S_P}\mathcal{S}_{++}^{p,g}$ with $\otimes$ as the Cartesian product. The above derivation comes from the following.

(1) The definition of gyro addition, gyro inverse, and gyro inner product on the SPSD manifold [51, Sec. 3.3].

(2) The proof of [51, Prop. 3.2] indicates that similar results also hold on the Grassmannian. Combining Prop. 3.2 and its counterparts in the Grassmannian, one can obtain the equation.

(3) The Riemannian product geometry.

By the product geometry of the SPSD manifold, we can immediately get

$$\tilde{A} = (\tilde{A}_{U_W}, \tilde{A}_{S_W}) = \Gamma_{I_{p,n} \to P}^{psd,g}\left(\mathrm{Log}_{I_{p,n}}^{psd,g}(W)\right) \tag{34}$$

where $I_{p,n} = \widetilde{I}_{p,n}\widetilde{I}_{p,n}^\top$ is the gyro identity on the SPSD manifold.

Next, we show the equivalence between SPSD pseudo-gyrodistance and our Riemannian margin distance:

$$
\begin{aligned}
\bar{d}\left(X, H_{W,P}^{psd,g}\right) &\overset{(1)}{=} \frac{\left|\langle \ominus_{psd,g} P \oplus_{psd,g} X, W\rangle^{psd,g}\right|}{\|W\|^{psd,g}} \\
&\overset{(2)}{=} \frac{\left|\langle \mathrm{Log}_P^{psd,g} X, \tilde{A}\rangle_P^{psd,g}\right|}{\|W\|^{psd,g}} \\
&\overset{(3)}{=} \frac{\left|\langle \mathrm{Log}_P^{psd,g} X, \tilde{A}\rangle_P^{psd,g}\right|}{\left\|\mathrm{Log}_{I_{p,n}}^{psd,g}(W)\right\|_{I_{p,n}}^{psd,g}} \\
&\overset{(4)}{=} \frac{\left|\langle \mathrm{Log}_P^{psd,g} X, \tilde{A}\rangle_P^{psd,g}\right|}{\left\|\Gamma_{I_{p,n}\to P}^{psd,g}\left(\mathrm{Log}_{I_{p,n}}^{psd,g}(W)\right)\right\|_P^{psd,g}} \\
&\overset{(5)}{=} \frac{\left|\langle \mathrm{Log}_P^{psd,g} X, \tilde{A}\rangle_P^{psd,g}\right|}{\left\|\tilde{A}\right\|_P^{psd,g}} \\
&\overset{(6)}{=} d(S, \tilde{H}_{\tilde{A},P})
\end{aligned}
\tag{35}
$$

(1) The definition of gyro addition, gyro inverse, gyro inner product, and gyro norm on the SPSD manifold.

(2) Eq. (33).

(3) The definition of SPSD gyro norm [51].

(4) Riemannian parallel transportation maintains the norm of the tangent vector [21, Def. 3.1]

(5) Eq. (34)

(6) Thm. 3.2

□

Remark D.4. We make the following remark w.r.t. gyro and our MLR on the SPSD manifold.

1. Eq. (34) indicates that when generating $\tilde{A}$ in our RMLR by parallel transporting a tangent vector $A \in T_{I_{p,n}}\mathcal{S}_{n,p}^{+}$, $\tilde{A}$ is the initial velocity of W in Eq. (32).
2. Putting pseudo-gyrodistance and Riemannian margin distance into Eq. (4), one can get gyro MLR and our Riemannian MLR. Therefore, Thm. D.3 indicates the equivalence of the gyro MLR with our RMLR on the SPSD.
3. As g are required to induce gyro structures, the metric g in gyro SPSD MLR is confined within AIM, LEM, and LCM. However, our SPSD MLR can be the product space of the Grassmannian and SPD manifold under other metrics, such as BWM and PEM, as our framework does not require gyro structures.

E Theories on the deformed metrics

E.1 Limiting cases of the deformed metrics

Thanwerdas and Pennec [59] generalized (α, β)-AIM into three-parameters families of metrics by power deformation, *i.e.*,(θ, α, β)-AIM. The family of (θ, α, β)-AIM comprises (α, β)-AIM for $\theta = 1$ and approaches (α, β)-LEM with $\theta \to 0$ [59].

Chen et al. [17] extended LCM and (α, β)-LEM into power-deformed metrics, denoted as (θ, α, β)-LEM and θ-LCM. The authors show that (θ, α, β)-LEM is equal to (α, β)-LEM, and θ-LCM interpolates between $\tilde{g}$-LEM ($\theta \to 0$) and LCM ($\theta = 1$), with $\tilde{g}$-LEM defined as

$$
\langle V_1, V_2\rangle_P = \frac{1}{2}\langle \widetilde{V_1}, \widetilde{V_2}\rangle - \frac{1}{4}\langle \mathbb{D}(\widetilde{V_1}), \mathbb{D}(\widetilde{V_2})\rangle, \forall V_i \in T_P\mathcal{S}_{++}^{n}, \tag{36}
$$

where $\widetilde{V}_i = \log_{*,P}(V_i)$ with $\log_{*,P}$ as the differential map of matrix logarithm, and $\mathbb{D}(V_i)$ is a diagonal matrix consisting of the diagonal elements of V_i.

Thanwerdas and Pennec [61] identified the Alpha-Procrustes metric [43] with power-deformed BWM, denote as 2θ-BWM. Similarly, 2θ-BWM becomes BWM with $\theta = 0.5$ [61]. We further show the limiting case of 2θ-BWM under $\theta \to 0$.

Proposition E.1. *2θ-BWM tends to be $(\frac{1}{4}, 0)$-LEM with $\theta \to 0$.*

Before starting the proof, we first recall a well-known property of deformed metrics [61].

Lemma E.2. *Let $\frac{1}{\theta^2}\phi_\theta^* g$ be the deformed metric on SPD manifolds pulled back from g by the matrix power ϕ_θ and scaled by $\frac{1}{\theta^2}$. Then when θ tends to 0, for all $P \in \mathcal{S}_{++}^n$ and all $V \in T_P\mathcal{S}_{++}^n$, we have*

$$(\frac{1}{\theta^2}\phi_\theta^* g)_P(V, V) \to g_I(\log_{*,P}(V), \log_{*,P}(V)). \tag{37}$$

Now, we present our proof for the limiting cases of deformed metrics.

Proof of Prop. E.1. First, we have

$$g_I^{\text{BWM}}(V, V) = \frac{1}{4}\langle V, V\rangle. \tag{38}$$

By Lem. E.2, we have the following:

$$\begin{aligned} g_P^{2\theta\text{-BWM}}(V, V) &\xrightarrow{\theta\to 0} g_I^{\text{BWM}}\left(\log_{*,P}(V), \log_{*,P}(V)\right) \\ &= \frac{1}{4}\langle \log_{*,P}(V), \log_{*,P}(V)\rangle \\ &= g_P^{(\frac{1}{4},0)\text{-LEM}}(V, V)\,. \end{aligned} \tag{39}$$

□

E.2 Proof of the properties of the deformed metrics (Tab. 2)

In this subsection, we prove the properties presented in Tab. 2. We first present a useful lemma and then present our detailed proof. This lemma will be useful in the proof of our SPD MLRs as well.

Lemma E.3. *Supposing a Riemannian manifold $\{\mathcal{M}, g\}$ and a positive real scalar $a > 0$, the scaling metric ag over $\mathcal{M}$ shares the same Riemannian logarithmic & exponential maps and parallel transportation with g.*

Proof. Since the Christoffel symbols of ag are identical to those of g, the geodesics and parallel transportation under both ag and g remain unchanged. The equivalence of geodesics implies that the Riemannian exponential maps are the same for ag and g. As the inverse of the Riemannian exponential maps, the Riemannian logarithm maps under ag and g are also identical. □

According to Lem. E.3, the geodesic completeness is independent of the scaling factor $a > 0$. By the definition of $\mathrm{O}(n)$-, left-, right-, and bi-invariance, these invariant properties are also independent of the scaling factor $a > 0$. Without loss of generality, we will omit the scaling factor in the following proof.

Proof. Firstly, we prove $\mathrm{O}(n)$-invariance of (θ, α, β)-LEM, (θ, α, β)-EM, (θ, α, β)-AIM, and 2θ-BWM. Since the differential of ϕ_θ is $\mathrm{O}(n)$-equivariant, and (α, β)-LEM, (α, β)-EM, (α, β)-AIM, and BWM are $\mathrm{O}(n)$-invariant [63], $\mathrm{O}(n)$-invariance are thus acquired.

Next, we focus on geodesic completeness. It can be easily proven that Riemannian isometries preserve geodesic completeness. On the other hand, (α, β)-LEM, (α, β)-AIM, and LCM are geodesically complete [63, 41]. As a direct corollary, geodesic completeness can be obtained since ϕ_θ is a Riemannian isometry.

Finally, we deal with Lie group invariance. Similarly, it can be readily proved that Lie group invariance is preserved under isometries. LCM, LEM, and (α, β)-AIM are Lie group bi-invariant [41], bi-invariant [3], and left-invariant [62]. As an isometric pullback metric from the standard LEM [63], (α, β)-LEM is, therefore, Lie group bi-invariant. As pullback metrics, (θ, α, β)-LEM, (θ, α, β)-AIM, and θ-LCM are therefore bi-invariant, left-invariant, and bi-invariant, respectively. □

F Computational details on the SPD MLR under power-deformed BWM

F.1 Matrix square roots in the SPD MLR under power-deformed BWM

In the case of MLRs induced by 2θ-BWM, computing square roots like $(BA)^{\frac{1}{2}}$ and $(AB)^{\frac{1}{2}}$ with $B, A \in \mathcal{S}_{++}^n$ poses a challenge. Eigendecomposition cannot be directly applied since BA and AB are no longer symmetric, let alone positive definitity. Instead, we use the following formulas to compute these square roots [43]:

$$(BA)^{\frac{1}{2}} = B^{\frac{1}{2}}(B^{\frac{1}{2}}AB^{\frac{1}{2}})^{\frac{1}{2}}B^{-\frac{1}{2}} \text{ and } (AB)^{\frac{1}{2}} = [(BA)^{\frac{1}{2}}]^{\top}, \tag{40}$$

where the involved square roots can be computed using eigendecomposition or singular value decomposition (SVD).

F.2 Numerical stability of the SPD MLR under power-deformed BWM

Let us first explain why we abandon parallel transportation on the SPD MLR derived from 2θ-BWM. Then, we propose our numerically stable methods for computing the SPD MLR based on 2θ-BWM.

F.2.1 Instability of parallel transportation under power-deformed BWM

As discussed in Thm. 3.3, there are two ways to generate $\tilde{A}$ in SPD MLR: parallel transportation and Lie group translation. However, parallel transportation under 2θ-BWM could cause numerical problems. W.l.o.g., we focus on the standard BWM as 2θ-BWM is isometric to the BWM.

Although the general solution of parallel transportation under BWM is the solution of an ODE, for the case of parallel transportation starting from the identity matrix, we have a closed-form expression [63]:

$$\Gamma_{I\to P}(V) = U\left[\sqrt{\frac{\sigma_i+\sigma_j}{2}}\left[U^{\top}VU\right]_{ij}\right]U^{\top}, \tag{41}$$

where $P = U\Sigma U^{\top}$ is the eigendecomposition of $P \in \mathcal{S}_{++}^n$. There would be no problem in the forward computation of Eq. (41). However, during backpropagation (BP), Eq. (41) would require the BP of eigendecomposition, involving the calculation of $1/(\sigma_i-\sigma_j)$ [33, Prop. 2]. When σ_i is close to σ_j, the BP of eigendecomposition could be problematic.

F.2.2 Numerically stable methods for the SPD MLR under power-deformed BWM

To bypass the instability of parallel transportation under BWM, we use Lie group left translation to generate $\tilde{A}$ in MLRs induced from 2θ-BWM. However, there is another problem that could cause instability. The computation of the Riemannian metric of 2θ-BWM requires solving the Lyapunov operator, *i.e.,*$\mathcal{L}_P[V]P + P\mathcal{L}_P[V] = V$. Under the case of symmetric matrices, the Lyapunov operator can be obtained by eigendecomposition:

$$\mathcal{L}_P[V] = U\left[\frac{V'_{ij}}{\sigma_i+\sigma_j}\right]_{i,j}U^{\top}, \tag{42}$$

where $V \in \mathcal{S}^n$, $UV'U^{\top} = V$, and $P = U\Sigma U^{\top}$ is the eigendecomposition of $P \in \mathcal{S}_{++}^n$. Similar with Eq. (41), the BP of Eq. (42) requires $1/(\sigma_i-\sigma_j)$, undermining the numerical stability.

To remedy this problem, we proposed the following formula to stably compute the BP of Eq. (42).

Proposition F.1. *For all $P \in \mathcal{S}_{++}^n$ and all $V \in \mathcal{S}^n$, we denote the Lyapunov equation as*

$$XP + PX = V, \tag{43}$$

where $X = \mathcal{L}_P[V]$. Given the gradient $\frac{\partial L}{\partial X}$ of loss L w.r.t. X, then the BP of the Lyapunov operator can be computed by:

$$\frac{\partial L}{\partial V} = \mathcal{L}_P[\frac{\partial L}{\partial X}], \tag{44}$$

$$\frac{\partial L}{\partial P} = -X\mathcal{L}_P[\frac{\partial L}{\partial X}] - \mathcal{L}_P[\frac{\partial L}{\partial X}]X, \tag{45}$$

where $\mathcal{L}_P[\cdot]$ can be computed by Eq. (42).

Proof. Differentiating both sides of Eq. (43), we obtain

$$\mathrm{d}\,XP + X\,\mathrm{d}\,P + \mathrm{d}\,PX + P\,\mathrm{d}\,X = \mathrm{d}\,V, \tag{46}$$

$$\Longrightarrow \mathrm{d}\,XP + P\,\mathrm{d}\,X = \mathrm{d}\,V - X\,\mathrm{d}\,P - \mathrm{d}\,PX, \tag{47}$$

$$\Longrightarrow \mathrm{d}\,X = \mathcal{L}_P[\mathrm{d}\,V - X\,\mathrm{d}\,P - \mathrm{d}\,PX]. \tag{48}$$

Besides, easy computations show that

$$\mathcal{L}_P[V] : W = V : \mathcal{L}_P[W], \forall W, V \in \mathcal{S}^n, \tag{49}$$

where $\cdot : \cdot$ denotes the standard Frobenius inner product.

Then we have the following:

$$\frac{\partial L}{\partial X} : \mathrm{d}\,X = \frac{\partial L}{\partial X} : \mathcal{L}_P[\mathrm{d}\,V - X\,\mathrm{d}\,P - \mathrm{d}\,PX], \tag{50}$$

$$\Longrightarrow \frac{\partial L}{\partial X} : \mathrm{d}\,X = \mathcal{L}_P[\frac{\partial L}{\partial X}] : \mathrm{d}\,V + \left(-X\mathcal{L}_P[\frac{\partial L}{\partial X}] - \mathcal{L}_P[\frac{\partial L}{\partial X}]X\right) : \mathrm{d}\,P. \tag{51}$$

□

Remark F.2. Eq. (42) needs to be computed in the Lyapunov operator's forward and backward process. Therefore, in the forward process, we can save the intermediate matrices U and K with $K_{i,j} = \left[\frac{1}{\sigma_i + \sigma_j}\right]_{i,j}$, and then use them to compute the backward process efficiently.

G Implementation details and additional experiments

This section offers additional details on the experiments of SPD and Lie MLRs.

G.1 Additional details and experiments on the SPD MLRs

G.1.1 Basic layers in SPDNet and TSMNet

SPDNet [30] is the most classic SPD neural network. SPDNet mimics the conventional densely connected feedforward network, consisting of three basic building blocks:

$$\text{BiMap layer: } S^k = W^k S^{k-1} W^{k\top}, \text{ with } W^k \text{ semi-orthogonal,} \tag{52}$$

$$\text{ReEig layer: } S^k = U^{k-1} \max(\Sigma^{k-1}, \epsilon I_n) U^{k-1\top}, \text{ with } S^{k-1} = U^{k-1}\Sigma^{k-1}U^{k-1\top}, \tag{53}$$

$$\text{LogEig layer: } S^k = \log(S^{k-1}). \tag{54}$$

where $\max()$ is element-wise maximization. BiMap and ReEig mimic transformation and non-linear activation, while LogEig maps SPD matrices into the tangent space at the identity matrix for classification.

SPDNetBN [10] further proposed Riemannian batch normalization based on AIM:

$$\text{Centering from geometric mean } \mathfrak{G} : \forall i \leq N, \bar{S}_i = \mathfrak{G}^{-\frac{1}{2}} S_i \mathfrak{G}^{-\frac{1}{2}}, \tag{55}$$

$$\text{Biasing towards SPD parameter } G : \forall i \leq N, \tilde{S}_i = G^{\frac{1}{2}} \bar{S}_i G^{\frac{1}{2}}. \tag{56}$$

SPD domain-specific momentum batch normalization (SPDDSMBN) [37] is an improved version of SPDNetBN. Apart from controlling the mean, it can also control variance. The key operation in SPDDSMBN of controlling mean and variance is:

$$\Gamma_{I\to G} \circ \Gamma_{\mathfrak{G}\to I}(S_i)\frac{\nu}{\bar{\nu}+\varepsilon}, \tag{57}$$

where $\mathfrak{G}$ and $\bar{v}$ are Riemannian mean and variance. Inspired by [75], during the training stage, SPDDSMBN generates running means and running variances for training and testing with distinct momentum parameters. Besides, it sets $\mathfrak{G}$ and $\bar{v}$ as the running mean and running variance w.r.t. training for training and the ones w.r.t. testing for testing. SPDDSMBN also applies domain-specific techniques [13], keeping multiple parallel BN layers and distributing observations according to the associated domains. To crack cross-domain knowledge, v is uniformly learned across all domains, and G is set to be the identity matrix. TSMNet [37] adopted SPDDSMBN to solve domain adaptation in EEG classification.

In the above models, the Euclidean MLR in the co-domain of matrix logarithm (matrix logarithm + FC + softmax) is used for classification. Following the terminology in [16], we call this classifier as **LogEig MLR**. The LogEig MLR is the Euclidean classifier in the tangent space at the identity, which might distort the innate geometry of the SPD manifold.

G.1.2 Datasets and preprocessing

Radar[2]**:** This dataset [10] consists of 3,000 synthetic radar signals. Following the protocol in [10], each signal is split into windows of length 20, resulting in 3,000 SPD covariance matrices of 20×20 equally distributed in 3 classes.

HDM05[3]**:** This dataset [44] contains 2,273 skeleton-based motion capture sequences executed by various actors. Each frame consists of 3D coordinates of 31 joints of the subjects, and each sequence can be, therefore, modeled by a 93×93 covariance matrix. Following the protocol in [10], we trim the dataset down to 2086 sequences scattered throughout 117 classes by removing some under-represented classes.

Hinss2021[4]**:** This dataset [28] is a recent competition dataset consisting of EEG signals for mental workload estimation. The dataset is used for two types of experiments: inter-session and inter-subject, which are modeled as domain adaptation problems. Recently, geometry-aware methods have shown promising performance in EEG classification [74, 37]. We choose the SOTA method, TSMNet [37], as our baseline model on this dataset. We follow the Python implementation[5] [37] to carry out preprocessing. In detail, the python package MOABB [35] and MNE [24] are used to preprocess the datasets. The applied steps include resampling the EEG signals to 250/256 Hz, applying temporal filters to extract oscillatory EEG activity in the 4 to 36 Hz range, extracting short segments ($\leq$ 3s) associated with a class label, and finally obtaining 40×40 SPD covariance matrices.

Disease [2]: It represents a disease propagation tree, simulating the SIR disease transmission model [2], with each node representing either an infection or a non-infection state.

Cora [54]: It is a citation network where nodes represent scientific papers in the area of machine learning, edges are citations between them, and node labels are academic (sub)areas.

Pubmed [46]: This is a standard benchmark describing citation networks where nodes represent scientific papers in the area of medicine, edges are citations between them, and node labels are academic (sub)areas.

For the Disease, Cora and Pubmed datasets, we follow [76] to model features into $\mathcal{S}^3_{++}$.

G.1.3 Implementation details

SPDNet [30] and TSMNet [37]: We follow the official Pytorch code of SPDNetBN[6] and TSMNet[7] to implement our experiments. To evaluate the performance of our intrinsic classifiers, we substitute the LogEig MLR in SPDNet and TSMNet with our SPD MLRs. We implement our SPD MLRs induced from five parameterized metrics. On the Radar and HDM05 datasets, the learning rate is $1e^{-2}$, and the batch size is 30. On the Hinss2021 dataset, following [37], the learning rate is $1e^{-3}$ with a $1e^{-4}$ weight decay, and batch size is 50. The maximum training epoch is 200, 200, and 50, respectively. We use the standard-cross entropy loss as the training objective and optimize the parameters with the Riemannian AMSGrad optimizer [4].

RResNet [36]: We focus on the AIM-based RResNet, and use the official code[8] and suggested network settings to implement the experiments on the RResNet. We conduct 10-fold and 5-fold experiments on the HDM05 and NTU datasets. Since RResNet is developed based on SPDNet, we use the same learning settings with the SPDNet for the action recognition task, and borrow the best (θ, α, β) from Tab. 4 for our SPD MLRs under the RResNet backbone.

SPDGCN [76]: We use the official code[9] and the suggested network settings in [76]. Note that the SPDGCN with SPD MLR remains the same network settings as the vanilla SPDGCN. Tab. 14 presents the hyperparameters (θ, α, β) on different datasets.

[2] https://www.dropbox.com/s/dfnlx2bnyh3kjwy/data.zip?dl=0

[3] https://resources.mpi-inf.mpg.de/HDM05/

[4] https://zenodo.org/record/5055046

[5] https://github.com/rkobler/TSMNet

[6] https://proceedings.neurips.cc/paper_files/paper/2019/file/6e69ebbfad976d4637bb4b39de261bf7-Supplemental.zip

[7] https://github.com/rkobler/TSMNet

[8] https://github.com/CUAI/Riemannian-Residual-Neural-Networks

[9] https://github.com/andyweizhao/SPD4GNNs

Table 14: (θ, α, β) of SPD MLRs on the SPDGCN backbone.

Datasets	(θ, α, β)-AIM	(θ, α, β)-EM	(α, β)-LEM	2θ-BWM	θ-LCM
Disease	(0.25,1,0)	(0.25,1,0)	(1,1)	0.25	0.5
Cora	(0.5,1,0)	(0.25,1,1/9)	(1,1/9)	0.25	0.5
Pubmed	(0.5,1,0)	(0.5,1,0)	(1,-1/3)	0.25	0.5

Network Architectures: We denote the network architecture as $[d_0, d_1, \cdots, d_L]$, where the dimension of the parameter in the i-th BiMap layer (App. G.1.1) is $d_i \times d_{i-1}$. For SPDNet, we also validate our SPD MLRs under different network architectures on the Radar and HDM05 datasets. The network architectures on the Radar dataset are [20, 16, 8] for the 2-block configuration and [20, 16, 14, 12, 10, 8] for the 5-block configuration, while on the HDM05 dataset, the network architectures are [93, 30] for 1-block, [93. 70, 30] for 2-block, and [93, 70, 50, 30] for 3-block. For TSMNet, the 1-block architecture is [40,20].

Scoring Metrics: In line with the previous work [10, 37, 76, 36], we use balanced accuracy, the average recall across classes, as the scoring metric for the Hinss2021 dataset, and accuracy for other datasets. On the Hinss2021 dataset, models are fit and evaluated with a randomized leave 5% of the sessions (inter-session) or subjects (inter-subject) out cross-validation (CV) scheme. On other datasets, K-fold experiments are carried out with randomized initialization and split,

G.1.4 Hyper-parameters

We implement the SPD MLRs induced by not only five standard metrics, *i.e.,*LEM, AIM, EM, LCM, and BWM, but also five families of parameterized metrics. Therefore, in our SPD MLRs, we have a maximum of three hyper-parameters, *i.e.,*θ, α, β, where (α, β) are associated with $\mathrm{O}(n)$-invariance and θ controls deformation. For (α, β) in (θ, α, β)-LEM, (θ, α, β)-AIM, and (θ, α, β)-EM, recalling Eq. (24), α is a scaling factors, while β measures the relative significance of traces. As scaling is less important [59], we set $\alpha = 1$. As for the value of β, we select it from a predefined set: $\{1, 1/n, 1/n^2, 0, -1/n + \epsilon, -1/n^2\}$, where n is the dimension of input SPD matrices in SPD MLRs. The purpose of including $\epsilon \in \mathbb{R}_+$ is to ensure $\mathrm{O}(n)$-invariance ($(\alpha, \beta) \in \mathbf{ST}$). These chosen values for β allow for amplifying, neutralizing, or suppressing the trace components, depending on the characteristics of the datasets. For the deformation factor θ, we roughly select its value around its deformation boundary, *i.e.,*[0.25,1.5] for (θ, α, β)-AIM, [0.5,1.5] for θ-LCM, [0.25,1.5] and (θ, α, β)-EM, [0.25,0.75] for 2θ-BWM. The details values are listed in Tab. 15.

Table 15: Candidate values for hyper-parameters in SPD MLRs

Metric	(θ, α, β)-AIM	(θ, α, β)-EM	θ-LCM	2θ-BWM
Candidate Values	{ 0.25,0.5,0.75,1,1.25,1.5 }	{0.5,1,1.5 }	{0.5,1,1.5 }	{0.25,0.5,0.75 }

G.1.5 Model efficiency

Table 16: Training efficiency (s/epoch).

Methods	Radar	HDM05	Hinss2021	
			Inter-session	Inter-subject
Baseline	1.36	1.95	0.18	8.31
AIM-MLR	1.75	31.64	0.38	13.3
EM-MLR	1.34	3.91	0.19	8.23
LEM-MLR	1.5	4.7	0.24	10.13
BWM-MLR	1.75	33.14	0.38	13.84
LCM-MLR	1.35	3.29	0.18	8.35

We adopt the deepest architectures, namely [20, 16, 14, 12, 10, 8] for the Radar dataset, [93, 70, 50, 30] for the HDM05 dataset, and [40, 20] for the Hinss2021 dataset. For simplicity, we focus on the SPD MLRs induced by standard metrics, *i.e.,*AIM, EM, LEM, BWM, and LCM. The average training time (in seconds) per epoch is reported in Tab. 16. Generally, when the number of classes is small

(*e.g.*,3 in the Radar and Hinss2021 datasets), our SPD MLRs only bring minor additional training time compared to the baseline LogEig MLR. However, when dealing with a larger number of classes (*e.g.*,117 classes in the HDM05 dataset), there could be some inefficiency caused by our SPD MLRs. This is because each class requires an SPD parameter, and each parameter might require matrix decomposition in the forward or optimization processes during training. Nonetheless, the SPD MLRs induced by EM or LCM generally achieve comparable efficiency with the vanilla LogEig MLR. This is due to the fast computation of their Riemannian operators, making them efficient choices for tasks with a larger number of classes. This result highlights the flexibility of our framework and its applicability to various scenarios.

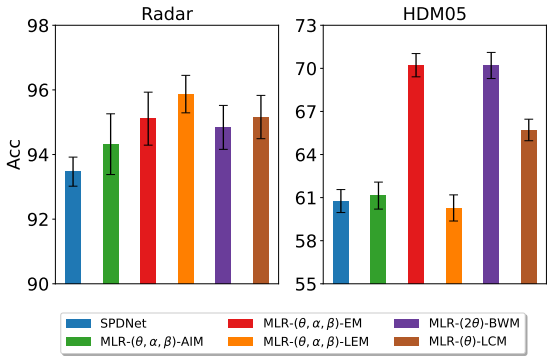

Figure 4: Visualization of 10-fold average accuracy of SPDNet with different SPD MLRs on the Radar and HDM05 datasets. The error bar denotes the standard deviation.

G.1.6 Visualization

We visualize the 10-fold average results of SPDNet with different classifiers on the Radar and HDM05 datasets. We focus on the deepest architectures, *i.e.*,. [20,16,14,12,10,8] for the Radar dataset, and [93,70,50,30] for the HDM05 dataset. Note that we only report the SPD MLR with the best hyper-parameters (θ, α, β). The figures are presented in Fig. 4. All the results are sourced from Tabs. 3 and 4.

G.2 Additional details and experiments on the Lie MLR

G.2.1 Basic layers in LieNet

LieNet [31] is the most classic neural network on rotation matrices. The latent space of LieNet is the Lie group $\mathrm{SO}^N(3) = \mathrm{SO}(3) \times \mathrm{SO}(3) \cdots \times \mathrm{SO}(3)$, *i.e.*,$R = (R_1, \cdots, R_N) \in \mathrm{SO}^N(3)$. The group and manifold structures on $\mathrm{SO}^N(3)$ are defined by product spaces. For instance, $R^1 \odot R^2 = (R_1^1 R_1^2, \cdots, R_N^1 R_N^2)$. There are three basic layers in LieNet:

$$\text{RotMap layer: } R^k = W^k \odot R^{k-1}, \text{ with } W^k \in \mathrm{SO}^N(3), \tag{58}$$

$$\text{RotPooling layer: } R_i^k = \begin{cases} R_{m_i,n_i}^{k-1}, & \text{if } \Theta\left(R_{m_i,n_i}^{k-1}\right) > \Theta\left(R_{n_i,m_i}^{k-1}\right), \\ R_{n_i,m_i}^{k-1}, & \text{otherwise,} \end{cases}, \tag{59}$$

$$\text{LogMap layer: } R^k = \log(R^{k-1}), \tag{60}$$

where $\Theta(\cdot)$ is the Euler angle, and (n_i, m_i) are two indexes. The RotMap and RotPooling layers mimic the convolution and pooling layers, while the LogMap layer map rotation matrices into tangent space for classification. In the official Matlab implementation, the LogMap layer is implemented as the Euler axis-angle representation. The classification is performed by Euler axis-angle + FC + Softmax. As the axis-angle is an equivalent representation of matrix logarithm, we call this classifier as **LogEig MLR** as well. This classifier is, therefore, also non-intrinsic.

In LieNet, each rotation feature has a shape of [num, frame, 3, 3], where num and frame denote spatial and temporal dimensions. The RotPooling layer is performed either along spatial or temporal dimensions, while the RotMap layer is performed along spatial dimensions, *i.e.*,W^k with a size of [num, 3, 3].

G.2.2 Datasets and preprocessing

For a fair comparison, we follow LieNet to use G3D [6] and HDM05 datasets to validate our Lie MLR.

G3D[6]: This dataset consists of 663 sequences of 20 different gaming actions. Each sequence is recorded by 3D locations of 20 joints (i.e., 19 bones).

HDM05: We trim it down by removing some under-represented sequences, resulting in 2,326 sequences scattered throughout 122 classes. Following [30], we use the code of [67] to represent each skeleton sequence as a point on the Lie group $\mathrm{SO}^{N\times T}(3)$, where N and T denote spatial and temporal dimensions. As preprocessed in [30], we set T as 100 and 16 for each sequence on the G3D and HDM05 datasets, respectively.

G.2.3 Implementation details

LieNet: Note that the official code of LieNet[10] is developed by Matlab. We follow the open-sourced Pytorch code[11] to implement our experiments. To reproduce LieNet more faithfully, we made the following modifications to this Pytorch code. We re-code the LogMap and RotPooling layers to make them consistent with the official Matlab implementation. In addition, we also extend the existing Riemannian optimization package geoopt [4] into $\mathrm{SO}(3)$ to allow for Riemannian version of SGD, ADAM, and AMSGrad on $\mathrm{SO}(3)$, which is missing in the current package. However, we find that SGD is the best optimizer for LieNet. Therefore, we use SGD as our optimizer during the experiments.

Lie MLR: We use our Lie MLR to replace the axis-angle classifier in LieNet and call the resulting network LieNet+LieMLR. To alleviate the computational burden, we set each P_k as the dimension of [num, 3, 3], where num is the spacial dimension of the input of the Lie MLR layer. In other words, P_k is shared in the temporal dimension. We adopt Pytroch3D [53] to calculate the matrix logarithm. Due to the instabilities of `pytorch3d.transforms.so3_log_map`, we use `pytorch3d.transforms.matrix_to_axis_angle` first to calculate the quaternion axis and angle, and then convert this representation into matrix logarithm[12].

Training Details: Following [31], we focus on the 3Blocks and 2Blocks architecture for the G3D and HDM05 datasets, which are the suggested architectures for these two datasets. The learning rate is $1e^{-2}$ on both datasets, and we further set weight decay as $1e^{-5}$ on the G3D dataset. For LieNet and LieNet+LieMLR, we use `torch.nn.utils.clip_grad_norm_` for gradient clipping with a clipping factor of 5. The clipping is imposed to the dimensionality reduction weight in the final FC linear on LieNet, or, accordingly, $A = \{A_1, \cdots, A_k\}$ in the Lie MLR layer on LieNet+LieMLR.

Scoring Metrics: For the G3D dataset, following LieNet [31], we adopt a 10-fold cross-subject test setting, where half the subjects are used for training and the other half are employed for testing. For the HDM05 dataset, following [31], we randomly select half of the sequences for training and the rest for testing. Due the instabilities of LieNet, we conduct 20-fold experiments and select the best 10 folds to evaluate the performance.

G.3 Hardware

All experiments use an Intel Core i9-7960X CPU with 32GB RAM and an NVIDIA GeForce RTX 2080 Ti GPU.

H Proofs

H.1 Proof of Thm. 3.2

Proof of Thm. 3.2. Let us first solve Y^* in Eq. (8), which is the solution to the following constrained optimization problem:

$$\max_{Y} \left(\frac{\langle \mathrm{Log}_P Y, \mathrm{Log}_P S\rangle_P}{\| \mathrm{Log}_P Y\|_P, \| \mathrm{Log}_P S\|_P} \right) \quad \text{s.t.} \langle \mathrm{Log}_P S, \tilde{A}\rangle_P = 0 \tag{61}$$

[10] https://github.com/zhiwu-huang/LieNet

[11] https://github.com/hjf1997/LieNet

[12] https://github.com/facebookresearch/pytorch3d/issues/188

Note that Eq. (61) is well-defined due to the existence of the Riemannian logarithm. Although, Eq. (61) is normally non-convex, Eq. (61) and Eq. (8) can be reduced to a Euclidean problem:

$$\max_{\tilde{Y}} \frac{\langle \tilde{Y}, \tilde{S} \rangle_P}{\|\tilde{Y}\|_P \|\tilde{S}\|_P} \quad \text{s.t.} \langle \tilde{Y}, \tilde{A} \rangle_P = 0, \tag{62}$$

$$d(S, \tilde{H}_{\tilde{A},P}) = \sin(\angle SPY^*) \|\tilde{S}\|_P, \tag{63}$$

where $\tilde{Y} = \operatorname{Log}_P Y$ and $\tilde{S} = \operatorname{Log}_P S$.

Let us first discuss Eq. (62). Denote the solution of Eq. (62) as $\tilde{Y}^*$. Note that $\tilde{Y}^*$ is not necessarily unique. Note that Exp_P is only well-defined locally. More precisely, Exp_P is well-defined in an open ball $\mathrm{B}_\epsilon(0)$ centered at $0 \in T_P\mathcal{M}$. Therefore, $\tilde{Y}^*$ might not be in $\mathrm{B}_\epsilon(0)$. In this case, we can scale $\tilde{Y}^*$ into $\mathrm{B}_\epsilon(0)$, and the scaled $\tilde{Y}^*$ is still the maximizer of Eq. (62). Therefore, w.l.o.g., we assume $\tilde{Y}^* \in \mathrm{B}_\epsilon(0)$.

Putting $\tilde{Y}^*$ into Eq. (63), Eq. (63) is reduced to the distance to the hyperplane $\langle \tilde{Y}, \tilde{A} \rangle_P = 0$ in the Euclidean space $\{T_P\mathcal{M}, \langle \cdot, \cdot \rangle_P\}$, which has a closed-form solution:

$$d(S, \tilde{H}_{\tilde{A},P}) = \frac{|\langle \tilde{S}, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P}, \tag{64}$$

$$= \frac{|\langle \operatorname{Log}_P S, \tilde{A} \rangle_P|}{\|\tilde{A}\|_P}. \tag{65}$$

□

H.2 Proof of Thm. 3.3

Proof for Thm. 3.3 . Putting the margin distance (Eq. (10)) into Eq. (4), we have the following:

$$\begin{aligned} p(y = k \mid S) &\propto \exp\left(\operatorname{sign}(\langle \tilde{A}_k, \operatorname{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} d(S, \tilde{H}_{\tilde{A}_k, P_k})\right), \\ &= \exp\left(\operatorname{sign}(\langle \tilde{A}_k, \operatorname{Log}_{P_k}(S) \rangle_{P_k}) \|\tilde{A}_k\|_{P_k} \frac{|\langle \operatorname{Log}_{P_k}(S), \tilde{A}_k \rangle_{P_k}|}{\|\tilde{A}_k\|_{P_k}}\right), \\ &= \exp\left(\langle \operatorname{Log}_{P_k} S, \tilde{A}_k \rangle_{P_k}\right). \end{aligned} \tag{66}$$

□

H.3 Proof of Prop. 4.1

Proof for Prop. 4.1 . The Riemannian metric (α, β)-EM at I is

$$g_I^{(\alpha,\beta)\text{-EM}}(V, V) = \langle V, V \rangle^{(\alpha,\beta)}. \tag{67}$$

By Lem. E.2, we have the following

$$\begin{aligned} g_P^{(\theta,\alpha,\beta)\text{-EM}}(V, V) &\xrightarrow{\theta \to 0} g_I^{(\alpha,\beta)\text{-EM}}\left(\log_{*,P}(V), \log_{*,P}(V)\right) \\ &= \langle \log_{*,P}(V), \log_{*,P}(V) \rangle^{(\alpha,\beta)} \\ &= g_P^{(\alpha,\beta)\text{-LEM}}(V, V). \end{aligned} \tag{68}$$

□

H.4 Proof of Thm. 4.2

As the five families of metrics presented in Thm. 4.2 are pullback metrics, we first present a general result regarding Riemannian MLRs under pullback metrics.

Lemma H.1 (Riemannian MLRs under Pullback Metrics)**.** *Supposing $\{\mathcal{N}, g\}$ is a Riemannian manifold and $\phi : \mathcal{M} \to \mathcal{N}$ is a diffeomorphism between manifolds, the Riemannian MLR by parallel transportation (Eq. (11) + Eq. (12)) on $\mathcal{M}$ under $\tilde{g} = \phi^* g$ can be obtained by g:*

$$p(y = k \mid S \in \mathcal{M}) \propto \exp\left[\langle \tilde{\mathrm{Log}}_{P_k} S, \tilde{\Gamma}_{Q \to P_k} A_k \rangle_{P_k}\right], \tag{69}$$

$$= \exp\left[\langle \mathrm{Log}_{\phi(P_k)} \phi(S), \tilde{A}_k \rangle_{\phi(P_k)}\right], \tag{70}$$

where $\tilde{A}_k = \Gamma_{\phi(Q) \to \phi(P_k)} \phi_{*,Q}(A_k)$ with $A_k \in T_Q \mathcal{M}$, $\tilde{\mathrm{Log}}, \tilde{\Gamma}$ are Riemannian logarithm and parallel transportation under $\tilde{g}$, and Log, Γ are the counterparts under g.

Furthermore, if $\mathcal{N}$ has a Lie group operation $\odot$, $\mathcal{M}$ could be endowed with a Lie group structure $\tilde{\odot}$ by f. The Riemannian MLR by left translation (Eq. (11) + Eq. (13)) on $\mathcal{M}$ under $\tilde{g}$ and $\tilde{\odot}$ can be calculated by g and $\odot$:

$$p(y = k \mid S \in \mathcal{M}) \propto \exp\left[\langle \tilde{\mathrm{Log}}_{P_k} S, \tilde{L}_{\tilde{R}_k *, Q} A_k \rangle_{P_k}\right], \tag{71}$$

$$= \exp\left[\langle \mathrm{Log}_{\phi(P_k)} \phi(S), \tilde{A}_k \rangle_{\phi(P_k)}\right], \tag{72}$$

where $\tilde{A}_k = L_{R_k *, \phi(Q)} \circ \phi_{*,Q}(A_k)$, $\tilde{R}_k = P_k \tilde{\odot} Q_{\tilde{\odot}}^{-1}$, $R_k = \phi(P) \odot \phi(Q)_{\odot}^{-1}$, and $\tilde{L}_{P_k \tilde{\odot} Q_{\tilde{\odot}}^{-1}}$ is the left translation under $\tilde{\odot}$.

Proof for Lem. H.1. Before starting, we should point out that since ϕ is a diffeomorphism, $\tilde{\odot}$ and $\tilde{g}$ are indeed well defined, and $\{\mathcal{M}, \tilde{g}\}$ forms a Riemannian manifold and $\{\mathcal{M}, \tilde{\odot}\}$ forms a Lie group. We denote ϕ_*^{-1} as the differential of ϕ^{-1}. We first focus on the Riemannian MLR by parallel transportation:

$$\begin{aligned}
&p(y = k \mid S \in \mathcal{M}) \\
&\propto \exp(\tilde{g}_{P_k}(\tilde{\mathrm{Log}}_{P_k} S, \tilde{\Gamma}_{Q \to P_k} A_k)) \\
&= \exp\left[g_{\phi(P_k)}\left(\phi_{*,P_k} \circ \phi_{*,\phi(P_k)}^{-1} \mathrm{Log}_{\phi(P_k)} \phi(S), \phi_{*,P_k} \circ \phi_{*,\phi(P_k)}^{-1} \Gamma_{\phi(Q) \to \phi(P_k)} \phi_{*,Q}(A_k)\right)\right] \\
&= \exp\left[g_{\phi(P_k)}(\mathrm{Log}_{\phi(P_k)} \phi(S), \Gamma_{\phi(Q) \to \phi(P_k)} \phi_{*,Q}(A_k))\right].
\end{aligned} \tag{73}$$

In the case of the Riemannian MLR by left translation, we first note that:

$$\tilde{L}_{\tilde{R}_k} = \phi^{-1} \circ L_{\phi(P_k) \odot \phi(Q)_{\odot}^{-1}} \circ \phi. \tag{74}$$

Therefore, the associated differential is:

$$\tilde{L}_{\tilde{R}_k *} = \phi_*^{-1} \circ L_{\phi(P_k) \odot \phi(Q)_{\odot}^{-1} *} \circ \phi_*. \tag{75}$$

Putting Eq. (75) in Eq. (71), we can obtain the results. □

Now, we apply Lem. H.1 to derive the expressions of our SPD MLRs presented in Thm. 4.2. For our cases of SPD MLRs, we set $Q = I$. For simplicity, we will omit the subscript k for P_k and A_k. We will first derive the expressions of SPD MLRs under (θ, α, β)-LEM, θ-LCM, (θ, α, β)-EM, and (θ, α, β)-AIM, as they are sourced from Eq. (70). Then we will proceed to present the expression of MLR under 2θ-BWM, which is sourced from Eq. (72). According to Lem. E.3, the scaled metric ag shares the same Riemannian operators as g. We will use this fact throughout the following proof.

Proof of Thm. 4.2. For simplicity, we abbreviate ϕ_θ as ϕ during the proof. Note that for 2θ-BWM, ϕ should be understood as $\phi_{2\theta}$. We first show $\phi(I)$ and the differential map $\phi_{*,I}$, which will be frequently required in the following proof:

$$\phi(I) = I, \tag{76}$$

$$\phi_{*,I}(A) = \theta A, \forall A \in T_I \mathcal{S}_{++}^n. \tag{77}$$

Denoting $\phi : \{\mathcal{S}_{++}^n, \tilde{g}\} \to \{\mathcal{S}_{++}^n, g\}$, then the SPD MLR under $\tilde{g}$ by parallel transportation with $Q = I$ is

$$p(y = k \mid S \in \mathcal{M}) = \exp\left[g_{\phi(P)}(\mathrm{Log}_{\phi(P)} \phi(S), \Gamma_{I \to \phi(P)} \theta A)\right], \tag{78}$$

Next, we begin to prove the five SPD MLRs one by one.

(α,β)-LEM: As shown by Chen et al. [20], the standard LEM is the pullback metric from the Euclidean space $\mathcal{S}^n$. Similarly, (α,β)-LEM is also a pullback metric:

$$\{\mathcal{S}_{++}^n, g^{(\alpha,\beta)\text{-LEM}}\} \xrightarrow{\log} \{\mathcal{S}^n, g^{(\alpha,\beta)}\} \tag{79}$$

By Eq. (70), we have

$$p(y=k \mid S \in \mathcal{M}) = \exp\left[\langle \log(S) - \log(P), \log_{*,I}(A)\rangle^{(\alpha,\beta)}\right] \tag{80}$$

$$= \exp\left[\langle \log(S) - \log(P), A\rangle^{(\alpha,\beta)}\right]. \tag{81}$$

θ-LCM: Simple computations show that θ-LCM is the scaled pullback metric of standard Euclidean metric in the Euclidean space of lower triangular matrices $\mathcal{L}^n$:

$$\{\mathcal{S}_{++}^n, \theta^2 g^{\theta\text{-LCM}}\} \xrightarrow{\phi} \{\mathcal{S}_{++}^n, g^{\text{LCM}}\} \xrightarrow{\text{Chol}} \{\mathcal{L}_{+}^n, g^{\text{CM}}\} \xrightarrow{\text{Dlog}} \{\mathcal{S}^n, g^{\text{E}}\}, \tag{82}$$

where g^{E} is the standard Frobenius inner product, and g^{CM} is the Cholesky metric on the Cholesky space $\mathcal{L}_+^n$ [41]. Denoting $\zeta = \text{Dlog} \circ \text{Chol} \circ \phi$, then we have

$$\zeta_{*,I}(A) = \theta\left(\lfloor A \rfloor + \frac{1}{2}\mathbb{D}(A)\right), \forall A \in T_I\mathcal{S}_{++}^n. \tag{83}$$

Similar with the case of (θ,α,β)-LEM, we have

$$p(y=k \mid S \in \mathcal{M}) \propto \exp\left[\frac{1}{\theta^2}\langle \zeta(S) - \zeta(P), \zeta_{*,I}A\rangle\right], \tag{84}$$

$$= \exp\left[\frac{1}{\theta}\langle \lfloor \tilde{K} \rfloor - \lfloor \tilde{L} \rfloor + \left[\text{Dlog}(\mathbb{D}(\tilde{K})) - \text{Dlog}(\mathbb{D}(\tilde{L}))\right], \lfloor A \rfloor + \frac{1}{2}\mathbb{D}(A)\rangle\right], \tag{85}$$

where $\tilde{K} = \text{Chol}(S^\theta)$, $\tilde{L} = \text{Chol}(P^\theta)$, $\mathbb{D}(\tilde{K})$ is a diagonal matrix with diagonal elements from $\tilde{K}$, and $\lfloor \tilde{K} \rfloor$ is a strictly lower triangular matrix from $\tilde{K}$.

(θ,α,β)-EM: Let $\eta = \frac{1}{|\theta|}\phi$. Simple computation shows that (θ,α,β)-EM is the pullback metric of (α,β)-EM:

$$\{\mathcal{S}_{++}^n, g^{(\theta,\alpha,\beta)\text{-EM}}\} \xrightarrow{\eta} \{\mathcal{S}_{++}^n, g^{(\alpha,\beta)\text{-EM}}\}. \tag{86}$$

Besides, we have the following for η:

$$\eta_{*,I}(A) = \text{sgn}\,\theta A, \forall A \in T_I\mathcal{S}_{++}^n. \tag{87}$$

According to Eq. (70), we have

$$p(y=k \mid S \in \mathcal{M}) \propto \exp\left[\langle \eta(S) - \eta(P), \text{sgn}(\theta)A\rangle\right], \tag{88}$$

$$= \exp\left[\frac{1}{\theta}\langle S^\theta - P^\theta, A\rangle^{(\alpha,\beta)}\right]. \tag{89}$$

(θ,α,β)-AIM: Putting $g^{(\alpha,\beta)\text{-AIM}}$ into Eq. (78), we have

$$p(y=k \mid S \in \mathcal{M}) \propto \exp\left[\frac{1}{\theta^2} g_{\phi(P)}^{(\alpha,\beta)\text{-AIM}}(P^{\frac{\theta}{2}}\log(P^{-\frac{\theta}{2}}S^\theta P^{-\frac{\theta}{2}})P^{\frac{\theta}{2}}, P^{\frac{\theta}{2}}\theta A P^{\frac{\theta}{2}})\right], \tag{90}$$

$$= \exp\left[\frac{1}{\theta}\langle \log(P^{-\frac{\theta}{2}}S^\theta P^{-\frac{\theta}{2}}), A\rangle^{(\alpha,\beta)}\right]. \tag{91}$$

2θ-BWM: We first simplify Eq. (72) under the cases of SPD manifolds and then proceed to focus on the case of $g = g^{\text{BWM}}$. Denote $\phi : \{\mathcal{S}_{++}^n, \tilde{g}, \tilde{\odot}\} \to \{\mathcal{S}_{++}^n, g, \odot\}$, where the Lie group operation $\odot$ [62] is defined as

$$S_1 \odot S_2 = L_1 S_2 L_1^T, \forall S_1, S_2 \in \mathcal{S}_{++}^n, \text{ with } L_1 = \text{Chol}(S_1). \tag{92}$$

Note that I is the identity element of $\{\mathcal{S}_{++}^n, \odot\}$, and for any $S \in \mathcal{S}_{++}^n$, the differential map of the left translation L_S under $\odot$ is

$$L_{S*,Q}(V) = LVL^\top, \forall Q \in \mathcal{S}_{++}^n, \forall V \in T_Q\mathcal{S}_{++}^n, \text{ with } L = \mathrm{Chol}(S). \tag{93}$$

For the induced Lie group $\{\mathcal{S}_{++}^n, \tilde{\odot}\}$, the left translation $\tilde{L}_{P\tilde{\odot}I_{\tilde{\odot}}^{-1}}$ under $\tilde{\odot}$ is

$$\tilde{L}_{P\tilde{\odot}I_{\tilde{\odot}}^{-1}} = \phi^{-1} \circ L_{\phi(P)\odot\phi(I)_{\odot}^{-1}} \circ \phi, \tag{94}$$

$$= \phi^{-1} \circ L_{P^{2\theta}} \circ \phi. \quad (\phi(P) \odot \phi(I)_{\odot}^{-1} = P^{2\theta}) \tag{95}$$

The associated differential at I is

$$\tilde{L}_{P\tilde{\odot}I_{\tilde{\odot}}^{-1}*,I}(A) = \phi^{-1}_{*,\phi(P)} \circ L_{P^{2\theta}*,\phi(I)} \circ \phi_{*,I}(A), \tag{96}$$

$$= 2\theta\phi^{-1}_{*,\phi(P)}(\bar{L}A\bar{L}^\top), \tag{97}$$

where $\bar{L} = \mathrm{Chol}(P^{2\theta})$. Then the SPD MLRs under $\tilde{g}$ and $\tilde{\odot}$ by left translation is

$$p(y = k \mid S \in \mathcal{M}) = \exp\left[2\theta g_{\phi(P)}\left(\mathrm{Log}_{\phi(P)}\,\phi(S), \bar{L}A\bar{L}^\top\right)\right], \tag{98}$$

Setting $g = g^{\mathrm{BWM}}$ (We omit the scaling factor.), we obtain the SPD MLR under 2θ-BWM:

$$p(y = k \mid S \in \mathcal{M}) = \exp\left[2\theta \cdot \frac{1}{4\theta^2} g^{\mathrm{BWM}}_{\phi(P)}\left(\mathrm{Log}^{\mathrm{BWM}}_{\phi(P)}\,\phi(S), \bar{L}A\bar{L}^\top\right)\right], \tag{99}$$

$$= \exp\left[\frac{1}{4\theta}\langle (P^{2\theta}S^{2\theta})^{\frac{1}{2}} + (S^{2\theta}P^{2\theta})^{\frac{1}{2}} - 2P^{2\theta}, \mathcal{L}_{P^{2\theta}}(\bar{L}A\bar{L}^\top)\rangle\right]. \tag{100}$$

□

H.5 Proof of Lem. 5.1

Proof of Lem. 5.1. During this proof, we use the ambient representation of tangent vectors. We only need to prove the following:

$$\Gamma_{Q\to P} = L_{PQ^{-1}*,Q}, \forall P, Q \in \mathrm{SO}(n). \tag{101}$$

Given rotation matrices P, Q and a tangent vector $H \in T_Q\mathrm{SO}(n)$, the parallel transportation [8, Tab. 1] is

$$\Gamma_{Q\to P}(H) = PQ^\top H = PQ^{-1}H. \tag{102}$$

On the other hand, given a curve $c(t)$ over $\mathrm{SO}(n)$, satisfying $c(0) = Q$ and $c'(0) = H$, the differential of the left translation $L_{PQ^{-1}}$ at Q is

$$L_{PQ^{-1}*,Q}(H) = \left.\frac{dPQ^{-1}c(t)}{dt}\right|_{t=0} = PQ^{-1}H, \tag{103}$$

which concludes the proof. □

H.6 Proof of Thm. 5.2

Proof of Thm. 5.2. Lem. 5.1 indicates we can use either parallel transportation or group translation. Putting the associated expressions from Tab. 13 into Eq. (11) + Eq. (12), one can directly obtain the results. □

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our abstract and introduction (Sec. 1) accurately reflect the paper's theoretical and empirical contributions.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations specifically in App. A.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Assumptions are clearly claimed in each theorem, and all the proofs are presented in App. H.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Implementation details are discussed in App. G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets (Apps. G.1.2 and G.2.2) are publicly available. The code will be released after the review.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In App. G, we present the experimental details for reproducing the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Mean, STD, and max of K-fold results are presented in Sec. 6.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware is mentioned in App. G.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: No ethic issue.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: No societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Original papers and datasets have been cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Code will be released after the review.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.