# Vector-valued Distance and Gyrocalculus on the Space of Symmetric Positive Definite Matrices

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

We propose the use of the vector-valued distance to compute distances and extract geometric information from the manifold of symmetric positive definite matrices (SPD), and develop gyrovector calculus, constructing analogs of vector space operations in this curved space. We implement these operations and showcase their versatility in the tasks of knowledge graph completion, item recommendation, and question answering. In experiments, the SPD models outperform their equivalents in Euclidean and hyperbolic space. The vector-valued distance allows us to visualize embeddings, showing that the models learn to disentangle representations of positive samples from negative ones.

## 1 Introduction

Symmetric Positive Definite (SPD) matrices have been applied in many tasks in computer vision such as pedestrian detection [70, 66], action [32, 45, 51] or face recognition [36, 37], object [38, 81] and image set classification [77], visual tracking [78], and medical imaging analysis [56, 3] among others. They have been used to capture statistical notions (Gaussian distributions [59], covariance [69]), while respecting the Riemannian geometry of the underlying SPD manifold, which offers a convenient trade-off between structural richness and computational tractability [23]. Previous work has applied approximation methods that locally flatten the manifold by projecting it to its tangent space [18, 74], or by embedding the manifold into higher dimensional Hilbert spaces [31, 81].

These methods face problems such as distortion of the geometrical structure of the manifold and other known concerns with regard to high-dimensional spaces [25]. To overcome these issues, several distances on SPD manifolds have been proposed, such as the Affine Invariant metric [56], the Stein metric [62], the Bures–Wasserstein metric [10] or the Log-Euclidean metric [3, 4], with their respective geometric properties. However, the representational power of SPD is not fully exploited in many cases [56, 4]. At the same time, it is hard to translate operations into their non-Euclidean domain given the lack of closed-form expressions. There has been a growing need to generalize basic operations, such as addition, rotation, reflection or scalar multiplication, to their Riemannian geometric counterparts to leverage this structure in the context of Geometric Deep Learning [15].

In this work, we introduce the vector-valued distance function to exploit the full representation power of SPD (§3.1). While in Euclidean or hyperbolic space the relative position between two points is completely captured by their distance (and this is the only invariant), in SPD this invariant is a vector, encoding much more information than a single scalar. This vector reflects the higher expressivity of SPD due to its richer geometry encompassing Euclidean as well as hyperbolic spaces, which it both contains. We develop algorithms using the vector-valued distance and showcase two main advantages: its versatility to implement universal models, and its use in explaining and visualizing what the model has learned.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

Furthermore, we bridge the gap between Euclidean and SPD geometry by developing gyrocalculus in SPD (§4), which yields closed-form expressions of arithmetic operations, such as addition, scalar multiplication and matrix scaling. This provides means to translate previously implemented ideas in different metric spaces to their analog notions in SPD.

We showcase this on knowledge graph completion, item recommendation, and question answering. In the experiments, the proposed SPD models outperform their equivalents in Euclidean and hyperbolic space (§6). These results reflect the superior expressivity of SPD, and show the versatility of the approach and ease of integration with downstream tasks.

The vector-valued distance allows us to develop a new tool for the analysis of the structural properties of the learned representations. With this tool, we visualize high-dimensional SPD embeddings, providing better explainability on what the models learn (§6.4). We show that the knowledge graph models are capable of disentangling and clustering positive triples from negative ones.

## 2 Related Work

Symmetric positive definite matrices are not new in the Machine Learning literature. They have been used in a plethora of applications [3, 25, 32, 35–38, 45, 51, 56, 59, 69, 70, 66, 77, 78, 81]), although not always respecting the intrinsic structure or the positive definiteness constraint [18, 27, 31, 74, 81]. The alternative has been to map manifold points onto a tangent space and employ Euclidean-based tools. Unfortunately, this mapping distorts the manifold structure in regions far from the origin of the tangent space affecting the performance [39, 87].

Previous work has proposed alternatives to the basic neural building blocks respecting the geometry of the space. For example, transformation layers [25, 29, 35], alternate convolutional layers based on SPDs [85] and Riemannian means [19], or appended after the convolution [17], recurrent models [20], projections onto Euclidean spaces [44, 48] and batch normalization [16]. Our work follows this line, providing explicit formulas for translating Euclidean arithmetic notions into SPDs.

Our general view, using the vector-valued distance function, allows us to treat Riemannian and Finsler metrics on SPD in a unified framework. Finsler metrics have previously been applied in compressed sensing [26], information geometry [60], for clustering categorical distributions [54], and in robotics [58]. With regard to optimization, matrix backpropagation techniques have been explored [38], with some of them accounting for different Riemannian geometries [35, 16]. Nonetheless, we opt for tangent space optimization [22] by exploiting the explicit formulations of the exponential and logarithmic map.

## 3 The Space $\mathrm{SPD}_n$

The space $\mathrm{SPD}_n$ is a Riemannian manifold of non-positive curvature of $n(n+1)/2$ dimensions. Points in $\mathrm{SPD}_n$ are positive definite real symmetric $n \times n$ matrices, with the identity matrix $I$ being a natural basepoint. The tangent space to any point of $\mathrm{SPD}_n$ can be identified with the vector space $S_n$ of all real symmetric $n \times n$ matrices. $\mathrm{SPD}_n$ contains $n$-dimensional Euclidean subspaces, $(n-1)$-dimensional hyperbolic subspaces as well as products of $\lfloor \frac{n}{2} \rfloor$ hyperbolic planes.

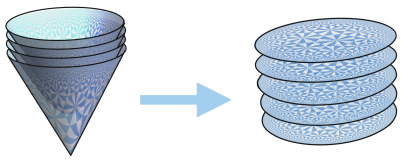

Figure 1: $\mathrm{SPD}_2$ is foliated by hyperboloids, each of which is a copy of the hyperbolic plane.

In Figure 1 we visualize the smallest nontrivial example. $\mathrm{SPD}_2$ identifies with the inside of a cone in $\mathbb{R}^3$, cut out by requiring both eigenvalues of the matrix $\left(\begin{smallmatrix} x & y \\ y & z \end{smallmatrix}\right)$ to be positive. It carries the product geometry of the hyperbolic plane times a line.

**Exponential and logarithmic maps:** The exponential map, $\exp\colon S_n \to \mathrm{SPD}_n$, gives a connection between the Euclidean geometry of the tangent space $S_n$ and the curved geometry of $\mathrm{SPD}_n$. Its inverse is the logarithm map, $\log\colon \mathrm{SPD}_n \to S_n$. Since we apply them based at $I \in \mathrm{SPD}_n$, this pair of functions forms a diffeomorphism that allows one to freely move between 'tangent space coordinates' or the original 'manifold coordinates'. We prove this in Appendix C.3.

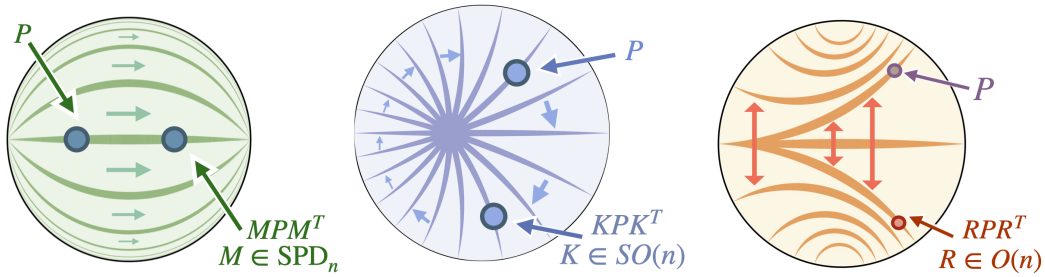


Figure 2: Some isometries of $\mathrm{SPD}_n$ have analogous Euclidean counterparts. Translation (left), rotation (center) and reflection (right).

**Symmetries:** The prototypical symmetries of $\mathrm{SPD}_n$ are parameterized by elements of $GL(n;\mathbb{R})$: any invertible matrix $M$ defines the symmetry $P \mapsto MPM^T$ acting on all points $P \in \mathrm{SPD}_n$. Thus many geometric transformations $\mathrm{SPD}_n$ can be completed using standard optimized matrix algorithms as opposed to custom-built procedures. See Appendix C.2. for a brief review of these symmetries.

Among these, we may find $\mathrm{SPD}_n$-generalizations of familiar symmetries of Euclidean geometry. When also $M$ is an element of $\mathrm{SPD}_n$, the symmetry $P \mapsto MPM^T$ is a generalization of an Euclidean *translation*, fixing no points of $\mathrm{SPD}_n$ . When $M$ is an orthogonal matrix, the symmetry $P \mapsto MPM^T$ is conjugation by $M$, and thus fixes the basepoint $I = MIM^T = MM^{-1} = I$. We think of elements fixing the basepoint as being $\mathrm{SPD}_n$*-rotations* or $\mathrm{SPD}_n$*-reflections*, when the matrix $M$ is a familiar rotation or reflection (see Figure 2).

The Euclidean symmetry of *reflecting in a point* also has a natural generalization to $\mathrm{SPD}_n$. Euclidean reflection in the origin is given by $p \mapsto -p$; and its $\mathrm{SPD}_n$-analog, reflection in the basepoint $I$, is matrix inversion $P \mapsto P^{-1}$. The general $\mathrm{SPD}_n$-reflection in a point $Q \in \mathrm{SPD}_n$ is a conjugate of this by an $\mathrm{SPD}_n$ translation, given by $P \mapsto QP^{-1}Q$.

### 3.1 Vector-valued Distance Function

In Euclidean or hyperbolic spaces, the relative position between two points is completely determined by their distance, which is given by a scalar. For the space $\mathrm{SPD}_n$, it is determined by a vector. To assign this vector in SPD we introduce the vector-valued distance (VVD) function $d_{vv}\colon \mathrm{SPD}_n \times \mathrm{SPD}_n \to \mathbb{R}^n$. For two points $P, Q \in \mathrm{SPD}_n$, the VVD is defined as:

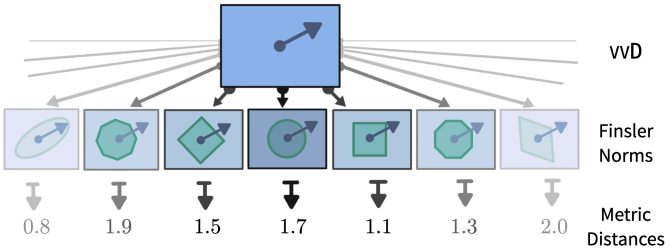


Figure 3: The vector-valued distance allows to reconstruct the Riemannian, or any Finsler distance.

$$d_{vv}(P,Q) = \log(\lambda_1(P^{-1}Q), \ldots, \lambda_n(P^{-1}Q)) \tag{1}$$

where $\lambda_1(P^{-1}Q) \geq \ldots \geq \lambda_n(P^{-1}Q)$ are the eigenvalues of $P^{-1}Q$ sorted in descending order. This vector is an invariant of the relative position of two points up to isometry. This means that we can recover completely the relative position of two points in $\mathrm{SPD}_n$ from this vector, and any pair of points can be mapped to each other by an isometry if and only if they have the same vector-valued distance. In this case, the Riemannian metric is obtained by using the standard $l_2$ norm on the VVD vector. This is: $d^R(P,Q) = ||d_{vv}(P,Q)||_2$. See [41] §2.6 and Appendix C.4 for a review of VVDs in symmetric spaces. In Appendix C.7, we detail how the VVD generalizes other SPD metrics.

**Finsler metrics:** Any norm on $\mathbb{R}^n$ that is invariant under permutation of the entries induces a metric on $\mathrm{SPD}_n$. Moreover, $\mathrm{SPD}_n$ do not only support a Riemannian metric, but also Finsler metrics, a whole family of distances with the same symmetry group (group of isometries). These metrics are of special importance since distance minimizing geodesics are not necessarily unique in Finsler geometry. Two different paths can have the same minimal length. This is particularly valuable when embedding graphs in $\mathrm{SPD}_n$, since in graphs there are generally several shortest paths. We obtain the

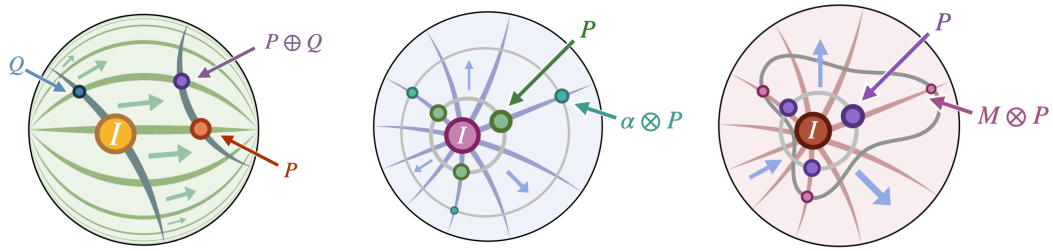


Figure 4: Gyro-addition (left), gyro-scalar multiplication (center) and matrix scaling (right).

Finsler metrics $\mathrm{F}_1$ or $\mathrm{F}_\infty$ by taking the respective $\ell_1$ or $\ell_\infty$ norms of the VVD in $\mathbb{R}^n$ (see Figure 3). See Planche [57] and Appendix C.6 for a review of the theory of Finsler metrics.

The proposed metric learning framework based on the VVD offers several advantages. First, a single model can be run with different metrics, according to the chosen norm. We can easily recover the Riemannian metric and extend the approach to many other variants. Second, the VVD provides much more information than just the distance, and can be used to analyze learned representation in $\mathrm{SPD}_n$, independent of the choice of the metric. Third, it can be leveraged to visualize the learned high-dimensional representations (see §6.4).

## 4 Gyrocalculus

To build an analog of many Euclidean operators in $\mathrm{SPD}_n$, we require also a translation of operations internal to Euclidean geometry, chief among these being the vector space operations of addition and scalar multiplication. We describe a gyro-vector space structure on $\mathrm{SPD}_n$, which provides geometrically meaningful extensions of these vector space operations, extending successful applications of this framework in geometric deep learning to hyperbolic space [28, 61]. These operations provide a template for translation, where one may attempt to replace $+, -, \times$ in formulas familiar from Euclidean spaces with the analogous operations $\oplus, \ominus, \otimes$ on $\mathrm{SPD}_n$. While straightforward, such translation requires some care, as gyro-addition is neither commutative nor associative. See Appendix D for a review of the underlying general theory of gyrogroups and additional guidelines for accurate formula translation.

**Addition and Subtraction:** Given a fixed choice $I$ of basepoint and two points $P, Q \in \mathrm{SPD}_n$, we define the gyroaddition of $P$ and $Q$ to be the point $P \oplus Q \in \mathrm{SPD}_n$ which is the image of $Q$ under the isometry which translates $I$ to $P$ along the geodesic connecting them. This directly generalizes the gyroaddition of hyperbolic space exploited by [28, 5, 61], via the geometric interpretation of Vermeer [75] (see Figure 4).

Fixing $P \in \mathrm{SPD}_n$, we may compute the value of $P \oplus Q$ for arbitrary $Q$ as the result of applying the $\mathrm{SPD}_n$-translation moving the basepoint to $P$, evaluated on $Q$. We see also the additive inverse of a point with respect to this operation must then be given by its geodesic reflection in $I$.

$$P \oplus Q = \sqrt{P} Q \sqrt{P} \qquad \ominus P = P^{-1} \tag{2}$$

As this operation encodes a symmetry of $\mathrm{SPD}_n$, it is possible to recast certain geometric statements purely in the gyrovector formalism. In particular, the vector-valued distance $d_{vv}(P, Q)$ may be computed as the logarithm of the eigenvalues of $\ominus P \oplus Q$ (see Appendix C.5).

**Scalar Multiplication and Matrix Scaling:** For a fixed basepoint $I$, we define the scalar multiplication of a point $P \in \mathrm{SPD}_n$ by a scalar $\alpha \in \mathbb{R}_+$ to be the point which lies at distance $\alpha d(I, P)$ from $I$ in the direction of $P$, where $d(\cdot, \cdot)$ is the metric distance on $\mathrm{SPD}_n$. Geometrically, this is a transfer of the vector-space scalar multiplication on the tangent space to $\mathrm{SPD}_n$:

$$\alpha \otimes P = P^\alpha = \exp(\alpha \log(P)), \tag{3}$$

where $\exp, \log$ are the matrix exponential and logarithm. We further generalize the notion of scalar multiplication to allow for different relative expansion rates in different directions. For a fixed basepoint $I$ and a point $P \in \mathrm{SPD}_n$, we can replace the scalar $\alpha$ from Eq. 3 with an arbitrary real symmetric matrix $A \in S_n$. We define this *matrix scaling* by:

$$A \otimes P = \exp(A \odot \log(P)) \tag{4}$$

where $A \odot X$ denotes the Hadamard product. We denote the matrix scaling with $\otimes$, extending the previous usage: for any $\alpha \in \mathbb{R}$, we have $[\alpha] \otimes P = \alpha \otimes P$ where $[\alpha]$ is the matrix with every entry $\alpha$.

# 5 Implementation

In this section we detail how we learn representations in $\mathrm{SPD}_n$, and implement different linear mappings so that they conform to the premises of each operator. The proposed mappings, along with the gyro-vector operations introduced in the previous section can be seen as feature transformations and employed as building blocks for SPD neural models.

**Points in $\mathrm{SPD}_n$ and $S_n$:** We are interested in learning embeddings in $\mathrm{SPD}_n$. To do so we exploit the connection between $\mathrm{SPD}_n$ and its tangent space $S_n$ through the exponential and logarithmic maps. To learn a point $P \in \mathrm{SPD}_n$, we first model it as a symmetric matrix $U \in S_n$. We impose symmetry on $U$ by learning a triangular matrix $X \in \mathbb{R}^{n \times n}$ with $n(n+1)/2$ parameters, such that $U = X + X^T$. To obtain the matrix $P \in \mathrm{SPD}_n$, we employ the exponential map: $P = \exp(U)$. Modeling points on the tangent space offers advantages for optimization, explained in §5. For the matrix scaling $A \otimes P$, we impose symmetry on the factor matrix $A \in S_n$ in the same way that we learn the symmetric matrix $U$.

**Isometries: Rotations and Reflections:** Rotations in $n$ dimensions are described as collections of pairwise orthogonal 2-dimensional rotations in planes (with a leftover 1-dimensional "axis of rotation" in odd dimensions). We utilize this observation to efficiently build elements of $O(n)$ out of two-dimensional rotations in coordinate planes. More precisely, for any $\theta \in [0, 2\pi)$ and choice of sign $\{+, -\}$ we let $R^{\pm}(\theta)$ denote the 2-dimensional rotation $(+)$ or reflection $(-)$ as $R^{\pm}(\theta) = \left(\begin{smallmatrix} \cos\theta & \mp\sin\theta \\ \sin\theta & \pm\cos\theta \end{smallmatrix}\right)$. Then for any pair $i < j$ in $1 \dots n$, we denote by $R^{\pm}_{ij}(\theta)$ the transformation which applies $R^{\pm}(\theta)$ to the $x_i x_j$-plane of $\mathbb{R}^n$, and leaves all other coordinates fixed. For example, in $O(5)$ the element $R^{+}_{24}(\theta)$ (see on the right) denotes the transformation where we replace the entries $(ii, ij, ji, jj)$ of $I_n$ with the corresponding values of $R^{+}(\theta)$. More general, rotations and reflections are built by taking products of these basic transformations. Given a $n(n-1)/2$-dimensional vector of angles $\vec{\theta} = (\theta_{12}, \dots, \theta_{ij}, \dots, )$ and a choice of sign, we define the rotation and reflection corresponding to $\vec{\theta}$ by:

$$R^{+}_{24}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathrm{Rot}(\vec{\theta}) = \prod_{i<j} R^{+}_{ij}(\theta_{ij}) \qquad \mathrm{Refl}(\vec{\theta}) = \prod_{i<j} R^{-}_{ij}(\theta_{ij}) \tag{5}$$

where $\mathrm{Rot}(\vec{\theta}), \mathrm{Ref}(\vec{\theta}) \in \mathbb{R}^{n \times n}$ are the isometry matrices, and the vector of angles $\vec{\theta}$ can be regarded as a learnable parameter of the model. Finally, we denote the application of the transformation $M$ to the point $P \in \mathrm{SPD}_n$ by:

$$M \circledcirc P = MPM^T \tag{6}$$

**Optimization:** For the proposed rotations and reflections, the learnable weights are vectors of angles $\vec{\theta} \in \mathbb{R}^{\frac{n(n-1)}{2}}$, which do not pose an optimization challenge. On the other hand, embeddings in SPD have to be optimized respecting the geometry of the manifold, but as explained in §5, we model them on the space of symmetric matrices $S_n$, and then we apply the exponential map. In this manner, we are able to perform tangent space optimization [22] using standard Euclidean techniques, and circumvent the need for Riemannian optimization [12, 9], which we found to be less numerically stable. Due to the geometry of $\mathrm{SPD}_n$ (see Appendix C.3), this is an exact procedure, which does not incur losses in representational power.

# 6 Experiments

In this section we employ the transformations developed on SPD to build neural models for knowledge graph completion, item recommendation and question answering. Task-specific models in different geometries have been developed in the three cases, hence we consider them adequate benchmarks for representation learning.

### 6.1 Knowledge Graph Completion

Knowledge graphs (KG) represent heterogeneous knowledge in the shape of *(head, relation, tail)* triples, where *head* and *tail* are entities and *relation* represents a relationships among entities. KG exhibit an intricate and varying structure where entities can be connected by symmetric, anti-symmetric, or hierarchical relations. To capture these non-trivial patterns more expressive modelling techniques become necessary [21], thus we choose this application to showcase the capabilities of our transformations on SPD manifolds. Given an incomplete KG, the task of is to predict which unknown links are valid.

**Problem formulation:** Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ be a knowledge graph where $\mathcal{E}$ is the set of entities, $\mathcal{R}$ is the set of relations and $\mathcal{T} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of triples stored in the graph. The usual approach is to learn a scoring function $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$ that measures the likelihood of a triple to be true, with the goal of scoring all missing triples correctly. To do so, we propose to learn representations of entities as embeddings in $\mathrm{SPD}_n$, and relation-specific transformation in the manifold, such that the KG structure is preserved.

**Scaling model:** We follow the base hyperbolic model MuRP [6] and adapt it into $\mathrm{SPD}_n$ by means of the *matrix scaling*. Its scoring function has shown success in the task given that it combines multiplicative and additive components, which are fundamental to model different properties of KG relations [2]. We translate it into $\mathrm{SPD}_n$ as:

$$\phi(h, r, t) = -d((\mathbf{M}_r \otimes \mathbf{H}) \oplus \mathbf{R}, \mathbf{T})^2 + b_h + b_t \tag{7}$$

where $\mathbf{H}, \mathbf{T} \in \mathrm{SPD}_n$ are embeddings and $b_h, b_t \in \mathbb{R}$ are scalar biases for the head and tail entities respectively. $\mathbf{R} \in \mathrm{SPD}_n$ and $\mathbf{M}_r$ are matrices that depend on the relation. For $d(\cdot, \cdot)$, we experiment with the Riemannian and the Finsler One metric distances.

**Isometric model:** A possible alternative is to embed the relation-specific transformations as elements of the $O(n)$ group (*i.e.*, rotations and reflections). This technique has proven effective in different metric spaces [79, 21]. In this case, $\mathbf{M}_r$ is a rotation or reflection matrix as in Eq. 5, and the scoring function is defined as:

$$\phi(h, r, t) = -d((\mathbf{M}_r \odot \mathbf{H}) \oplus \mathbf{R}, \mathbf{T})^2 + b_h + b_t \tag{8}$$

**Datasets:** We employ two standard benchmarks, namely WN18RR [13, 24] and FB15k-237 [13, 67]. WN18RR is a subset of WordNet [50] containing $11$ lexical relationships between $40,943$ word senses. FB15k-237 is a subset of Freebase [11], with $14,541$ entities and $237$ relationships.

**Training:** We follow the standard data augmentation protocol by adding inverse relations to the datasets [42]. We optimize the cross-entropy loss with uniform negative sampling defined in Equation 9, where

$$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{T}} \log(1 + \exp(Y_t \phi(h, r, t))) \tag{9}$$

$\mathcal{T}$ is the set of training triples, and $Y_t = -1$ if $t$ is a factual triple or $Y_t = 1$ if $t$ is a negative sample. We employ the AdamW optimizer [46]. We conduct a grid search with matrices of dimension $n \times n$ where $n \in \{14, 20, 24\}$ (this is the equivalent of $\{105, 210, 300\}$ degrees of freedom respectively) to select optimal dimensions, learning rate and weight decay, using the validation set. More details and set of hyperparameters in Appendix B.1.

**Evaluation metrics:** At test time, we rank the correct tail or head entity against all possible entities using the scoring function, and use inverse relations for head prediction [42]. Following previous work, we compute two ranking-based metrics: mean reciprocal rank (MRR), which measures the mean of inverse ranks assigned to correct entities, and hits at K (H@K, $K \in \{1, 3, 10\}$), which measures the proportion of correct triples among the top K predicted triples. We follow the standard evaluation protocol of filtering out all true triples in the KG during evaluation [13].

**Baselines:** We compare our models with their respective equivalents in different metric spaces, which are also state-of-the-art models for the task. For the scaling model, these are MuRE and MuRP [6], which perform the scaling operation in Euclidean and hyperbolic space respectively. For the isometric models, we compare to RotC [64], RotE and RotH, [21] (rotations in Complex, Euclidean and hyperbolic space respectively), and RefE and RefH [21] (reflections in Euclidean and hyperbolic space). Baseline results are taken from the original papers.

**Results** We report the performance for all analyzed models, segregated by operation, in Table 1. On both dataset, the scaling model $\mathrm{SPD}_{\mathrm{Sca}}$ outperforms its direct competitors MuRE and MuRP, and this

Table 1: Results for Knowledge graph completion.

| Operation | Model | WN18RR | | | | FB15k-237 | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | MRR | HR@1 | HR@3 | HR@10 | MRR | HR@1 | HR@3 | HR@10 |
| Scaling | MURE | 47.5 | 43.6 | 48.7 | 55.4 | 33.6 | 24.5 | 37.0 | 52.1 |
| | MURP | 48.1 | **44.0** | 49.5 | 56.6 | 33.5 | 24.3 | 36.7 | 51.8 |
| | $\mathrm{SPD}^{R}_{\mathrm{Sca}}$ | 48.1 | 43.1 | 50.1 | 57.6 | **34.5** | **25.1** | **38.0** | **53.5** |
| | $\mathrm{SPD}^{F_1}_{\mathrm{Sca}}$ | **48.4** | 42.6 | **51.0** | **59.0** | 32.9 | 23.6 | 36.3 | 51.5 |
| Rotations | ROTC | 47.6 | 42.8 | 49.2 | 57.1 | 33.8 | 24.1 | 37.5 | 53.3 |
| | ROTE | 49.4 | 44.6 | 51.2 | 58.5 | **34.6** | **25.1** | **38.1** | **53.8** |
| | ROTH | **49.6** | **44.9** | **51.4** | **58.6** | 34.4 | 24.6 | 38.0 | 53.5 |
| | $\mathrm{SPD}^{R}_{\mathrm{Rot}}$ | 46.2 | 39.7 | 49.6 | 57.8 | 32.9 | 23.6 | 36.2 | 51.5 |
| | $\mathrm{SPD}^{F_1}_{\mathrm{Rot}}$ | 40.9 | 30.5 | 48.2 | 57.3 | 32.1 | 22.9 | 35.4 | 50.5 |
| Reflections | REFE | 47.3 | 43.0 | 48.5 | 56.1 | **35.1** | **25.6** | **39.0** | **54.1** |
| | REFH | 46.1 | 40.4 | 48.5 | 56.8 | 34.6 | 25.2 | 38.3 | 53.6 |
| | $\mathrm{SPD}^{R}_{\mathrm{Ref}}$ | 48.3 | 44.0 | 49.7 | 56.7 | 32.5 | 23.4 | 35.6 | 51.0 |
| | $\mathrm{SPD}^{F_1}_{\mathrm{Ref}}$ | **48.7** | **44.3** | **50.1** | **57.4** | 31.6 | 22.5 | 34.6 | 50.0 |

is specially notable in HR@10 for WN18RR: $59.0$ for $\mathrm{SPD}^{F_1}_{\mathrm{Sca}}$ vs $55.4$ and $56.6$ respectively. SPD reflections are very effective on WN18RR as well. They outperform their Euclidean and hyperbolic counterparts RefE and RefH, in particular when equipped with the Finsler metric. Rotations on the SPD manifold, on the other hand, seem to be less effective. However, Euclidean and hyperbolic rotations require $500$ dimensions whereas the $\mathrm{SPD}_{\mathrm{Rot}}$ models are trained on matrices of rank $14$ (equivalent to $105$ dims).

Regarding the choice of a distance metric, the Finsler One metric is better suited with respect to HR@3 and HR@10 when using scalings and reflections on WN18RR. For the FB15k-237 dataset, SPD models operating with the Riemannian metric outperform their Finsler counterparts. This suggests that the Riemannian metric is capable of disentangling the large number of relationships in this dataset to a better extent.

In these experiments we have evaluated models applying equivalent operations and scoring functions in different geometries, thus they can be thought as a vis-a-vis comparison of the metric spaces. We observe that SPD models tie or outperform baselines in most instances. This showcases the improved representation capacity of the SPD manifold when compared to Euclidean and hyperbolic spaces. Moreover, it demonstrates the effectiveness of the proposed metrics and operations in this manifold.

### 6.2 Knowledge Graph Recommender Systems

Recommender systems (RS) model user preferences to provide personalized recommendations [84]. KG embedding methods have been widely adopted into the recommendation problem as an effective tool to model side information and enhance the performance [82, 30]. For instance, one reason for recommending a movie to a particular user is that the user has already watched many movies from the same genre or director [47]. Given multiple relations between users, items, and heterogeneous entities, the goal is to predict the user's next item purchase or preference.

**Model:** We model the recommendation problem as a link prediction task over users and items [43]. In addition, we aim to incorporate side information between users, items and other entities. Hence we apply our KG embedding method from §6.1 as is, to embed this multi-relational graph. We evaluate the capabilities of the approach by only measuring the performance over user-item interactions.

**Datasets:** To investigate the recommendation problem endowed with added relationships, we employ the Amazon dataset [49, 52] (branches "Software", "Luxury & Beauty" and "Prime Pantry"), with users' purchases of products, and the MindReader dataset [14] of movie recommendations. Both datasets provide additional relationships between users, items and entities such as product brands, or movie directors and actors. To generate evaluation splits, the penultimate and last item the user has interacted with are withheld as dev and test sets respectively.

**Training:** In this setup we also augment the data by adding inverse relations and optimize the loss from Equation 9. We set the size of the matrices to $10 \times 10$ dimensions (equivalent to $55$ free parameters). More details about relationships and set of hyperparameters in Appendix B.2.

**Evaluation and metrics:** We follow the standard procedure of evaluating against $100$ randomly selected samples the user has not interacted with [33]. To evaluate the recommendation performance

Table 2: Results for Knowledge graph-based recommender systems.

| Model | SOFTWARE | | LUXURY | | PANTRY | | MINDREADER | |
|---|---|---|---|---|---|---|---|---|
| | MRR | H@10 | MRR | H@10 | MRR | H@10 | MRR | H@10 |
| TRANSE | 28.5±0.1 | 47.2±0.5 | 35.6±0.1 | 52.3±0.1 | 16.6±0.0 | 35.3±0.1 | 19.1±0.4 | 37.6±0.1 |
| ROTC | 28.5±0.3 | 45.4±1.4 | 33.0±0.1 | 49.8±0.2 | 14.5±0.0 | 31.3±0.2 | 25.3±0.3 | 50.3±0.6 |
| MURE | 29.4±0.4 | 47.1±0.4 | 35.6±0.7 | 54.0±0.3 | 19.4±0.1 | 39.5±0.2 | 25.2±0.3 | 49.9±0.6 |
| MURP | 29.6±0.3 | 47.9±0.3 | **37.5±0.1** | **55.2±0.3** | 19.4±0.1 | 39.8±0.2 | 25.3±0.3 | 49.3±0.2 |
| $\mathrm{SPD}^{R}_{\mathrm{Sca}}$ | 29.4±0.4 | 48.1±0.8 | **37.5±0.2** | 55.1±0.2 | 19.5±0.0 | 39.6±0.3 | 25.4±0.1 | 49.8±0.3 |
| $\mathrm{SPD}^{F_1}_{\mathrm{Sca}}$ | 28.8±0.1 | 46.9±0.5 | 37.3±0.3 | 54.1±0.9 | 19.0±0.1 | 38.8±0.2 | **25.7±0.5** | 49.5±0.1 |
| $\mathrm{SPD}^{R}_{\mathrm{Rot}}$ | **30.3±0.2** | 48.6±0.9 | 37.2±0.1 | 54.8±0.4 | **20.0±0.1** | **40.3±0.1** | 25.3±0.0 | **50.5±0.3** |
| $\mathrm{SPD}^{F_1}_{\mathrm{Rot}}$ | 30.1±0.1 | **49.1±0.3** | 36.9±0.1 | 54.5±0.6 | 19.2±0.0 | 39.3±0.1 | **25.7±0.0** | 49.5±0.2 |
| $\mathrm{SPD}^{R}_{\mathrm{Ref}}$ | 29.6±0.2 | 48.0±0.5 | 37.3±0.2 | 55.0±0.2 | 19.3±0.0 | 39.7±0.3 | 25.3±0.0 | 49.1±0.1 |
| $\mathrm{SPD}^{F_1}_{\mathrm{Ref}}$ | 29.3±0.1 | 47.5±0.6 | 36.8±0.0 | 54.8±0.1 | 18.6±0.2 | 38.3±0.3 | 24.8±0.2 | 47.9±1.8 |

we focus on the *buys / likes* relation. For each user $u$ we rank the items $i_j$ according to the scoring function $\phi(u, buys, i_j)$. We adopt MRR and H@10, as ranking metrics for recommendations.

**Baselines:** We compare to TransE [13], RotC [64], MuRE and MuRP [6] trained with 55 dimensions.

**Results:** In Table 2 we observe that the SPD models tie or outperform the baselines in both MRR and HR@10 across all analyzed datasets. Rotations in both, Riemannian and Finsler metrics, are more effective in this task, achieving the best performance in 3 out of 4 cases, followed by the scaling models. Overall, this shows the capabilities of the systems to effectively represent user-item interactions enriched with relations between items and their attributes, thus learning to better model users' preferences. Furthermore, it displays the versatility of the approach to diverse data domains.

### 6.3 Question Answering

We evaluate our approach on the task of Question Answering (QA). In this manner we also showcase the capabilities of our methods to train word embeddings.

**Model:** We adapt the model from HyperQA [65] to SPD. We model word embeddings $t_i \in \mathrm{SPD}_n$, and represent question/answers as a summation of the embeddings of their corresponding tokens. We apply a feature transformation $T(\cdot)$ followed by a bias addition, as an equivalent of a neural linear layer. $T(\cdot)$ can be a scaling, rotation or reflection. Finally we compute a distance-based similarity function between the resulting question/answer representations as defined in Equation 10, where $w_f, w_b \in \mathbb{R}$, $B \in \mathrm{SPD}_n$ and the transformation $T$ are parameters of the model.

$$\begin{aligned} \mathrm{sim}(q,a) &= -w_f d(\mathbf{Q},\mathbf{A}) + w_b, \\ \text{where } \mathbf{Q} &= T(\bigoplus_{i=1}^{n} t_i^q) \oplus B \end{aligned} \tag{10}$$

**Datasets:** We analyze two popular benchmarks for QA: TrecQA [76] (clean version) and WikiQA [80], filtering out questions with multiple answers from the dev and test sets.

**Training:** We optimize the cross-entropy loss from Eq. 9, where we replace $\phi$ for $\mathrm{sim}(q,a)$ and for each question we use wrong answers as negative samples. We set the size of the matrices to $14 \times 14$ dimensions (equivalent to 105 free parameters). The set of hyperparameters can be found in Appendix B.3.

**Evaluation metrics:** At test time, for each question we rank its candidate answers according to Eq. 10. We adopt MRR and H@1 as evaluation metrics.

Table 3: Results for Question Answering.

| Model | TRECQA | | WIKIQA | |
|---|---|---|---|---|
| | MRR | H@1 | MRR | H@1 |
| Euclidean | 55.9±2.0 | 41.0±2.0 | 43.4±0.3 | 22.4±1.1 |
| Hyperbolic | 58.0±1.3 | 39.3±2.0 | 44.0±0.4 | 22.8±0.6 |
| $\mathrm{SPD}^{R}_{\mathrm{Sca}}$ | 55.4±0.1 | 37.1±0.1 | **45.5±0.5** | 24.4±1.1 |
| $\mathrm{SPD}^{F_1}_{\mathrm{Sca}}$ | 57.1±0.7 | 38.6±0.2 | 44.8±0.5 | 24.0±0.6 |
| $\mathrm{SPD}^{R}_{\mathrm{Rot}}$ | 58.7±1.5 | 41.4±2.9 | 44.6±0.6 | 23.6±0.6 |
| $\mathrm{SPD}^{F_1}_{\mathrm{Rot}}$ | 58.1±0.5 | **43.6±1.0** | 43.7±0.4 | 23.8±0.8 |
| $\mathrm{SPD}^{R}_{\mathrm{Ref}}$ | 57.3±0.3 | 40.7±1.1 | 43.9±0.7 | 23.4±2.0 |
| $\mathrm{SPD}^{F_1}_{\mathrm{Ref}}$ | **59.6±0.5** | 42.1±1.0 | 44.7±1.2 | **25.0±2.5** |

**Baselines:** We compare against Euclidean and hyperbolic spaces of 105 dimensions. For the Euclidean model we employ a linear layer as feature transformation. For the hyperbolic model, we operate on the tangent space and project the points into the Poincaré ball to compute distances.

**Results:** We present results in Table 3. In both datasets, we see that the word embeddings and transformations learned by the SPD models are able to place questions and answers representations in the space such that they outperform Euclidean and hyperbolic baselines. Finsler metrics seem to be

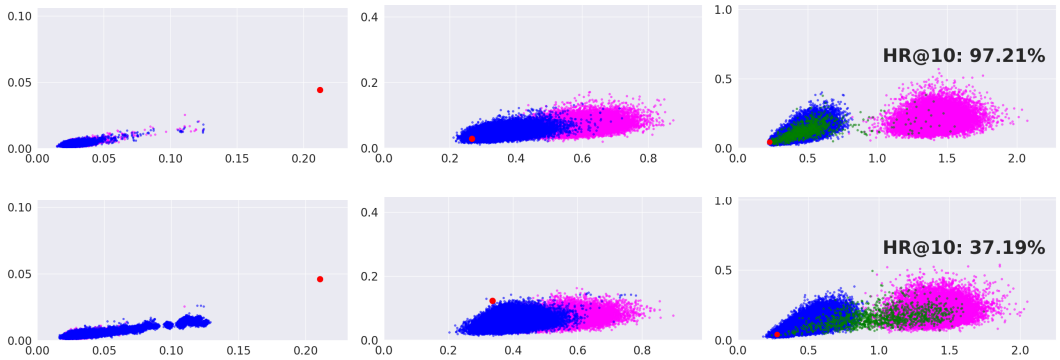


Figure 5: Train, negative and validation triples for relationships *'derivationally related form'* (top) and *'hypernym'* (bottom) for 5 (left), 50 (center) and 3000 (right) epochs for the $\mathrm{SPD}^{F_1}_{\mathrm{Sca}}$ model. The red dot corresponds to the relation addition $\mathbf{R}$.

very effective in this scenario, improving the performance of their Riemannian counterparts in many cases. Overall, this suggests that embeddings in SPD manifolds learn meaningful representations that can be exploited into downstream tasks. Moreover, we showcase how to employ different operations as feature transformations and bias additions, replicating the behavior of linear layers in classical deep learning architectures that can be seamlessly integrated with different distance metrics.

### 6.4 Analysis

One reason to embed data into Riemannian manifolds, such as SPD, is to use geometric properties of the manifold to analyze the structure of the data. Visualizations in SPD are difficult due to their high dimensionality. As a solution we use the vector-valued distance function to develop a new tool to visualize and analyze structural properties of the learned representations.

We adopt the vector $(n-1, n-3, \cdots, -n+3, -n+1)$, as the barycenter of the space in $\mathbb{R}^n$ where the VVD is contained. Then, we plot the norm of the VVD vector and its angle with respect to this barycenter. In Figure 5, we compute and plot the VVD corresponding to $d(\mathbf{M}_r \otimes \mathbf{H}, \mathbf{T})$ and $\mathbf{R}$ as defined in Eq. 7 for KG models trained on WN18RR. In early stages of the training, all points fall near the origin (left side of the plots). As training evolves, the model learns to separate true $(h, r, t)$ triples from corrupted ones (center part). When the training converges, the model is able to clearly disentangle and cluster positive and negative samples. We observe how the position of the validation triples (green points, not seen during training) directly correlates with the performance of each relation. Plots for more relations in Appendix B.1.

## 7 Conclusions

Riemannian geometry has gained attention due to its capacity to represent non-Euclidean data arising in several domains. In this work we introduce the vector-valued distance function, which allows to implement universal models (generalizing previous metrics on SPD), and can be leveraged to provide a geometric interpretation on what the models learn. Moreover, we bridge the gap between Euclidean and SPD geometry under the lens of the gyrovector theory, providing means to transfer standard arithmetic operations from the Euclidean setting to their analog notions in SPD. These tools enable practitioners to exploit the full representation power of SPD, and profit from the enhanced expressivity of this manifold. We propose and evaluate SPD models on three tasks and eight datasets, which showcases the versatility of the approach and ease of integration with downstream tasks. The results reflect the superior expressivity of SPD when compared to Euclidean or hyperbolic baselines.

This work is not without limitations. We consider the computational complexity of working with spaces of matrices to be the main drawback. In Appendix A.1 we detail the complexity of different operations. Although a matrix of rank $n$ implies $n(n+1)/2$ dimensions thus a large $n$ value is usually not required, the cost of many operations is polynomial instead of linear.

## References


[1] Ai, Q., Azizi, V., Chen, X., and Zhang, Y. (2018). Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms*, 11(9).

[2] Allen, C., Balazevic, I., and Hospedales, T. (2021). Interpreting knowledge graph relation representation from word embeddings. In *International Conference on Learning Representations*.

[3] Arsigny, V., Fillard, P., Pennec, X., and Ayache, N. (2006a). Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM J. Matrix Anal. Appl.*, 29(1):328–347.

[4] Arsigny, V., Fillard, P., Pennec, X., and Ayache, N. (2006b). Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine*, 56(2):411–421.

[5] Bachmann, G., Becigneul, G., and Ganea, O.-E. (2020). Constant curvature graph convolutional networks. In *37th International Conference on Machine Learning (ICML)*.

[6] Balazevic, I., Allen, C., and Hospedales, T. (2019a). Multi-relational poincaré graph embeddings. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 4463–4473. Curran Associates, Inc.

[7] Balazevic, I., Allen, C., and Hospedales, T. (2019b). TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5185–5194, Hong Kong, China. Association for Computational Linguistics.

[8] Ballmann, W., Brin, M., and Eberlein, P. (1985). Structure of manifolds of nonpositive curvature. I. *Annals of Mathematics*, 122(1):171–203.

[9] Bécigneul, G. and Ganea, O. (2019). Riemannian adaptive optimization methods. In *7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA.

[10] Bhatia, R., Jain, T., and Lim, Y. (2019). On the Bures–Wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191.

[11] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

[12] Bonnabel, S. (2011). Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58.

[13] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 26, pages 2787–2795. Curran Associates, Inc.

[14] Brams, A. H., Jakobsen, A. L., Jendal, T. E., Lissandrini, M., Dolog, P., and Hose, K. (2020). Mindreader: Recommendation over knowledge graph entities with explicit user ratings. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2975–2982, New York, NY, USA. Association for Computing Machinery.

[15] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

[16] Brooks, D., Schwander, O., Barbaresco, F., Schneider, J.-Y., and Cord, M. (2019a). Riemannian batch normalization for SPD neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 15489–15500. Curran Associates, Inc.

[17] Brooks, D. A., Schwander, O., Barbaresco, F., Schneider, J.-Y., and Cord, M. (2019b). Exploring complex time-series representations for riemannian machine learning of radar data. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3672–3676.

[18] Carreira, J., Caseiro, R., Batista, J., and Sminchisescu, C. (2012). Semantic segmentation with second-order pooling. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision – ECCV 2012*, pages 430–443, Berlin, Heidelberg. Springer Berlin Heidelberg.

[19] Chakraborty, R., Bouza, J., Manton, J., and Vemuri, B. C. (2020). Manifoldnet: A deep neural network for manifold-valued data with applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

[20] Chakraborty, R., Yang, C.-H., Zhen, X., Banerjee, M., Archer, D., Vaillancourt, D., Singh, V., and Vemuri, B. (2018). A statistical recurrent model on the manifold of symmetric positive definite matrices. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

[21] Chami, I., Wolf, A., Juan, D.-C., Sala, F., Ravi, S., and Ré, C. (2020). Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914, Online. Association for Computational Linguistics.

[22] Chami, I., Ying, Z., Ré, C., and Leskovec, J. (2019). Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems 32*, pages 4869–4880. Curran Associates, Inc.

[23] Cruceru, C., Becigneul, G., and Ganea, O.-E. (2020). Computationally tractable Riemannian manifolds for graph embeddings. In *37th International Conference on Machine Learning (ICML)*.

[24] Dettmers, T., Pasquale, M., Pontus, S., and Riedel, S. (2018). Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818.

[25] Dong, Z., Jia, S., Zhang, C., Pei, M., and Wu, Y. (2017). Deep manifold learning of symmetric positive definite matrices with application to face recognition. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4009–4015. AAAI Press.

[26] Donoho, D. L. and Tsaig, Y. (2008). Fast solution of $\ell_1$-norm minimization problems when the solution may be sparse. *IEEE Trans. Information Theory*, 54(11):4789–4812.

[27] Feragen, A., Lauze, F., and Hauberg, S. (2015). Geodesic exponential kernels: When curvature and linearity conflict. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3032–3042.

[28] Ganea, O., Becigneul, G., and Hofmann, T. (2018). Hyperbolic neural networks. In *Advances in Neural Information Processing Systems 31*, pages 5345–5355. Curran Associates, Inc.

[29] Gao, Z., Wu, Y., Bu, X., Yu, T., Yuan, J., and Jia, Y. (2019). Learning a robust representation via a deep network on symmetric positive definite manifolds. *Pattern Recognition*, 92:1–12.

[30] Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2020). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

[31] Ha Quang, M., San Biagio, M., and Murino, V. (2014). Log-Hilbert-Schmidt metric between positive definite operators on Hilbert spaces. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

[32] Harandi, M. T., Salzmann, M., and Hartley, R. (2014). From manifold to manifold: Geometry-aware dimensionality reduction for SPD matrices. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, pages 17–32, Cham. Springer International Publishing.

[33] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T.-S. (2017). Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 173–182, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

[34] Helgason, S. (1978). *Differential geometry, Lie groups, and symmetric spaces*. Academic Press New York.

[35] Huang, Z. and Gool, L. V. (2017). A Riemannian network for SPD matrix learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 2036–2042. AAAI Press.

[36] Huang, Z., Wang, R., Shan, S., and Chen, X. (2014). Learning Euclidean-to-Riemannian metric for point-to-set classification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1677–1684.

[37] Huang, Z., Wang, R., Shan, S., Li, X., and Chen, X. (2015). Log-Euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 720–729. JMLR.org.

[38] Ionescu, C., Vantzos, O., and Sminchisescu, C. (2015). Matrix backpropagation for deep networks with structured layers. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2965–2973.

[39] Jayasumana, S., Hartley, R., Salzmann, M., Li, H., and Harandi, M. (2013). Kernel methods on the Riemannian manifold of symmetric positive definite matrices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[40] Kapovich, M., Leeb, B., and Millson, J. (2009). Convex functions on symmetric spaces, side lengths of polygons and the stability inequalities for weighted configurations at infinity. *J. Differential Geom.*

[41] Kapovich, M., Leeb, B., and Porti, J. (2017). Anosov subgroups: dynamical and geometric characterizations. *European Journal of Mathematics*, 3(3):808–898.

[42] Lacroix, T., Usunier, N., and Obozinski, G. (2018). Canonical tensor decomposition for knowledge base completion. In Dy, J. and Krause, A., editors, *Proceedings of Machine Learning Research*, volume 80 of *Proceedings of Machine Learning Research*, pages 2863–2872, Stockholmsmässan, Stockholm Sweden. PMLR.

[43] Li, J., Zhang, L., Meng, F., and Li, F. (2014). Recommendation algorithm based on link prediction and domain knowledge in retail transactions. *Procedia Computer Science*, 31:875 – 881. 2nd International Conference on Information Technology and Quantitative Management, ITQM 2014.

[44] Li, P., Xie, J., Wang, Q., and Gao, Z. (2018). Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 947–955.

[45] Li, Y. and Lu, R. (2018). Locality preserving projection on SPD matrix lie group: algorithm and analysis. *Sci. China Inf. Sci.*, 61(9):092104:1–092104:15.

[46] Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

[47] Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., and Ren, X. (2019). Jointly learning explainable rules for recommendation with knowledge graph. In *The World Wide Web Conference*, WWW '19, page 1210–1221, New York, NY, USA. Association for Computing Machinery.

[48] Mao, Y., Wang, R., Shan, S., and Chen, X. (2019). Cosonet: Compact second-order network for video face recognition. In Jawahar, C. V., Li, H., Mori, G., and Schindler, K., editors, *Computer Vision – ACCV 2018*, pages 51–67, Cham. Springer International Publishing.

[49] McAuley, J. and Leskovec, J. (2013). Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, page 165–172, New York, NY, USA. Association for Computing Machinery.

[50] Miller, G. A. (1992). WordNet: A lexical database for English. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

[51] Nguyen, X. S., Brun, L., Lezoray, O., and Bougleux, S. (2019). A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[52] Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics.

[53] Nickel, M. and Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 6341–6350. Curran Associates, Inc.

[54] Nielsen, F. and Sun, K. (2019). *Clustering in Hilbert's Projective Geometry: The Case Studies of the Probability Simplex and the Elliptope of Correlation Matrices*, pages 297–331. Springer International Publishing, Cham.

[55] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

[56] Pennec, X., Fillard, P., and Ayache, N. (2006). A Riemannian framework for tensor computing. *Int. J. Comput. Vision*, 66(1):41–66.

[57] Planche, P. (1995). *Géométrie de Finsler sur les espaces symétriques*. PhD thesis, Université de Genève, Geneve, Switzerland.

[58] Ratliff, N. D., Wyk, K. V., Xie, M., Li, A., and Rana, M. A. (2020). Generalized nonlinear and Finsler geometry for robotics. *CoRR*, abs/2010.14745.

[59] Said, S., Bombrun, L., Berthoumieu, Y., and Manton, J. H. (2017). Riemannian Gaussian distributions on the space of symmetric positive definite matrices. *IEEE Transactions on Information Theory*, 63(4):2153–2170.

[60] Shen, Z. (2006). Riemann-Finsler geometry with applications to information geometry. *Chinese Annals of Mathematics, Series B*, 27:73–94.

[61] Shimizu, R., Mukuta, Y., and Harada, T. (2021). Hyperbolic neural networks++. In *International Conference on Learning Representations*.

[62] Sra, S. (2012). A new metric on the manifold of kernel matrices with application to matrix geometric means. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[63] Sra, S. (2015). Positive definite matrices and the S-divergence. *Proceedings of the American Mathematical Society*. Published electronically: October 22, 2015.

[64] Sun, Z., Deng, Z.-H., Nie, J.-Y., and Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

[65] Tay, Y., Tuan, L. A., and Hui, S. C. (2018). Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 583–591, New York, NY, USA. ACM.

[66] Tosato, D., Farenzena, M., Spera, M., Murino, V., and Cristani, M. (2010). Multi-class classification on Riemannian manifolds for video surveillance. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II*, volume 6312 of *Lecture Notes in Computer Science*, pages 378–391. Springer.

[67] Toutanova, K. and Chen, D. (2015). Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

[68] Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., and Bouchard, G. (2016). Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080. JMLR.org.

[69] Tuzel, O., Porikli, F., and Meer, P. (2006). Region covariance: A fast descriptor for detection and classification. In Leonardis, A., Bischof, H., and Pinz, A., editors, *Computer Vision – ECCV 2006*, pages 589–600, Berlin, Heidelberg. Springer Berlin Heidelberg.

[70] Tuzel, O., Porikli, F., and Meer, P. (2008). Pedestrian detection via classification on Riemannian manifolds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1713–1727.

[71] Ungar, A. (2005). Gyrovector spaces and their differential geometry. *Nonlinear Functional Analysis and Applications*, 10.

[72] Ungar, A. A. (2008). *A Gyrovector Space Approach to Hyperbolic Geometry*. Morgan & Claypool.

[73] Ungar, A. A. (2018). *Beyond Pseudo-Rotations in Pseudo-Euclidean Spaces*. Mathematical Analysis and its Applications. Academic Press.

[74] Vemulapalli, R. and Jacobs, D. (2015). Riemannian metric learning for symmetric positive definite matrices. *ArXiv*, abs/1501.02393.

[75] Vermeer, J. (2005). A geometric interpretation of Ungar's addition and of gyration in the hyperbolic plane. *Topology and Its Applications: a journal devoted to general, geometric, set-theoretic and algebraic topology*, 152(3):226–242.

[76] Wang, M., Smith, N. A., and Mitamura, T. (2007). What is the Jeopardy model? A quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic. Association for Computational Linguistics.

[77] Wang, W., Wang, R., Huang, Z., Shan, S., and Chen, X. (2018). Discriminant analysis on Riemannian manifold of Gaussian distributions for face recognition with image sets. *IEEE Transactions on Image Processing*, 27(1):151–163.

[78] Wu, Y., Jia, Y., Li, P., Zhang, J., and Yuan, J. (2015). Manifold kernel sparse representation of symmetric positive-definite matrices and its applications. *IEEE Transactions on Image Processing*, 24(11):3729–3741.

[79] Yang, T., Sha, L., and Hong, P. (2020). *NagE: Non-Abelian Group Embedding for Knowledge Graphs*, page 1735–1742. Association for Computing Machinery, New York, NY, USA.

[80] Yang, Y., Yih, W.-t., and Meek, C. (2015). WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

[81] Yin, M., Guo, Y., Gao, J., He, Z., and Xie, S. (2016). Kernel sparse subspace clustering on symmetric positive definite manifolds. In *CVPR*, pages 5157–5164. IEEE Computer Society.

[82] Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 353–362, New York, NY, USA. Association for Computing Machinery.

[83] Zhang, S., Tay, Y., Yao, L., and Liu, Q. (2019a). Quaternion knowledge graph embeddings. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32, pages 2735–2745. Curran Associates, Inc.

[84] Zhang, S., Yao, L., Sun, A., and Tay, Y. (2019b). Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1).

[85] Zhang, T., Zheng, W., Cui, Z., and Li, C. (2018a). Deep manifold-to-manifold transforming network. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 4098–4102.

[86] Zhang, Y., Ai, Q., Chen, X., and Wang, P. (2018b). Learning over knowledge-base embeddings for recommendation. *CoRR*, abs/1803.06540.

[87] Zhao, K., Wiliem, A., Chen, S., and Lovell, B. C. (2019). Convex class model on symmetric positive definite manifolds. *Image and Vision Computing*, 87:57–67.

# Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] In the conclusions (§7) and in Appendix A.1 we highlight the computational complexity as the main limitation.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The authors consider that the main contributions of this work are Representation Learning tools, and they do not pose societal risks by themselves.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Assumptions and proofs can be found on Appendix C and D.
   (b) Did you include complete proofs of all theoretical results? [Yes] Assumptions and proofs can be found on Appendix C and D.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Datasets are well-known and publicly accessible. For experiment reproducibility see §6, and Appendix B. Code will be released upon publication.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See previous answer.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] For KG recommender systems and QA experiments, we report mean and standard deviation of the three best runs for all models. For Knowledge graph completion we do not report standard deviation, given that none of the previous work for this task has reported it either, hence a comparison of these values is not possible.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix B
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Corresponding citations are made to the creators of PyTorch and works that introduced the different datasets.
   (b) Did you mention the license of the assets? [No]

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Code will be released upon publication.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] Datasets and baselines models used in this work are publicly available. We cite the corresponding work where they were published for the first time.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]