Learning distributions on Riemannian manifolds

Autoencoder for SPD matrices

Charlotte BOUCHERIE, Florian YGER, Thibault DE SURREL

LITIS

2025

Table of contents

Table of contents

Use of SPD matrices

- Covariances matrices
- Used in computer vision, brain imaging, brain-computer interface (EEG)

A Riemannian Network for SPD Matrix Learning

Introduction of a network architecture that preserves the properties of positive definite matrices for Deep Learning [6]

- 3 different layers: BiMap, ReEig, LogEig
- We will base ourselves on these layers for our autoencoder

Different works

DreamNet: A Deep Riemannian Manifold Network for SPD Matrix Learning [10]

- Methodology for creating deep networks
- Stacked Riemannian Autoencoder (SRAE) at the end of the network

Riemannian Multinomial Logistics Regression for SPD Neural Networks [3]

- Adapting logistic regression for SPD matrices
- New specific layer for classification
- Use of Log-Euclidean Metric or Log-Cholesky Metric

SPD domain-specific batch normalization to crack interpretable unsupervised domain adaptation in EEG [7]

- Specific batch normalization for SPD matrices

Different works

Riemannian batch normalization for SPD neural networks [1]

- Specific batch normalization for SPD matrices

A Riemannian Residual Learning Mechanism for SPD Network [2]

- Improves learning process for SPD networks

U-SPDNet: An SPD manifold learning-based neural network for visual classification [12]

- SPD matrices from visual data

Different works

Reducing the Dimensionality of SPD Matrices with Neural Networks in BCI [8]

- Simplification of complex data for a better interpretability and processing in BCI data

Schur's Positive Definite Network: Deep Learning in the SPD Cone With Structure [9]

- Shows that the use of the structure in the network improves the performances

Modeling Graphs Beyond Hyperbolic: Graph Neural Networks in Symmetric Positive Definite Matrices [13]

- Applies GNN to SPD matrices
- To model graph structures in SPD matrices

Different works

SymNet: A Simple Symmetric Positive Definite Manifold Deep Learning Method for Image Set Classification [11]

- Image set classification

From Manifold to Manifold: Geometry-Aware Dimensionality Reduction for SPD Matrices [4]

- Lower-dimensional and more discriminative SPD matrices from SPD matrices with orthonormal projection

Geometry-Aware Principal Component Analysis for Symmetric Positive Definite Matrices [5]

- PCA applied to SPD matrices
- Preserves more data variance
- Extends PCA from Euclidean to Riemannian geometries

Existing solutions in Euclidean geometry

- Flatten to tangent space
- Distance approximation: Frobenius distance
  $L = ||A - B||_F = \sqrt{\sum_{i,j}(A_{ij} - B_{ij})^2}$

Problems in Euclidean geometry

- Does not preserve the curvature of space
- Non-optimal results with euclidean distance
- Swelling effect : the determinants of the interpolation of flattened matrices

Riemannian geometry

Riemann metric *(AIRM)*: $\delta_R^2(X, Y) = ||\log(X^{-1/2} Y X^{-1/2})||_F^2$

- Measure the similarity between two SPD matrices while respecting the structure
- We will use it in our AE in the model, in the cost function and in the trustworthiness.

Representing information with SPD matrices has proven beneficial for many recognition tasks. Considering Riemannian geometry comes at a high cost especially in high-dimensional ones that limits applicability.

Objectives

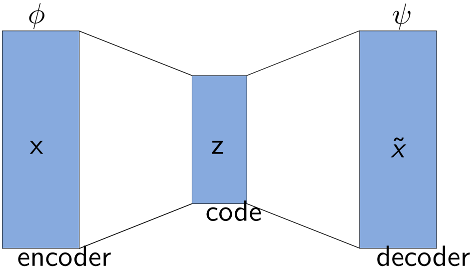
- Autoencoder for SPD matrices for dimension reduction
- Layer to do the reverse operations of the autoencoder
- Comparison of different models
- Impact of distance for reconstruction error

Table of contents

Autoencoder Basics

- Unsupervised learning: measurement of reconstruction error
- Dimension reduction
- Learn the underlying patterns
- Used for generative models

$$\phi : \mathcal{X} \to \mathcal{F} \text{ , encoder}$$
$$\psi : \mathcal{F} \to \mathcal{X} \text{ , decoder}$$
$$\phi, \psi = \underset{\phi,\psi}{\arg\min} ||X - (\psi \circ \phi)X||^2$$



Reconstruction error

- For each matrix, we calculate the Riemannian distance with its reconstruction.

$$\phi : \mathcal{X} \rightarrow \mathcal{F}$$
$$\psi : \mathcal{F} \rightarrow \mathcal{X}$$
$$\phi, \psi = \arg\min_{\phi,\psi} \delta_R^2(X, \psi(\phi(X))) = \arg\min_{\phi,\psi} ||\log(X^{-1/2}\psi(\phi(X))X^{-1/2})||_F^2$$

Trustworthiness

- For each matrix, we take its k closest matrices in the output space and its closest matrices in the input space.
- The distance is the same used to calculate our cost function.
- We penalize proportionally to the difference in ranks in the input space.
- We do not penalize matrices coming closer together.

$$T(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i^k} \max(0, (r(i,j) - k))$$

Accuracy

- We use MDM (Minimum Distance to Mean) to know the precision before reconstituting our matrices.
- For each class, a centroid is estimated according to our distance.
- We compare the initial precision with the final precision.

BiMap layer

- The function of this layer is to generate more compact and more discriminative SPD matrices.
- Layer which performs a bilinear map f_b to transform the initial matrices into new matrices of lower dimension.

$$X_k = f_b^{(k)}(X_{k-1}; W_k) = W_k X_{k-1} W_k^T$$

W_k is of full rank to guarantee that X_k remains SPD.

Network parameters

Number of input filters/channels *hi*, number of output filters/channels *ho*, size of input matrix *ni*, size of output matrix *no*

ReEig layer

- The function of this layer is to improve discriminative performance by introducing nonlinearity, in the same way as ReLU.
- Introduction of a non-linear function f_r which corrects the matrices by setting a threshold for low eigenvalues.

$$X_k = f_r^{(k)}(X_{k-1}) = U_{k-1} \max(\epsilon I, \Sigma_{k-1}) U_{k-1}^T$$

LogEig/ExpEig layers

LogEig

The function of this layer is to be able to apply Riemann geometry to the output matrix.

$$X_k = f_l^{(k)}(X_{k-1}) = \log(X_{k-1}) = U_{k-1} \log(\Sigma_{k-1}) U_{k-1}^T$$

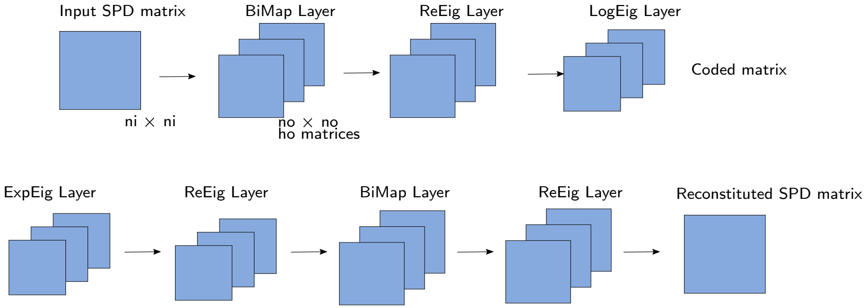
ExpEig

The function of this layer is to apply the inverse function of the LogEig layer.

$$X_k = f_e^{(k)}(X_{k-1}) = \exp(X_{k-1})$$

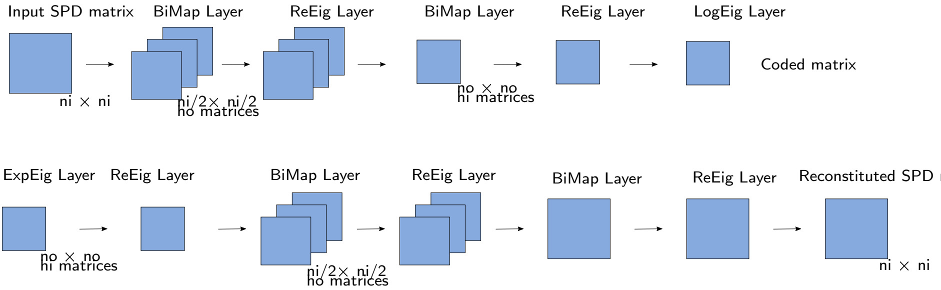
One layer

- Single BiMap layer for the encoder from $ni \to no$ and $hi \to ho$.
- We look at the influence of the output dimension and the output layer.
- The decoder does the opposite operation.



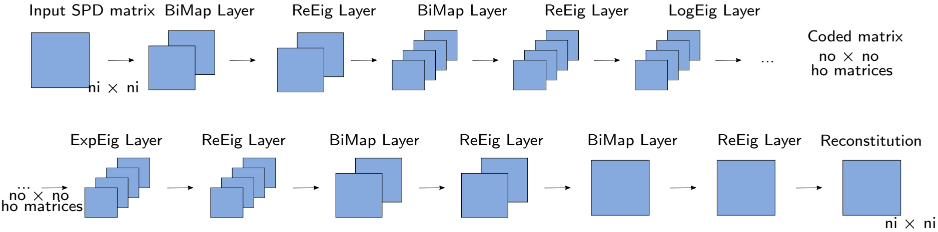
Two layers with funnel channels

- Two BiMap layers.
  - $ni \to ni/2$ and $hi \to ho$.
  - $ni/2 \to no$ and $ho \to hi$.
- We look at the influence of the number of intermediate channels and the output dimension.
- The decoder does the opposite operation.



Multiple layers evenly distributed

- Number of BiMap layers set in parameters.
- Channels and intermediate matrix sizes based on the number of layers.
- The decoder does the opposite operation.



Multiple layers halved in dimension

- Number of BiMap layers and filters in layers depends on *ni* and *no*.
- Matrix size divided by two, number of filters multiplied by two at each layer.
- The decoder does the opposite operation.

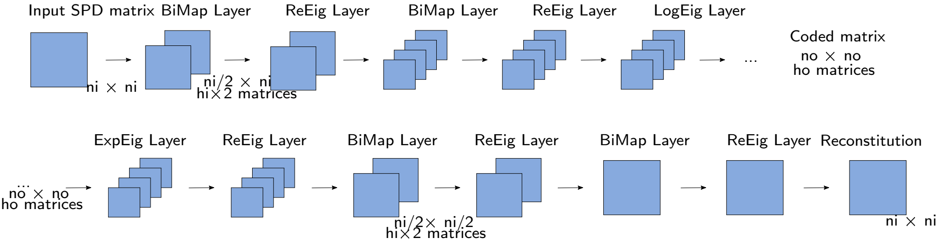


Table of contents

Synthetic data

Adding noises

Gaussian noise

Salt and pepper noise

Masking noise

BCI data

Results

- Prediction less accurate
- The more we preserve the neighborhood, the worse the accuracy becomes.
- Lossy compression

Accuracy is already good without the autoencoding. We should try with a dataset

Daniel Brooks, Olivier Schwander, Frederic Barbaresco, Jean-Yves Schneider, and Matthieu Cord.
Riemannian batch normalization for spd neural networks, 2019.

Zhenyu Cai, Rui Wang, Tianyang Xu, Xiaojun Wu, and Josef Kittler.
A riemannian residual learning mechanism for spd network.
In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, 2024.

Ziheng Chen, Yue Song, Gaowen Liu, Ramana Rao Kompella, Xiaojun Wu, and Nicu Sebe.
Riemannian multinomial logistics regression for spd neural networks, 2024.

Mehrtash T. Harandi, Mathieu Salzmann, and Richard Hartley.
From manifold to manifold: Geometry-aware dimensionality reduction for spd matrices, 2014.

References II

Inbal Horev, Florian Yger, and Masashi Sugiyama.
Geometry-aware principal component analysis for symmetric positive definite matrices.
Machine Learning, 106, 04 2017.

Zhiwu Huang and Luc Van Gool.
A riemannian network for SPD matrix learning.
CoRR, abs/1608.04233, 2016.

Reinmar J Kobler, Jun ichiro Hirayama, Qibin Zhao, and Motoaki Kawanabe.
Spd domain-specific batch normalization to crack interpretable unsupervised domain adaptation in eeg, 2022.

References III

Zhen Peng, Hongyi Li, Di Zhao, and Chengwei Pan.
Reducing the dimensionality of spd matrices with neural networks in bci.
Mathematics, 11:1570, 03 2023.

Can Pouliquen, Mathurin Massias, and Titouan Vayer.
Schur's Positive-Definite Network: Deep Learning in the SPD cone with structure.
working paper or preprint, November 2024.

Rui Wang, Xiao-Jun Wu, Ziheng Chen, Tianyang Xu, and Josef Kittler.
Dreamnet: A deep riemannian network based on spd manifold learning for visual classification, 2022.

References IV

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, 2022.

Rui Wang, Xiao-Jun Wu, Tianyang Xu, Cong Hu, and Josef Kittler.
U-spdnet: An spd manifold learning-based neural network for visual classification.
Neural Networks, 161:382–396, 2023.

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, 2023.