Locally Preserving Projection on Symmetric Positive Definite Matrix Lie Group

Yangyang Li*†
*Academy of Mathematics and Systems Science Key Lab of MADIS CAS, Beijing
† University of Chinese Academy of Sciences, Beijing, China
Email: liyangyang12@mails.ucas.ac.cn

***Abstract*—Symmetric Positive Definite (SPD) matrices have been widely used as feature descriptors in image recognition. However, the dimension of an SPD matrix built by image feature descriptors is usually high. So SPD matrices oriented dimensionality reduction techniques are needed. The existing manifold learning algorithms just reduce the dimension of high dimensional vector-form data. Our work is based on the fact that the set of all SPD matrices is known to have a Lie group structure. Our method aims to transform the manifold learning algorithm to SPD matrix Lie group. We first construct the corresponding Laplacian matrix on SPD matrix Lie group, then make use of the basic idea of manifold learning algorithm LPP (locality preserving projection). Thus we call our approach Lie-LPP to emphasize its Lie group character. We also do some experiments on human face recognition and achieve effective results.**

***Index Terms*—SPD matrix Lie group, Laplacian matrix, LPP, Dimension reduction.**

1. Introduction

Image recognition is a very popular research project in the field of machine vision and pattern recognition. The original feature descriptor of an image is the corresponding pixel matrix. In general, the type of image feature descriptor highly influences the recognition rate during image recognition. For original image pixel matrix, it is very difficult to directly do the recognition on the matrix space. It is usual to transform a pixel matrix to a row feature vector. In practice the dimension of the feature vector is especially high. In addition, most of the features in this vector are not effective for the image recognition. In order to diminish the negative effect of dimension curse, a large number of dimensionality reduction algorithms have been invented to implement image recognition [1]. But the vector-form descriptor breaks the geometric structure of pixel matrix space and loses the spatial information of the pixel matrix space. To avoid these disadvantages, F. Porikli [6] presented a new form of feature descriptor via computing a feature covariance matrix within any size region in an image, which preserves the local geometric structure of the image pixel matrix. The method of covariance matrix descriptor has mainly been used in image recognition [4, 5, 7].

Non-singular covariance matrix mentioned in this paper is an SPD matrix, and the set of all SPD matrices with the same size forms a Riemannian manifold. Generally the dimension of SPD Riemannian manifold is very high. Harandi et al. [11] proposed to learn a kernel function to map the SPD matrix into a higher dimensional Euclidean space and then use traditional manifold learning algorithm LPP to reduce its dimension. But this method would run hard if the original dimension of SPD Riemannian manifold is especially high. In addition, the geometric and algebraic structure of SPD manifold would be broken during the dimension reduction process. To overcome this limitation, Harandi et al. [12] suggested to map the original SPD manifold to a lower dimensional SPD manifold which preserves the geometric and algebraic structure of SPD manifold. But this method needed to get help of Grassmann manifold and the time cost is especially high. In order to reduce the time cost, Huang et al. [3] proposed transforming SPD matrices to the tangent space by logarithm mapping and learning an optimal dimension reduction matrix on that tangent space. However, this algorithm needs several parameters which are sensitive factors influencing the algorithm. In addition, this method only preserved the local structure of manifold and did not analysis the global structure.

SPD matrices with the same dimension lie on a Riemannian manifold. What's more, they form a Lie group called SPD matrix Lie group S_+^D. All the abovementioned methods can not reduce the dimension of SPD matrix Lie group directly. In this paper, we propose to set a new method to avoid these disadvantages.

Recent years manifold learning [2, 8, 9, 10] has become a significant branch of machine learning and has been widely used in computer vision and pattern recognition. The main purpose of manifold learning is to reduce the dimension of high dimensional vector-form data by uncover the intrinsic geometric structure of high dimensional data points. For example, LPP [2] and LEP [10] aimed to construct the corresponding graph Laplacian matrix which reflects the local geometric structure of data points. Since a covariance matrix is represented by the product of feature vectors and is bilinear, there is an intuitive idea that we can solve the dimensionality reduction problem directly on the SPD matrix Lie group. We extended the idea of LPP to a dimensionality reduction problem on SPD matrix Lie group and attempted to construct the corresponding Laplacian matrix on SPD

matrices dataset which reflects the intrinsic structure of SPD matrix Lie group. We developed the Lie-LPP algorithm to get a dimension reduction algorithm on SPD matrix Lie group.

2. Lie-LPP on SPD Matrix Lie Group

In this section we first analyze the geometric and algebra structure of SPD matrix Lie group. Then we give the construction of Laplacian matrix on SPD matrix Lie group. Finally we describe our proposed dimensionality reduction algorithm Lie-LPP.

2.1. Geometric Structure of SPD Matrix Lie Group

Distance metric is required to measure the distance between two different images. In this paper we choose Log-Euclidean metric from [15] as the Riemannian metric of SPD matrix Lie group. The SPD matrix Lie group that we consider in this paper is represented by $\mathcal{S}_+^D$, where the size of every point $S_1 \in \mathcal{S}_+^D$ is $D \times D$. The tangent space of $\mathcal{S}_+^D$ at the identity is *Sym(D)*, the space of symmetric matrices.

Definition 2.1. (Log-Euclidean Metric) [15] *The Riemannian metric at a point $S_1 \in \mathcal{S}_+^D$ is a scalar product defined in the tangent space $T_{S_1}\mathcal{S}_+^D$:*

$$\langle T_1, T_2 \rangle = \langle Dlog.T_1, Dlog.T_2 \rangle, \tag{1}$$

where $T_1, T_2 \in T_{S_1}\mathcal{S}_+^D$.

Log-Euclidean metric is a bi-invariant metric defined on SPD matrix Lie group because the corresponding metric on the tangent space is Euclidean metric. So endowed with Log-Euclidean metric, $\mathcal{S}_+^D$ is a flat manifold and locally isometric to the tangent space *Sym(D)*. The corresponding exponential and logarithmic maps based on this metric are shown as [3]:

$$exp_{S_1}(T_1) = exp\left(log(S_1) + D_{S_1}log \cdot T_1\right), \tag{2}$$

$$log_{S_1}(S_2) = D_{log(S_1)}exp \cdot \left(log(S_2) - log(S_1)\right). \tag{3}$$

where the exponential map is defined at point $S_1 \in \mathcal{S}_+^D$, $T_1 \in T_{S_1}\mathcal{S}_+^D$ a tangent vector, and the corresponding point of S_2 at $T_{S_1}\mathcal{S}_+^D$ is $log_{S_1}(S_2)$.

In addition, the geodesic distance between $S_1, S_2 \in \mathcal{S}_+^D$ is defined as follows [3]:

$$\begin{aligned} d_G(S_1, S_2) &= \langle log_{S_1}(S_2), log_{S_1}(S_2) \rangle \\ &= \| log(S_1) - log(S_2) \|_F^2 . \end{aligned} \tag{4}$$

Based on the Log-Euclidean metric, the geodesic distance between two SPD matrices is equivalent to the Euclidean distance between the corresponding logarithmic forms.

2.2. Laplace Operator on SPD Matrix Lie Group

The critical step of LPP algorithm is to construct a Laplacian matrix to represent the intrinsic geometrical structure of vector-form data points. The Laplacian matrix on a graph is a discrete analogue of the Laplace operator that we're familiar with in functional analysis. In this subsection, we give the graph Laplacian matrix construction on SPD matrix Lie group endowed with Log-Euclidean metric. Suppose a parameterized SPD matrix Lie group Σ defined as $\Sigma : \mathcal{R}^d \to \mathcal{S}_+^D$, we denote the vector $\overrightarrow{\Sigma(x)\Sigma(x+u)}$ corresponding to the standard first-order derivative on $\mathcal{S}_+^D$ [14]:

$$\overrightarrow{\Sigma(x)\Sigma(x+u)} = \Sigma(x)^{\frac{1}{2}} \left(log\Sigma(x+u) - log\Sigma(x)\right)\Sigma(x)^{\frac{1}{2}} . \tag{5}$$

For the numerical computation of Laplacian, we may approximate the first and second order tensor derivative by their Euclidean derivative. This gives a fourth order approximation of the Laplace-Beltrami operator:

$$\begin{aligned} \Delta_u\Sigma &= \partial_u^2\Sigma - 2\left(\partial_u\Sigma\right)\Sigma^{(-1)}\left(\partial_u\Sigma\right) \\ &= \overrightarrow{\Sigma(x)\Sigma(x+u)} + \overrightarrow{\Sigma(x)\Sigma(x-u)} + O\left(\|u\|^4\right) \end{aligned} \tag{6}$$

To compute the complete manifold Laplacian of Eq.6, we just have to compute the above numerical approximations of the function derivatives along d orthonormal basis vectors.

In practical application, suppose $\{\Sigma_1, \Sigma_2, \cdots, \Sigma_N\}$ is a set of SPD matrices generated from Σ. The 'normal' graph Laplacian on this data set is:

$$\left(\Delta_{nm}\Sigma_i\right) = \Sigma_i^{\frac{1}{2}} \left(log(\Sigma_i) - \sum_{j=1}^{n} W_{ij} log(\Sigma_j) \right) \Sigma_i^{\frac{1}{2}} \tag{7}$$

where $W_{ij} = e^{-\frac{\|log(\Sigma_i) - log(\Sigma_j)\|_F^2}{t}}$, if i and j are connected, else $W_{ij} = 0$. The graph Laplacian matrix on SPD matrix Lie group is $L = Q - W$, where W is a symmetric matrix, W_{ij} defined as above and Q is a diagonal matrix, $Q_{ii} = \sum_j W_{ji}$.

2.3. Lie-LPP Algorithm

Based on the definition of graph Laplacian matrix on $\mathcal{S}_+^D$ and the algorithmic procedure of LPP, we give Lie-LPP algorithm.

Suppose the input data points are $S_1, S_2, \ldots, S_N \in \mathcal{S}_+^D$, and the learnt output data points are $Y_1, Y_2, \ldots, Y_N \in \mathcal{S}_+^d$, where N is the number of sampled points. For SPD matrix Lie group $\mathcal{S}_+^D$, the SPD matrix logarithms in the tangent space are also symmetric matrices. The linear dimension reduction mapping between tangent spaces is:

$$f\left(log(S_i)\right) = A^T log(S_i) A. \tag{8}$$

The corresponding mapping between $\mathcal{S}_+^D$ and $\mathcal{S}_+^d$ is :

$$F(S_i) = exp \circ f\left(log(S_i)\right) = exp\left(A^T log(S_i) A\right), \tag{9}$$

where $S_i \in \mathcal{S}_+^D$, $F(S_i) \in \mathcal{S}_+^d$, A is a linear dimensionality reduction map.

It is easy to prove that $F(S_i)$ is still an SPD matrix. In this paper we attempt to learn the transformation matrix A, where $A \in \mathcal{R}^{D \times d}$ is a full column rank matrix, $D >> d$.

In order to obtain a more discriminative SPD matrix Lie group $\mathcal{S}_+^d$, A should also inherit and preserve the geometric structure of $\mathcal{S}_+^D$. According to the idea of LPP, the key step of Lie-LPP algorithm is to construct the Laplacian matrix L on $\mathcal{S}_+^D$ which reflects the local geometrical structure of $\mathcal{S}_+^D$.

The algorithm steps of Lie-LPP are shown as follows:

- The first step is to find the K-nearest neighborhoods $\{U_i\}$ of every point S_i. We choose the geodesic distance to measure the similarity among SPD matrices.
- The second step is to construct the weight matrix W on N input data points.
  $W_{ij} = e^{-\frac{\|log(S_i)-log(S_j)\|_F^2}{t}}$, if $S_j \in U_i$,
  $W_{ij} = 0$, else if $S_j \notin U_i$.
- The third step is to compute the eigenvectors and eigenvalues for the generalized eigenfunction problem:
  $$S^T LSA = \lambda S^T QSA$$
  where $S = [log(S_1), log(S_2), \cdots, log(S_N)]^T$ is a partitioned matrix.
  Details will be shown in the next subsection.

2.4. Optimal Embedding

The optimal dimensionality reduction map A is learnt by minimizing the following energy function:

$$\frac{1}{2}\sum_{i,j} d_G(Y_i, Y_j) W_{ij}, \tag{10}$$

where $d_G(Y_i, Y_j)$ is the geodesic distance between Y_i and Y_j while W_{ij} is the corresponding weight.

According to the definition of geodesic distance and Log-Euclidean metric on SPD matrix Lie group in Eq.4, the energy function Eq.10 can be transformed to the following equation:

$$\frac{1}{2}\sum_{i,j} \| logY_i - logY_j \|_F^2 W_{ij}, \tag{11}$$

According to Eq.8, the optimization function Eq.11 is represented as follows:

$$\begin{aligned} &\frac{1}{2}\sum_{i,j} \| A^T logS_i A - A^T logS_j A \|_F^2 W_{ij} \\ &= tr\left(P^T S^T (Q-W) SP\right) \\ &= tr\left(P^T S^T LSP\right), \end{aligned} \tag{12}$$

$P = AA^T \in R^{D\times D}$, P and $L = Q - W$ are all semi-SPD matrices. So $P^T S^T (Q-W) SP$ is a semi-SPD matrix, the eigenvalues of it are all non-negative. We have $tr\left(P^T S^T LSP\right) \geq 0$. In order to compute the minimum value of Eq.12, we just need to compute the minimum eigenvalues of matrix $P^T S^T LSP$.
To avoid getting a singular solution, we impose a constraint: $tr\left(P^T S^T QSP\right) = 1$.
Then the corresponding minimization problem turns to:

$$\begin{aligned} min \quad & tr(P^T S^T LSP), \\ s.t. \quad & tr\left(P^T S^T QSP\right) = 1. \end{aligned} \tag{13}$$

We use Lagrange multiplier method to solve the minimization problem:

$$L(A, \lambda) = tr\left(P^T S^T LSP\right) - \lambda\left(tr\left(P^T S^T QSP\right) - 1\right). \tag{14}$$

As L's derivative to A we obtain the following:

$$\begin{aligned} \frac{\partial L(A,\lambda)}{A} &= 4tr\left(A^T AA^T S^T LS - \lambda A^T AA^T S^T QS\right) \\ &= 4tr\left(\left(S^T LSA - \lambda S^T QSA\right) A^T A\right). \end{aligned} \tag{15}$$

Since A is a $D \times d$ full rank matrix, $A^T A$ is a $d \times d$ SPD matrix. For obtaining $\frac{\partial L(A,\lambda)}{\partial A} = 0$, we just need solving the following generalized eigenvector problem:

$$S^T LSA = \lambda S^T QSA. \tag{16}$$

We obtain the bottom smallest d eigenvalues $0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_d$, and eigenvectors $A_1, A_2, \ldots, A_d \in \mathcal{R}^{D\times 1}$. $A = [A_1, A_2, \ldots, A_d] \in \mathcal{R}^{D\times d}$ is the linear dimensionality reduction map matrix. The corresponding dimension reduction map g between $\mathcal{S}_+^D$ and $\mathcal{S}_+^d$ is:

$$Y_i = F(S_i) = exp\left(A^T log(S_i) A\right). \tag{17}$$

The lower dimensional SPD matrix Lie group $\mathcal{S}_+^d$ preserves the local geometric and algebraic structure of $\mathcal{S}_+^D$. The local structure between $\mathcal{S}_+^D$ and $\mathcal{S}_+^d$ is kept by the Laplacian matrix L on $\mathcal{S}_+^D$.

3. Experiments

In this section we test our algorithms on two face image databases e.g. Yale Face Database and YouTube Celebrities Database. Correspondingly we use LEML algorithm [3] and SPD-ML algorithm [12] to compare with our algorithm.

3.1. Databases

Yale Face database contains 165 grayscale images of 15 different people. There are 11 images per subject, one per different facial expression or configuration The face region in each image is resized into 32×32. We use the raw intensity feature to construct the corresponding SPD matrix, in which we follow [3] to construct the SPD matrix for each image. On this dataset since the image size is 32×32, the size of the corresponding SPD matrix is 1024×1024.

YTC database collects 1910 video sequences of 47 subjects from YouTube. The face regions of the original images from the video sequences are not all the same and especially high. We resize the face region into 20×20 intensity image as the same in [3]. Each video clip generates an image set of faces. In this experiment, we choose 497

TABLE 1. CLASSIFICATION PERFORMANCE OF YALE FACE DB, YTC DB, TOGETHER WITH THE COMPARISON RESULTS FOR LIE-LPP AND TRADITIONAL MANIFOLD LEARNING ALGORITHMS PCA, LPP AS WELL AS LEML, SPD-ML-STAIN, SPD-ML-AIRM.

YaleFace DB	PCA [13]	LPP [2]	LEML [3]	SPD-ML-Stain [12]	SPD-ML-Airm [12]	Lie-LPP
YFD-trn2/tst9	**46.2 ± 3.2**	44.5 ± 3.5	43.8 ± 2.6	43.0 ± 1.9	44.1 ± 2.3	45.3 ± 2.4
YFD-trn3/tst8	48.3 ± 2.8	**56.4 ± 4.5**	50.8 ± 1.2	51.7 ± 1.8	52.6 ± 2.7	55.2 ± 3.2
YFD-trn4/tst7	52.7 ± 1.9	62.8 ± 4.8	51.2 ± 1.7	50.5 ± 2.3	51.2 ± 3.6	**63.4 ± 2.7**
YFD-trn5/tst6	54.3 ± 2.7	67.2 ± 2.4	73.6 ± 4.6	70.4 ± 3.7	69.3 ± 1.9	**73.9 ± 1.8**
YTC DB	PCA [13]	LPP [2]	LEML [3]	SPD-ML-Stain [12]	SPD-ML-Airm [12]	Lie-LPP
YTC-trn3/tst7	44.2 ± 2.2	56.5 ± 1.4	**67.5 ± 2.2**	57.3 ± 1.9	58.5 ± 2.3	66.3 ± 2.1
YTC-trn4/tst6	47.3 ± 2.2	58.6 ± 2.3	68.2 ± 1.4	59.7 ± 1.8	59.6 ± 1.7	**70.5 ± 2.6**
YTC-trn5/tst5	49.6 ± 1.5	59.8 ± 2.1	70.3 ± 1.7	60.5 ± 2.3	62.2 ± 1.6	**76.1 ± 1.6**
YTC-trn6/tst4	50.1 ± 1.7	61.2 ± 1.7	73.6 ± 2.6	61.6 ± 1.7	64.6 ± 1.9	**78.5 ± 2.2**

video sequences of 10 different people. As the same in Yale Face Database, we use the 20×20 intensity feature to construct the corresponding SPD matrix descriptor. So the size of the corresponding SPD matrix descriptor is 400×400.

3.2. Recognition

These experiments are mainly divided into two steps. The first step is to use these algorithms to reduce the dimension of data points. The second step is to do the human face recognition in the low dimensional space. In the recognition step, Lie-LPP uses Log-Euclidean metric the same with LEML [3] to compute the geodesic distances between arbitrary two SPD matrices. SPD-ML [12] used Affine-Invariant metric as well as Stein divergence metric. Different from the other two algorithms LEML and SPD-ML, our algorithm aims to construct Laplacian matrix on SPD matrix Lie group, and then learns a more discriminable Lie group which preserves the geometric and algebra structure of the original one.

For YaleFace DB, we totally do the experiment four times by each algorithm. In each time, we randomly choose $p\,(p = 2, 3, 4, 5)$ image sets per subject as the training dataset, the rest $11 - p$ as the test dataset respectively. The recognition accuracy results of different algorithms are reported in Table 1. In these experiments we choose the same classification methods with LEML and SPD-ML in the recognition step. From Table 1 we can see that the recognition result of our proposed algorithm is especially similar to the results of LEML and SPD-ML when we choose $p = 5$ images per subject as the training dataset. But the accuracy recognition rates of Lie-LPP are higher than the accuracies of LEML and SPD-ML when we choose $p = 2, 3, 4$. The results of these experiments mean that the effect of dimensionality reduction for SPD matrices by Lie-LPP is better than the effects of LEML and SPD-ML. In addition, LEML and SPD-ML need several parameters when performing their algorithms. Our algorithm just needs to analyze the Laplace operator of SPD matrix Lie group and solves the dimensionality reduction problem directly on the SPD matrix Lie group.

For the recognition of YTC Database, we also do the experiments four times by each algorithm respectively. As the same method with YaleFace DB, we randomly choose $p\,(p = 3, 4, 5, 6)$ respectively image sets per subject for training, the rest for test. The results of all these experiments are presented in Table 1 lower part. In Table 1, we can see that the recognition of Lie-LPP is higher than the other four algorithms, with more than 20% average improvement. Compared with SPD-ML and LEML algorithms, our algorithm achieves comparable performance with them for $p = 3$. In addition, for $p = 4, 5, 6$, our proposed method outperforms them. The results show that constructing the corresponding graph Laplacian matrix on SPD matrix Lie group is a necessary step to uncover the intrinsic structure of SPD matrix Lie group.

4. Conclusion and Future Directions

In this paper, we extend manifold learning algorithm LPP to SPD Matrix Lie group, and successfully apply it to human face recognition. We use the relationship between Lie group and Lie algebra, and construct the discrete Laplace matrix of SPD matrices. Finally, we map the high-dimensional SPD Matrix Lie group to lower-dimensional SPD Matrix Lie group directly, which can preserve the local geometrical and algebraic structure of the original Lie group. Among all the dimensionality reduction algorithms, our method is the most direct and simple one which just using local neighborhood structure of SPD Matrix Lie group. As the introduction of Lie-LPP algorithm, we lead manifold learning algorithm to Lie group. It is the first time that we introduce Lie-LPP algorithm based on LPP to SPD Matrix Lie group. From the results of these experiments, it has a good application on human face recognition.

In the future, we will go a further step forwards to improve this algorithm. In practice, we will try to do it on the human action recognition. Furthermore, we will give the detailed theoretical analysis of Lie-LPP.

Acknowledgments

This work is supported by the National Key Research and Development Program of China under grant 2016YFB1000902, NSFC project No.61232015, No.61472412, No.61621003, the Beijing Science and Technology Project: Machine Learning based Stomatology and Tsinghua-Tencent-AMSS-Joint Project: WWW Knowledge Structure and its Application.

References

[1] A. J Ma, Pong C Yuen, Wilman W W Zou and Jh Lai, *Supervised Spatio-temporal Neighborhood Topology Learning for Action Recognition.* IEEE Trans. Circuits and Systems for Video Technology, vol. 23, lss. 8, pp. 1447-1460, 2013.

[2] Xiaofei He and Partha Niyogi, *Locality Preserving Projections.* Presented at the Int. Conf Advances in Neural Information Processing Systems, 2003.

[3] Z. W Huang, R. P Wang, S. G Shan, X. Q Li and X. L Chen, *Log-Euclidean Metric Learning on Symmetric Positive Definite Manifold with Application to Image Set Classification.* In ICML, 2015.

[4] V. Kwatra and Mei Han, *Fast Covariance Computation and Dimensionality Reduction for Sub-Window Features in Image.* In ECCV'10 Proceedings of the 11th European conference on Computer vision: Part II, 2010.

[5] Oncel Tuzel, Fatih Porikli and Peter Meer, *Pedestrian Detection Via Classification on Riemannian Manifolds.* In IEEE Transaction on PAMI, 2008.

[6] Fatih Porikli and Oncel Tuzel, *Fast Construction of Covariance Matrices for Arbitrary Size Image Windows.* In ICIP, 2006.

[7] Oncel Tuzel, Fatih Porikli and Peter Meer, *Region Covariance: A Fast Descriptor for Detection and Classification.* In ECCV, 2006.

[8] S. Roweis and L. Saul, *Nonlinear Dimensionality Reduction by Locally Linear Embedding.* Science, vol. 290, 2000.

[9] J. B. Tenenbaum, V. de Silva and J. C. Langford, *A Global Geometric Framework for Nonlinear Dimensionality Reduction.* Science, vol. 290, 2000.

[10] M. Belkin and P. Niyogi, *Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering.* In proc. Int. Conf. Advances in Neural Information Processing Systems, 2001.

[11] M. T Harandi, C. Sanderson, A. Wiliem, B. C. Lovell, *Kernel Analysis over Riemannian Manifolds for Visual Recognition of Actions, Pedestrians and Textures.* In IEEE Workshop on the Applications of Computer Vision, 2012.

[12] Harandi M. T., Salzmann M. and Hartley R. , *From Manifold to Manifold: Geometry-aware Dimensionality Reduction for SPD Matrices* In ECCV, 2014.

[13] Lindsay I Smith, *A Tutorial on Principal Components Analysis.* February 26, 2002.

[14] X. Pennec, P. Fillard and N. Ayache, *A Riemannian framework for tensor computing,* IJCV, 66(1):41-66, 2006.

[15] V. Arsigny, P. Fillard, X. Pennec and N. Ayache, *Geometric means in a novel vector space structure on symmetric positive-definite matrices,* SIAM J.Matrix Analysis and Applications, 29(1):328-347, 2007.