National Research University Higher School of Economics

Faculty of computer science

Statistical Learning Theory Programme

Term paper

Depression and Epilepsy Classification by Riemannian Geometry

Student:	Kuzina Anna
Supervisor:	Burnaev Evgeny

Moscow

2018

Contents

1 Introduction

This work is mostly devoted to the detection of the depression and epilepsy from functional magnetic resonance imaging (or fMRI) data. Depression is the most prevalent psychiatric illnesses nowadays, which not only influence patients' behaviour and perception of the environment but also causes deviations in the structure and functioning of their brain. In addition, more than one-third of people who suffer from epilepsy also have depression and both deceases potentially share similar pathological pathways since fMRI of patients with depression revealed abnormal functional connectivity measured during the resting state. However, these connections are not studied yet and there are no biological tests which can effectively detect depression.

Raw fMRI sequences were preprocessed in SPM toolbox, then manual ICA denoising in FSL Melodic toolbox was performed. Further, the functional connectivity matrices for 117 brain regions were calculated in Nilearn using the previously denoised data. Effectively, these matrices represent pairwise correlations between all possible pairs of brain regions. To obtain sparse adjacency matrices of the functional brain network different correlation thresholds were applied. These matrices are symmetric. We further take their Laplacian, which is always symmetric non-negative and add small regularization to the diagonal elements to achieve positive definiteness.

SPD matrices are commonly used with medical images. In [2], [4] classification of SPD matrices is used for signal processing framework in brain-computer interface. Authors in [19] use a similar approach to classify structural connectomes. In [18] and [1] the same concept is applied to diffusion tensor magnetic resonance imaging.

The paper is organized as follows: we begin with theoretical part, which discusses the geometry of the space of SPD matrices, derive affine invariant metric and distances on the Riemannian manifold. Then classification of SPD matrices is described, we discuss several approached including classification on the manifold, kernel methods and Riemannian network. Finally, the last section presents the result, which was obtained for depression and epilepsy classification by these methods.

2 Differential Geometry in Space of SPD Matrices

2.1 Notations and Definitions

- $M(n)$ — space of $n \times n$ real matrices
- $GL(n)$ — General linear group, set of $n \times n$ invertible matrices
- $S(n) = \{S \in \mathcal{M}(n) : S^T = S\}$ — space of $n \times n$ symmetric matrices.
  $S(n)$ is a $\frac{n(n+1)}{2}$ vector space.
- $P(n) = \{P \in S(n) : u^T P u > 0 \; \forall u \in \mathbb{R}^n\}$ — space of $n \times n$ SPD matrices

Statement 2.1. *Matrix exponential is defined and unique for all $S \in S(n)$*

$$\exp(S) = P = C \; diag(e^{\lambda_1}, \dots, e^{\lambda_n}) \; C^T \in P(n)$$

Proof. Matrix exponential is defined by the power series:

$$\mathrm{e}^X = \sum_{k=0}^{\infty} \frac{1}{k!} X^k$$

which converges for all square matrices.
If we consider only symmetric ones, there exists spectral decomposition $S = C\Lambda C^T$, where C — unitary matrix ($CC^T = I$.) and Λ — diagonal. Then $S^i = C\Lambda C^T \cdot C\Lambda C^T \cdots C\Lambda C^T = C\Lambda^i C^T$ and expression for the exponent takes the following form:

$$\mathrm{e}^S = \sum_{k=0}^{\infty} \frac{1}{k!} C\lambda^k C^T = C \left(\sum_{k=0}^{\infty} \frac{1}{k!} \lambda^k \right) C^T = C \mathrm{diag}(\mathrm{e}^{\lambda_1}, \dots, \mathrm{e}^{\lambda_n}) C^T$$

Uniqueness follows from the fact that both P and e^S have the same eigenvectors (from matrix C) and their eigenvalues are uniquely related. $\square$

Statement 2.2. *For all SPD matrices logarithm function is defined as an inverse of the matrix exponential:*

$$\log(P) = S = C \; diag(\log(\lambda_1), \dots, \log(\lambda_n)) \; C^T \in S(n)$$

Proof. Matrix logarithm here is defined as an inverse of the matrix exponential. Since P is symmetric positive definite, it is always diagonalizable with Λ having all positive values. Therefore $\log(\lambda_i)$ exists and $\mathrm{e}^{\log(\lambda_i)} = \lambda_i$. $\square$

So, we have constructed a bijection between vector space $S(n)$ and space $P(n)$, therefore, we endowed $P(n)$ with a smooth structure.

Statement 2.3. $P^{\frac{1}{2}}$ — *symmetric matrix A, s.t. $AA = P$ is defined $\forall P \in P(n)$ and unique*

Proof. Existence: Since P is a SPD matrix, it is diagonalizable and all its eigenvalues are positive, i.e. $P = UDU^T = U\text{diag}(d_i)U^T, \quad d_i > 0$. Therefore, $\exists\, D^{\frac{1}{2}} = \text{diag}(\sqrt{d_i})$ and $(UD^{\frac{1}{2}}U^T)^2 = UDU^T = P$

Uniqueness: Assume, there exists A and B: $A^2 = P$ and $B^2 = P$. This means that A^2 commutes B^2 and both can be diagonalized with the same basis and $D_1^2 = D_2^2$ (where $A = UD_1U^T,\ B = UD_2U^T$). But since both A and B are positive definite, all elements of diagonal matrices D_1 and D_2 are positive $\Rightarrow\ D_1 = D_2\ \Rightarrow A = B$ $\square$

Space of SPD matrices is a differentiable $n(n+1)/2$ dimensional Riemannian manifold $\mathcal{M}$, more specifically, it forms an open cone on the space $S(n)$. Derivative at the point P lies on the tangent space $T_P\mathcal{M}$, which is also a subspace of $S(n)$, moreover tangent space is a finite-dimensional Euclidean space with inner product g_P. An important property of the Riemannian manifold is that metrics g_P varies smoothly from one point on the manifold to another and it is usually called **Riemannian metric**

2.2 Affine-invariant metric

To begin with, let us give $P(n)$ a structure of a Lie group

Definition 2.1. Smooth manifold $\mathcal{M}$, which is also a group, satisfying group operation of multiplication:

$$\mathcal{M} \times \mathcal{M} \to \mathcal{M}: \quad P_1, P_2 \to P_1 \odot P_2$$

and inversion:

$$\mathcal{M} \to \mathcal{M}: \quad P \to P^{-1}$$

is called a **Lie group**.

Using Statements 2.1 and 2.2 we showed that $P(n)$ is a smooth manifold. As a multiplication we will use the following smooth operation [18]:

$$P_1 \odot P_2 = P_1^{\frac{1}{2}} P_2 P_1^{\frac{1}{2}}$$

It can be easily shown that this multiplication conserves identity and inversion, moreover, if both matrices are symmetric positive definite, then $P_1 \odot P_2$ is also an SPD matrix.

One of the most popular metrics in the literature is Affine invariant [12], [18] or natural metric [10].

To determine the metric, let us define the scalar product on the tangent space at the identity matrix to be a standard Frobenius inner product:

$$\langle S_1, S_2 \rangle_I = Tr(S_1 S_2^T) = Tr(S_1 S_2)$$

On general, we want our metric to be invariant under congruent transformations:

$$A \star P = APA^T, \quad \forall\, A \in GL(n)$$

This can be also viewed as a change of basis operation. And we can ensure affine invariance be the identity:

$$\langle S_1, S_2 \rangle_P = \langle A \star S_1, A \star S_2 \rangle_{A \star P}, \quad \forall A \in GL_n,\ S_1, S_2 \in T_P\mathcal{M} \tag{1}$$

Theorem 2.1. [1] *Let $\langle \cdot, \cdot \rangle_I$ be a scalar product on a Lie algebra $T_I\mathcal{M}$. Then any left-invariant metric on a Lie group $\mathcal{M}$ is defined as:*

$$\forall S_1, S_2 \in T_P\mathcal{M} \quad \langle S_1, S_2 \rangle_P = \langle P^{-1} \odot S_1, P^{-1} \odot S_2 \rangle_I$$

Applying introduced multiplication to this theorem we get:

$$\langle S_1, S_2 \rangle_P = \mathrm{Tr}(P^{-\frac{1}{2}} S_1 P^{-1} S_2 P^{-\frac{1}{2}}) = Tr(P^{-1} S_1 P^{-1} S_2) \tag{2}$$

Finally, we can check that affine invariance eq. (1) holds:

$$\begin{aligned}\langle A \star S_1, A \star S_2 \rangle_{A \star P} &= \mathrm{Tr}((APA^T)^{-1} A S_1 A^T (APA^T)^{-1} A S_2 A^T) = \\ &= \mathrm{Tr}(A^{T^{-1}} P^{-1} A^{-1} A S_1 A^T A^{T^{-1}} P^{-1} A^{-1} A S_2 A^T) = \\ &= \mathrm{Tr}(P^{-1} S_1 P^{-1} S_2) = \langle S_1, S_2 \rangle_P \quad \square\end{aligned}$$

Riemannian Norm

Riemannian metric eq. (2) naturally defines norm on any tangent space $T_P\mathcal{M}$:

$$\|S\|_P^2 = Tr((P^{-1}S)^2)$$

Moreover, if we consider $P = I$

$$\|S\|_I^2 = \mathrm{Tr}(S^2) = \mathrm{Tr}(S^T S) = \|S\|_F^2 = \|\mathrm{vect}(S)\|_2^2$$

$$\mathrm{vect}(S) = \left[S_{1,1}, \sqrt{2}S_{1,2}, S_{2,2}, \sqrt{2}S_{1,3}, \sqrt{2}S_{2,3}, S_{3,3}, \ldots, S_{n,n}\right]^T \tag{3}$$

[1] [16], p.181

2.3 Geodesics and distances

Geodesics

Introduced metrics can be used to define invariant distances between objects on the manifold. If we consider any curve on a manifold, we can compute it's instantaneous speed as a norm of the tangent vector at this point and then, integrating this value along the curve, we obtain it's length:

$$\Gamma(t) : [0, 1] \to P(n)$$
$$\mathrm{L}(\Gamma(t)) = \int_0^1 \|\dot{\Gamma(t)}\|_{\Gamma(t)} \mathrm{d}t$$

Further on we will use another fact from the differentiable geometry, which states that for the invariant metrics on an affine symmetric space geodesic is generated by one-parameter subgroups of the given Lie group.[2]

Definition 2.2. Smooth mapping $x(t) : \mathbb{R} \to \mathcal{M}$ is a one-parameter subgroup of a Lie group if for any $t_1, t_2 \in \mathbb{R}$:

$$x(t_1 + t_2) = x(t_1) \odot x(t_2)$$

Statement 2.4. *$x(t) = \exp(tS)$ is a one-parameter subgroup of a Lie group $P(n)$*

Proof. $x(t_1) \odot x(t_2) = \exp(t_1 S)^{\frac{1}{2}} \exp(t_2 S) \exp(t_1 S)^{\frac{1}{2}} = \exp((t_1 + t_2 S) = x(t_1 + t_2)$ □

Therefore, geodesics going through identity with tangent vector S is defined as:

$$\Gamma_{I,S}(t) = \exp(tS)^{\frac{1}{2}} I \exp(tS)^{\frac{1}{2}} = \exp(tS) \tag{4}$$

When it comes to geodesics, we also want to ensure invariance under affine transformations. Any geodesics starting at P with tangent vector S can be seen as $\Gamma_{P,S}(t) = P + tS + o(t^2)$. And consequently:

$$A \star \Gamma_{P,S}(t) = A \star P + tA \star S + o(t^2) = \Gamma_{A\star P, A\star S}(t), \quad \forall A \in GL(n) \tag{5}$$

Since eq. (5) is true for all invertable matrices A, it is also true for $A = P^{-\frac{1}{2}}$:

$$P^{-\frac{1}{2}} \Gamma_{P,S}(t) P^{-\frac{1}{2}} = \Gamma_{I, P^{-\frac{1}{2}} S P^{-\frac{1}{2}}}(t)$$
$$\Gamma_{P,S}(t) = P^{\frac{1}{2}} \Gamma_{I, P^{-\frac{1}{2}} S P^{-\frac{1}{2}}}(t) P^{\frac{1}{2}} = P^{\frac{1}{2}} \exp(t P^{-\frac{1}{2}} S P^{-\frac{1}{2}}) P^{\frac{1}{2}}$$

2 [20], Theorem 5.1, p.232

Exponential Map

One of the most important properties of the Riemannian manifold is that any geodesic $\Gamma_{P,S}(t)$ can be considered as a diffeomorphism from a tanget space $T_P\mathcal{M}$ to the point $\Gamma_{P,S}(1)$ on the manifold. More specifically, geodesics $\Gamma_{P,S}$ at time 1 associates point on the manifold to the tangent vector S, this is called an exponential map and for an intrudies invariant metrics it is defined as:

$$\exp_P(S) = P^{\frac{1}{2}} \exp(P^{-\frac{1}{2}} S P^{-\frac{1}{2}}) P^{\frac{1}{2}}$$

We can also define an inverse operation, which maps point $\exp_P(S)$ on the manifold to the tangent space $T_P\mathcal{M}$:

$$\log_P(P_i) = P^{\frac{1}{2}} \log(P^{-\frac{1}{2}} P_i P^{-\frac{1}{2}}) P^{\frac{1}{2}}$$

Distances

Finally, we will derive formulas for distances between points on the manifold.

$$\begin{aligned} L(\Gamma(t)) = \int_0^1 \|\dot{\Gamma(t)}\|_{\Gamma(t)} dt = \int_0^1 \sqrt{\langle \dot{\Gamma(t)}, \dot{\Gamma(t)} \rangle_{\Gamma(t)}} dt = \\ = \int_0^1 \sqrt{Tr\left((\dot{\Gamma(t)}\Gamma(t)^{-1})^2 \right)} dt \end{aligned}$$

Using eq. (4) we can easily compute distance from I to $P = \exp(tS)$:

$$\begin{aligned} &\dot{\Gamma}_{I,S}(t) = C\text{diag}(\lambda_i \exp(t\lambda_i))C^T \\ &\Gamma_{I,S}(t)^{-1} = C\text{diag}(\exp(-t\lambda_i))C^T \\ &\dot{\Gamma}_{I,S}(t)\Gamma_{I,S}(t)^{-1} = C\text{diag}(\lambda_i \exp(t\lambda_i))\text{diag}(\exp(-t\lambda_i))C^T = C\text{diag}(\lambda_i)C^T = S \\ &\sigma_R(I, P) = L(\Gamma_{I,S}(t)) = \int_0^1 \sqrt{Tr\left(S^2\right)} dt = \|S\|_F = \|\log(P)\|_F \end{aligned}$$

Now, we can use the same trick, as in derivation of geodesics:

$$\sigma_R(P_1, P_2) = \sigma_R(A \star P_1, A \star P_2),\ \forall A \in GL(n) \tag{6}$$

That is, distance between two points on the manifold should not change under congruent transformations. Since eq. (6) is true for all possible A, we can take $A = P_1^{-\frac{1}{2}}$:

$$\begin{aligned} \sigma_R(P_1, P_2) = \sigma_R(I, P_1^{-\frac{1}{2}} P_2 P_1^{\frac{1}{2}}) = \|\log(P_1^{-\frac{1}{2}} P_2 P_1^{\frac{1}{2}})\|_F = \\ = \|\log(P_1^{-1} P_2)\|_F = \left(\sum \log^2 \lambda_i\right)^{\frac{1}{2}} \end{aligned} \tag{7}$$

3 Classification of SPD matrices

At this point, we finally defined the structure of our space of SPD matrices. We have explicit formulas to compute distances on the manifold, map matrices to the tangent space at any point and so on. In this section, we will discuss, how this could be used in the classification task.

3.1 Classification in the manifold

Minimum Distance to Riemannian Mean (MDRM)

Following [2], we will begin with the simplest supervised classification algorithm. The idea is to find a mean value for each class $\overline{P}^{(k)}$ $k = 1 \dots K$ using the training set and then assign test sample to the class, whose so-called centre (or mean value) is the closest.

We just need to define mean value. We want it to be point on the manifold, which has the minimum distance to all the points inside a class:

$$\overline{P}^{(k)} = \arg \min_{P \in P(n)} \sum_i \sigma_R^2(P, P_i^{(k)})$$

Thus minimum exists [2], but there is no closed-form solution. Therefore, we need to use algorithm, introduced in [11]:

Algorithm 1 Geometric mean

```
Input: $P_1, \dots P_N \in P(n)$
Output: $\mu^*$ — estimated geometric mean
1: $\mu_0 \leftarrow I$
2: while $\|X_i\|_F > \varepsilon$ do
3:     $X_i \leftarrow \frac{1}{N} \sum_{k=1}^{N} \log_{\mu_i} P_k$
4:     $\mu_{i+1} \leftarrow \exp_{\mu_i} X_i$
```

k-Nearest Neighbors

Another simple supervised algorithm, that could be applied, is k-nearest neighbours. We can apply convention machine learning algorithms for this method, using Riemannian distance eq. (7) instead of Euclidean one.

3.2 Kernel method

The second type of algorithms is based on the projection of data into some hyperplane. At this point, tangent space comes in. In kernel methods, all the matrices are projected into a tangent space and vectorized. Since all tangent spaces are vector spaces, we may apply conventional classification algorithms further on.

Assume, that we have chose a reference point P^* to construct tangent space. Then obviously the mapping function has the form:

$$\phi(P_i) = \log_{P^*}(P_i)$$

And we use it to derive kernel, which is a scalar product on a new (tangent) subspace:

$$\begin{aligned} k_R(P_i, P_j) &= \langle \phi(P_i), \phi(P_j) \rangle_{P^*} = \operatorname{Tr}(\log_{P^*}(P_i) P^{*-1} \log_{P^*}(P_j) P^{*-1}) = \\ &= \operatorname{Tr}(\log(P^{*-\frac{1}{2}} P_i P^{*-\frac{1}{2}}) \log(P^{*-\frac{1}{2}} P_j P^{*-\frac{1}{2}})) = \operatorname{Tr}(\tilde{P}_i \tilde{P}_j) = \\ &= \langle \tilde{P}_i, \tilde{P}_j \rangle_F = \operatorname{vect}(\tilde{P}_i)^T \operatorname{vect}(\tilde{P}_j) = \langle \operatorname{vect}(\tilde{P}_i), \operatorname{vect}(\tilde{P}_j) \rangle_2 \end{aligned} \tag{8}$$

Note that vect() operation here is the same as in eq. (3). Therefore, we are left with vectors of the form $\operatorname{vect}\left(\log(P^{*-\frac{1}{2}} P_j P^{*-\frac{1}{2}})\right)$ in the Euclidean space.

Reference point

If we choose reference point to be equal to identity, we will get so-called log-Euclidean kernel:

$$k_{LE}(P_i, P_j) = \operatorname{Tr}(\log(P_i) \log(P_j))$$

Another solution is to assume that our data is not uniformly distributed among the manifold but lie on some specific area. In that case, it is not optimal to approximate data by a tangent space at identity. So, we can choose P^* to be geometric mean of our data, calculated with algorithm 1.

3.3 Riemannian Network

One may also consider neural networks, which are based on the introduced theory. The idea behind Riemannian network [13] is to be discussed in this section.

SPDNet is based on the idea, which is similar to the convolution networks. First layers consist of alternating linear and non-linear transformations, which shrink dimensionality of the input data. These transformations, thus based on regular convolutions and ReLU, differ in the sense that they preserve the initial structure of the data — outputs still lie in $P(n)$. Then data is projected on the tangent space at identity and vectorized. The overall scheme of the neural network can be seen in fig. 1.

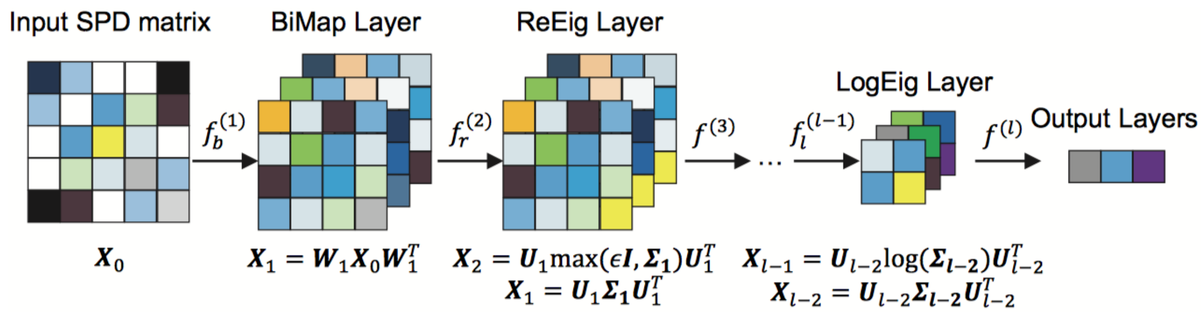

Input SPD matrix
BiMap Layer
ReEig Layer
LogEig Layer
Output Layers
$f_b^{(1)}$
$f_r^{(2)}$
$f^{(3)}$
$f_l^{(l-1)}$
$f^{(l)}$
X_0
$X_1 = W_1 X_0 W_1^T$
$X_2 = U_1 \max(\epsilon I, \Sigma_1) U_1^T$
$X_1 = U_1 \Sigma_1 U_1^T$
$X_{l-1} = U_{l-2} \log(\Sigma_{l-2}) U_{l-2}^T$
$X_{l-2} = U_{l-2} \Sigma_{l-2} U_{l-2}^T$

Figure 1: General SPDNet architecture [13]

BiMap Layer

Analogous to convolutions, BiMap layer produce linear transformation of the input SPD matrix

$$X_k = W_k X_{k-1} W_k^T$$

The inly constraint required is that W_k should be row full-rank matrix.

ReEig Layer

The next layer corresponds to the rectified linear unit (ReLU), which aims to introduce non-linearity in the network, but instead of using restrictions on the values itself, we restrict eigenvalues of the input SPD matrix:

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

Where $X_{k-1} = U_{k-1} \Sigma_{k-1} U_{k-1}^T$ is an eigenvalue decomposition, which is defined for all SPD matrices. Contrary to a simple rectified linear unit

LogEig Layer

This layer implements projection on the tangent plane at Identity. Conceptually, it corresponds to application of Log-Euclidean kernel, discussed in section 3.2.

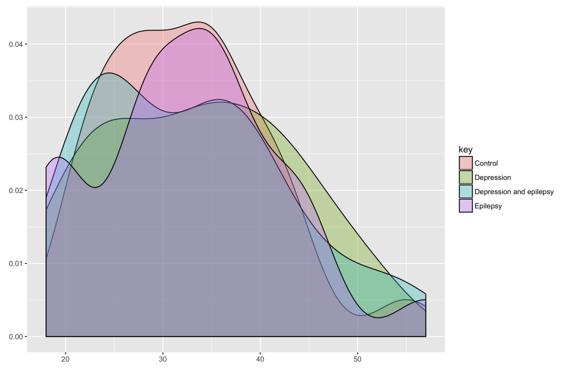
$$X_k = U_{k-1} \log(\Sigma_{k-1}) U_{k-1}^T$$

4 Experiments

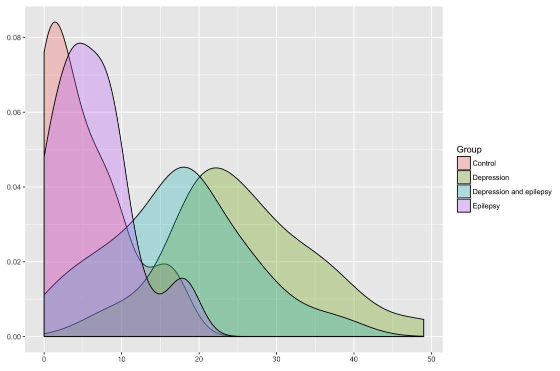
4.1 Data

To implement and compare different methods, introduced in the previous section, we will use dataset provided by Russian Scientific and practical psycho-neurological centre named after Z.P.Solovyov under Skoltech Biomedical Initiative program. The dataset includes 1.5T resting-state functional MRI (fMRI) of the four groups: 25 healthy controls, 25 patients with depression, 25 patients with epilepsy and 25 patients with both depression and epilepsy. For each patient 117×117 symmetric positive definite matrix was created for further analysis.

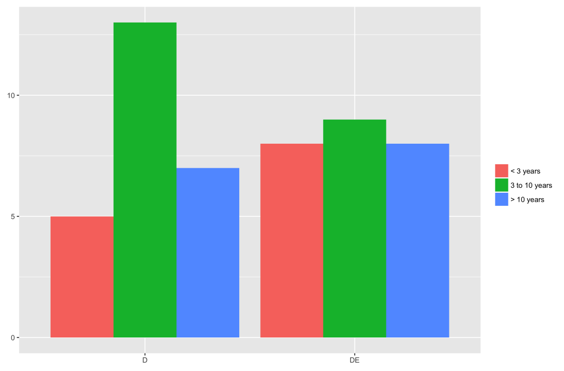
To detect depression The Beck Depression Inventory (BDI), shown on fig. 4a, was used, it is one of the most widely used tests to measure the severity of depression. Dataset also contains demographic and medical data, summarized in fig. 2.



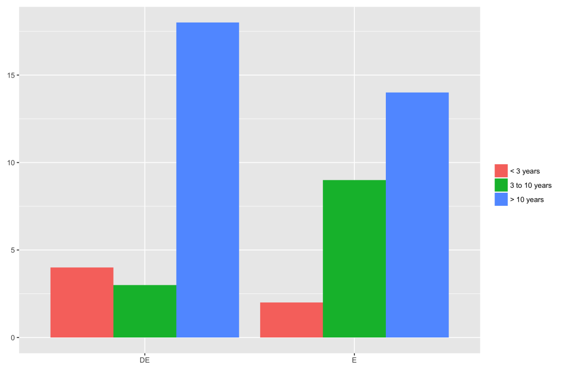
(a) Age distribution



(b) BDI-II results



(c) Duration of the depression



(d) Duration of the epilepsy

Figure 2: Characteristics of the dataset

Overall, 8 classification problems were distinguished:

- Depression detection, 100 patients **(D+DE vs E+C)**
- Epilepsy detection, 100 patients **(E+DE vs D+C)**
- Depression detection, 50 patients **(D vs C)**

- Epilepsy detection, 50 patients **(E vs C)**
- Depression detection in patients with epilepsy, 50 patients **(DE vs E)**
- Epilepsy detection in patients with depression, 50 patients **(DE vs D)**
- Temporal lobe epilepsy detection, 81 patients **(TLE (31) vs C+D (50))**
- Temporal lobe epilepsy detection, 56 patients **(TLE (31) vs C (25))**

4.2 Models

For each classification problem above four models have been estimated, with hyperparameters selected with grid search on 10-fold stratified cross-validation. Below parameters and their possible values are discussed in more detail.

Minimum distance to mean (MDM)

Parameter	Values
Metric	[‘riemann‘, ‘logeuclid‘, ‘euclid“]

Table 1: Parameter grid for MDRM

In this case, only one hyperparameter needs to be chosen: type of metric used for centroid and distance estimation. If we choose `‘riemann‘`, then MDRM from section 3.1 is estimated.

kNN

Parameter	Values
Number of neighbors	1 to 50 with step 5 (1 to 25 for smaller sample)
Metric	[‘riemann‘, ‘logeuclid‘, ‘euclid“]

Table 2: Parameter grid for k-nearest neighbors

In this model, we have two hyperparameters. First one, `number of neighbors`, is a usual parameter for this method, it controls the number of nearest objects, used to identify the class of the current object. Another one appeared, because we again want to compare the Riemannian distance to the Euclidean one.

Support Vector Machine

When it comes to SVM, there are two possible ways to include Riemannian geometry in the model. One may implement so-called kernel SVM, using eq. (8). Another approach, implemented in this work, is to firstly map all the data into the tangent plane and then apply SVM algorithm to the obtained vectors. as we can see from eq. (8), there is a closed form equation for this mapping:

$$\tilde{P}_i = \log(P^{*-\frac{1}{2}} P_i P^{*-\frac{1}{2}})$$

The benefit of this approach is that we may further apply kernel SVM on top of the projected data. As shown in table 3, two different kernels were applied. For `‘rbf‘` kernel parameter `Gamma` controls how much influence a single training example has. Regularization coefficient, penalty parameter of the error term, is also chosen from the grid.

Parameter	Values
kernel	[‘linear‘, ‘rbf‘]
Regularization coefficient C	[1e-04, 1e-2, 1, 10, 100]
gamma	[1e-04, 1e-2, 1, 10]
Metric	[‘riemann‘, ‘logeuclid‘, ‘euclid’‘]

Table 3: Parameter grid for SVM

Logistic Regression

Finally, logistics regression was estimated with different metrics and regularization coefficients, presented in table 4.

Parameter	Values
Regularization coefficient C	[1e-04, 1e-2, 1, 10, 100]
Metric	[‘riemann‘, ‘logeuclid‘, ‘euclid’‘]

Table 4: Parameter grid for LR

4.3 Results

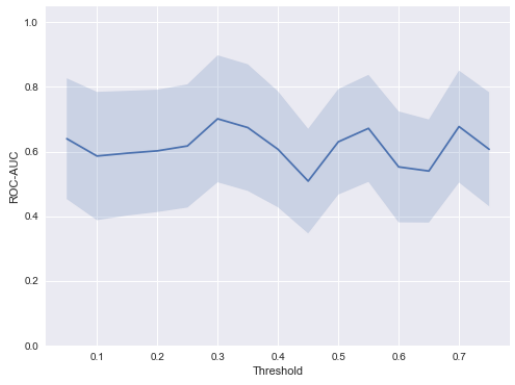
Kernel Methods and classification on manifold

Finally, accuracy and area under ROC-curve for the best models are presented in table 5. The grid search was performed on stratified cross-validation. For models classification problems with 100 observations 10-folds cross-validation was applied, while for ones with smaller sample 5-folds were used.

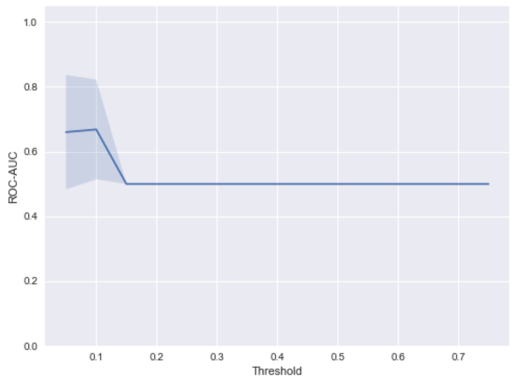
	Best Model	Metric	Accuracy	ROC-AUC
E/no E	SVC	riemann	0.57(0.17)	0.57
D/no D	SVC	riemann	**0.68(015)**	0.70
E/C	SVC	riemann	0.59(0.13)	0.70
D/C	LR	riemann	**0.67(0.22)**	0.70
TLE/no E	LR	euclid	0.62(0.17)	0.60
TLE/C	SVC	riemann	0.57(0.15)	0.57
DE/E	SVC	riemann	**0.70(0.18)**	0.77
DE/D	SVC	riemann	**0.73(0.17)**	0.70

Table 5: Accuracy of the best models on cross-validation

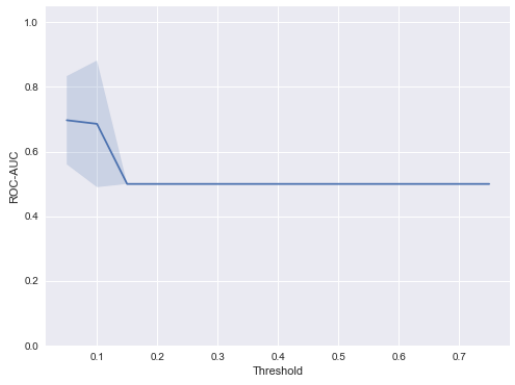
As it was already mentioned, when obtaining adjacency matrix for the graph, correlation threshold was applied. An optimal value for this threshold was chosen together with other hyperparameters from values $[0.1, 0.2, 0.3]$. The sensitivity of ROC-AUC value to the changes in threshold for the four best models is depicted on fig. 3. It can be seen that one could not obtain significantly better results by tuning this hyperparameter.



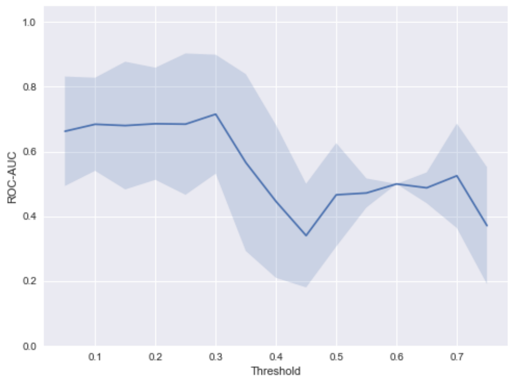
(a) Depression vs Control



(b) Depression vs No depression



(c) Depression with epilepsy vs epilepsy

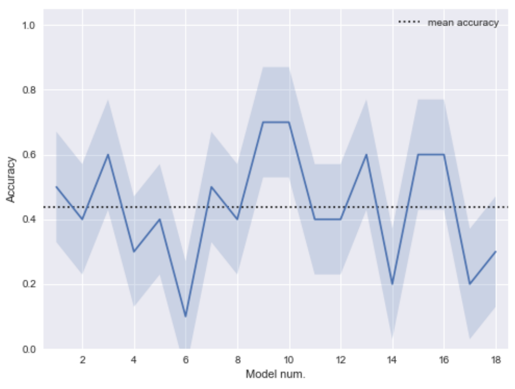


(d) Depression with epilepsy vs depression

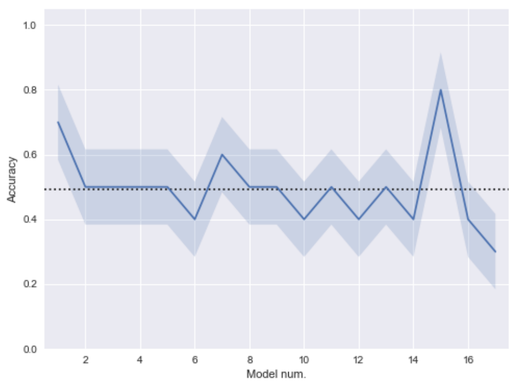
Figure 3: Sensitivity to threshold changes

Riemannian Network

The Riemannian network was also fitted to the described problem and dataset. Implementation from the original paper on Matlab is available on Gihub. Unfortunately, due to the extremely small dataset, there were no reasonable results obtained.



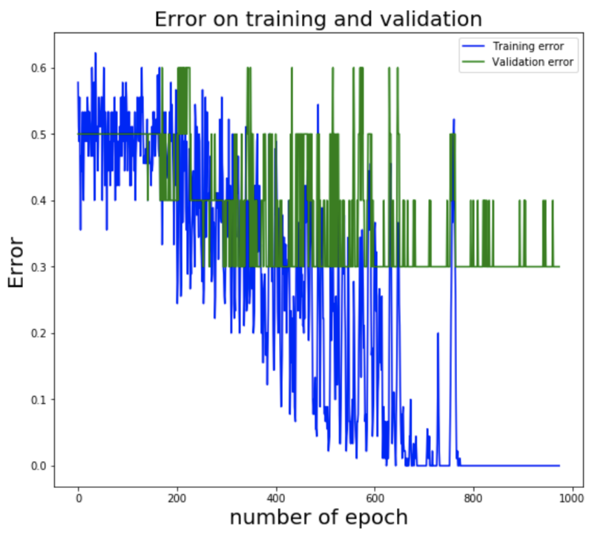
(a) Depression vs Control



(b) Epilepsy vs Control

Figure 4: SPDnet accuracy on cross-validation

When using minimal architecture, with two BiMap + ReEig layers and one LogEig layer, results are very unstable and on average close to the random choice with accuracy $\frac{1}{2}$, as shown of the fig. 4. In most cases, network was not able to learn anything useful from the data. One of the best results with accuracy 0.7 on the validation set is shown on the fig. 5. But again, this is a very unstable, as seen from the total statistics.



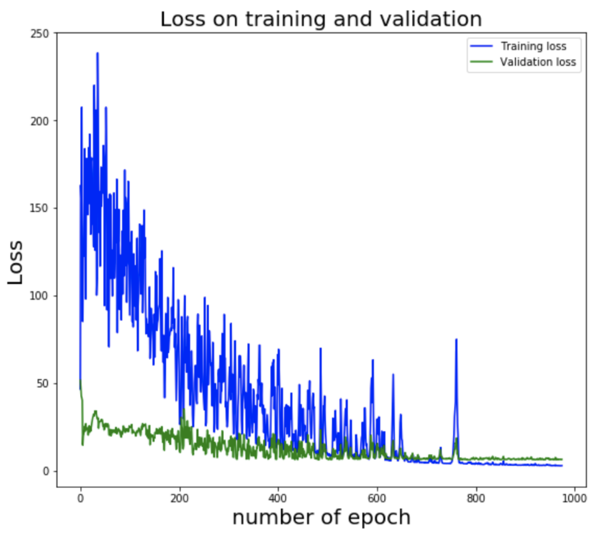


Figure 5: Training of the SPDnet for D/no D problem

Extra hyperparameter tuning did not improve the result. In order to make sure, that network architecture was correctly transferred from Matlab to python and applied to fMRI data the same procedure was repeated with ABIDE dataset [9]. It contains preprocessed fMRI of 539 individuals suffering from ASD and 573 typical controls. When training the same network on this dataset (see fig. 6), the more positive picture can be seen. Moreover, the result is stable when validating it on cross-validation

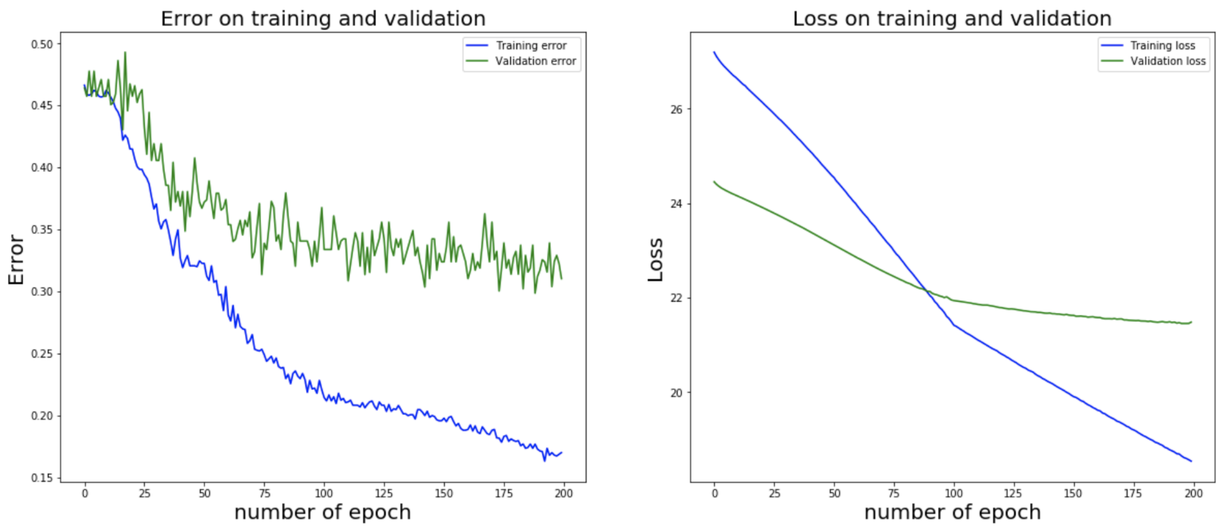


Figure 6: Training of the SPDnet on ABIDE dataset

5 Summary

This work extensively discusses machine learning techniques which can be applied to classification of SPD matrices, taking into account the Riemannian structure of their space. Affine-invariant metric and distances are used to adopt such methods as k-nearest neighbours, support vector machines or even neural network to this special data. Result shows, that utilization of Riemannian metric in most cases outperform Euclidean one on the conventional machine learning algorithms. However, in order to get reasonable results on the SPDnet, a larger dataset is needed.

References

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

[2] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. Riemannian geometry applied to BCI classification. *Lva/Ica 2010*, pages 346–353, 2010.

[3] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. Multiclass brain-computer interface classification by Riemannian geometry. *IEEE Transactions on Biomedical Engineering*, 59(4):920–928, 2012.

[4] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. Classification of covariance matrices using a Riemannian-based kernel for BCI applications. *Neurocomputing*, pages pp.172–178, 2013.

[5] M. Belkin and P. Niyogi. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. *Nips*, 14:585–591, 2001.

[6] M. Berger and B. Gostiaux. *Differential Geometry: Manifolds, Curves, and Surfaces.pdf.* 1927.

[7] S. Bonnabel and R. Sepulchre. Riemannian Metric and Geometric Mean for Positive Semidefinite Matrices of Fixed Rank. 2008.

[8] I. Chavel. Riemannian Geometry. A modern Introduction.

[9] C. Craddock, Y. Benhajali, C. Chu, F. Chouinard, E. Alan, A. Jakab, S. K. Budhachandra, D. L. John, L. Qingyang, M. Milham, C. Yan, and P. Bellec. The Neuro Bureau Preprocessing Initiative: open sharing of preprocessed neuroimaging data and derivatives. *Neuroinformatic*, 2013.

[10] J. Faraut and A. Korányi. *Analysis on Symmetric Cones.* 1995.

[11] T. Fletcher and S. Joshi. Principal Geodesic Analysis on Symmetric Spaces: Statistics of Diffusion Tensors. pages 87–98, 2004.

[12] W. Förstner and B. Moonen. A Metric for Covariance Matrices. *Quo vadis geodesia*, 66:113–128, 1999.

[13] Z. Huang and L. Van Gool. A Riemannian Network for SPD Matrix Learning. pages 2036–2042, 2016.

[14] M. Moakher. A Differential Geometric Approach to the Geometric Mean of Symmetric Positive-Definite Matrices. *SIAM Journal on Matrix Analysis and Applications*, 26(3):735–747, 2005.

[15] M. Moakher and P. Batchelor. Symmetric Positive Definite Matrices: From Geometry to Applications and Visualization. page 17, 2005.

[16] S. Novikov and I. Taymanov. *Modern geometric structures and fields.* 2005.

[17] X. Pennec. Probabilities and Statistics on Riemannian A Geometric approach. *RESEARCH rapport*, (January), 2004.

[18] X. Pennec, P. Fillard, and N. Ayache. A Riemannian Framework For Tensor Computing. *Intl. Journal on Computer Vision*, 66(1):41–66, 2006.

[19] A. Pimkin, M. Belyaev, J. Dudonova, B. Gutman, J. Faskowitz, N. Djahanshad, and P. Tompson. Classification of brain network structures using analysis of symmetric semidefinite matrices. 2017.

[20] S. Sternberg. *Lectures on Differential Geometry.* Prentice–Hall, NJ, 1964.

[21] R. Wang, H. Guo, L. S. Davis, and Q. Dai. Covariance Discriminative Learning : A Natural and Efficient Approach to Image Set Classification. *Computer Vision and Pattern Recognition*, pages 2496–2503, 2012.