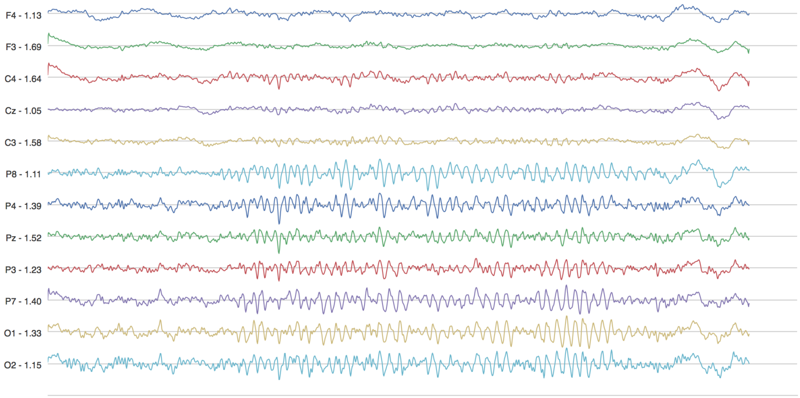
Information Geometry: A framework for manipulation and classification of Neural Timeseries

Alexandre Barachant

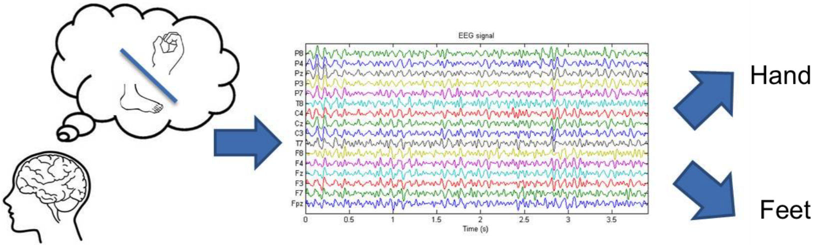
June 29, 2017

Electrical signals from the brain



- *EEG, MEG, ECoG*

Predict the task/condition/stimulus from M/EEG Recording



Context

Decoding brain signals :

- *Brain computer interfaces*
- *Medical diagnosis*
- *Basic Science*

Context

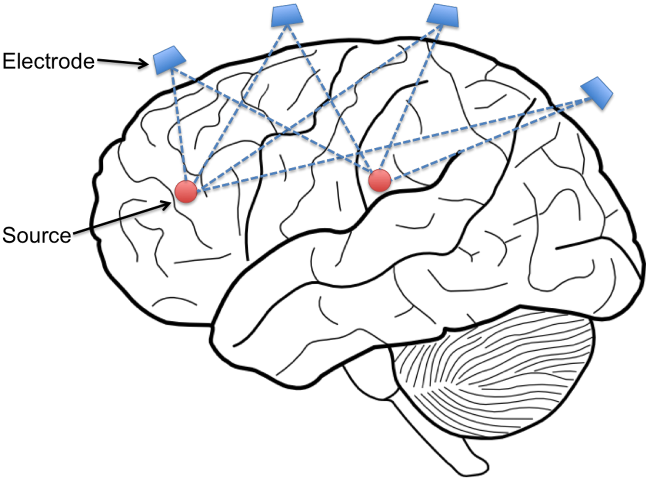
Decoding brain signals :

- *Brain computer interfaces*
- *Medical diagnosis*
- *Basic Science*

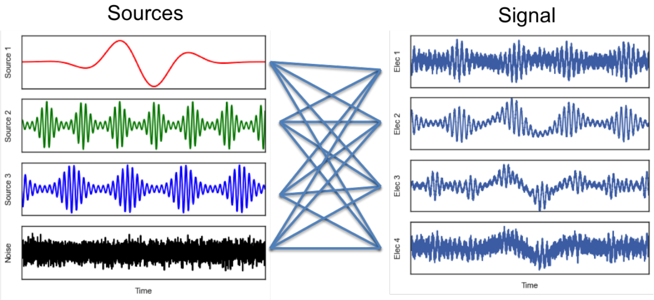
We need better algorithms

- *Adapted to the nature of the signals*
- *Effective with a low number of examples*
- *Generalize across session / subjects*

EEG



EEG

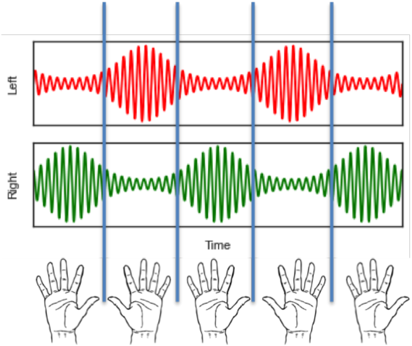


EEG model

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$$

Example : Neural oscillations

Sources are the signal of interest : Example with Motor execution

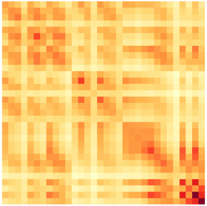


Sources power (variance) depend on the motor task

Build a pipeline to measure change in source signals :

Build a pipeline to measure change in source signals :

EEG Cov

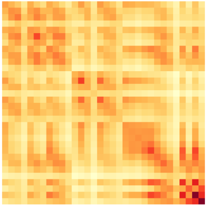


$$\boldsymbol{\Sigma}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^T$$

Feature Extraction pipeline

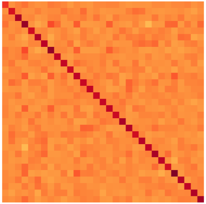
Build a pipeline to measure change in source signals :

EEG Cov



$\mathbf{\Sigma}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^T$

"Source" Cov

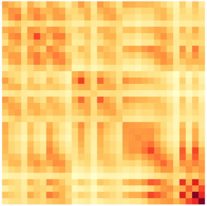


$\mathbf{\Sigma}_s = \mathbf{V}\mathbf{\Sigma}_x\mathbf{V}^T$,
$\mathbf{V} \approx \mathbf{A}^{-1}$

Feature Extraction pipeline

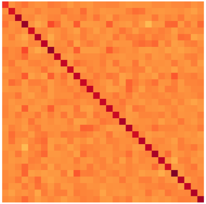
Build a pipeline to measure change in source signals :

EEG Cov



$\mathbf{\Sigma}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^T$

"Source" Cov



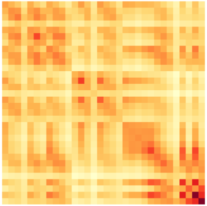
$\mathbf{\Sigma}_s = \mathbf{V}\mathbf{\Sigma}_x\mathbf{V}^T$,
$\mathbf{V} \approx \mathbf{A}^{-1}$

log-variance

$\log(diag(\mathbf{\Sigma}_s))$,

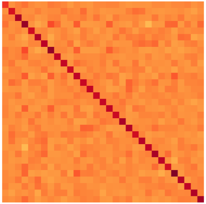
Build a pipeline to measure change in source signals :

EEG Cov



$\mathbf{\Sigma}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^T$

"Source" Cov

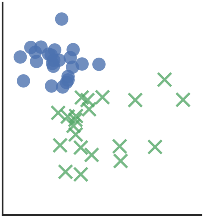


$\mathbf{\Sigma}_s = \mathbf{V}\mathbf{\Sigma}_x\mathbf{V}^T$,
$\mathbf{V} \approx \mathbf{A}^{-1}$

log-variance

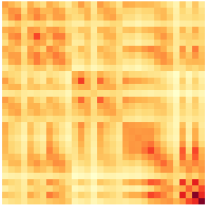
$\log(diag(\mathbf{\Sigma}_s))$,

Features



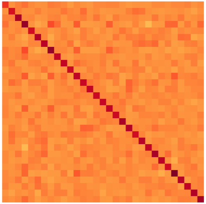
Build a pipeline to measure change in source signals :

EEG Cov



$\mathbf{\Sigma}_x = \frac{1}{N}\mathbf{X}\mathbf{X}^T$

"Source" Cov

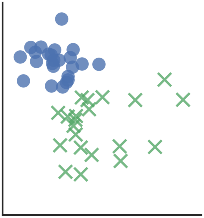


$\mathbf{\Sigma}_s = \mathbf{V}\mathbf{\Sigma}_x\mathbf{V}^T$,
$\mathbf{V} \approx \mathbf{A}^{-1}$

log-variance

$\log(diag(\mathbf{\Sigma}_s))$,

Features



We rely on feature engineering to build of measure of what happen in the sources space.

Limitation of this approach

Problems with Source separation:

- Hit or Miss
- Robustness
- Parameter estimation (consume data)
- Change in mixing matrix $\mathbf{A} \Rightarrow$ Calibration

Is sources separation really necessary ?

- For a classification problem, we don't need the sources
- The Signal Covariance matrix $\mathbf{\Sigma}_x$ contains all the information.
- We only performed manipulation on $\mathbf{\Sigma}_x$.

Let's classify $\mathbf{\Sigma}_x$ (or $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_x)$) !

Option 1: vectorization

- Ignore the SPD structure of the matrices.
- Non-linear

Classifying Covariance matrices

Option 1: vectorization

- Ignore the SPD structure of the matrices.
- Non-linear

Option 2: Kullback leibler divergence

- Not a true metric.
- limited to a few classification algorithm

Classifying Covariance matrices

Option 1: vectorization

- Ignore the SPD structure of the matrices.
- Non-linear

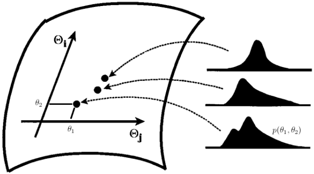
Option 2: Kullback leibler divergence

- Not a true metric.
- limited to a few classification algorithm

Option 3: Information Geometry

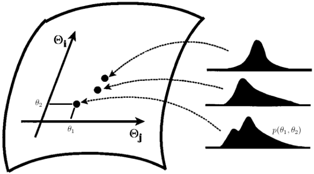
Probability distributions :

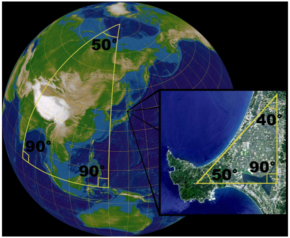
- Points of a Riemannian Manifold.



Probability distributions :

- Points of a Riemannian Manifold.



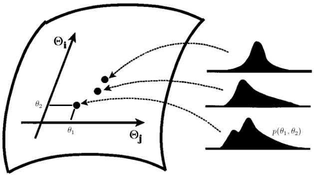


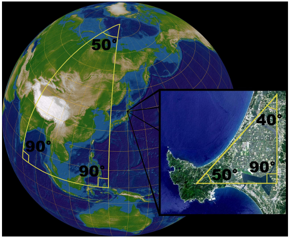
Riemannian Manifold

- Topological space (curved)
- Euclidean geometry only valid locally
- Has a metric

Probability distributions :

- Points of a Riemannian Manifold.





Riemannian Manifold

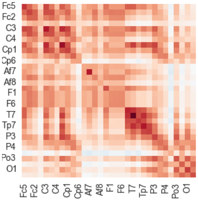
- Topological space (curved)
- Euclidean geometry only valid locally
- Has a metric

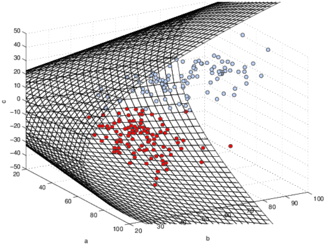
The Fisher information matrix is the "Natural" metric

Multivariate Normal distribution

Covariance matrices ?

- Multivariate Normal distribution
- $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$





- Manifold dimension : $C(C+1)/2$
- Metric : $\frac{1}{2}\mathrm{Tr}\left(\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_i}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{\Sigma}}{\partial\theta_j}\right)$

from the metric :

- Distance, geodesic, mean, median, tangent space, gradient

Riemannian Distance

Riemannian Distance

$$\delta_R(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = \|\log\left(\mathbf{\Sigma}_1^{-1/2}\mathbf{\Sigma}_2\mathbf{\Sigma}_1^{-1/2}\right)\|_F = \left[\sum_{c=1}^{C} \log^2 \lambda_c\right]^{1/2}$$

Key property :

- Invariance by affine transformation, i.e, $\forall\, \mathbf{A} \in \mathcal{G}l(C)$

$$\delta_R(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) = \delta_R(\mathbf{A}\mathbf{\Sigma}_1\mathbf{A}^T, \mathbf{A}\mathbf{\Sigma}_2\mathbf{A}^T).$$

Information Geometry and Source separation

- linear mixing model + invariance:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \Longrightarrow \mathbf{\Sigma}_x = \mathbf{A}\mathbf{\Sigma}_s\mathbf{A}^T$$

- The Riemannian distances in the sensor space and in the source space are equal!

$$\delta_R(\mathbf{\Sigma}_{x,1}, \mathbf{\Sigma}_{x,2}) = \delta_R(\mathbf{\Sigma}_{s,1}, \mathbf{\Sigma}_{s,2}).$$

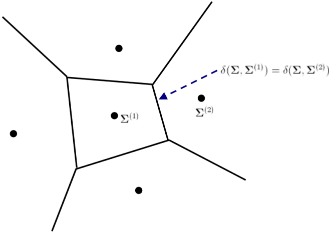
- Uncorrelated sources :

$$\delta_R(\mathbf{\Sigma}_{x,1}, \mathbf{\Sigma}_{x,2}) = \| \log(\sigma_{s,1}) - \log(\sigma_{s,2}) \|_2$$

The Riemannian distance measure change in sources log-variance

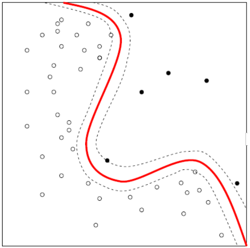
Distance Based classification

- k-NN
- Nearest centroid
- k-Means



Tangent Space classification

- Logistic Regression
- SVM
- Neural Networks



Benefits of Riemannian Geometry

Benefits:

- Measure sources variance without source separation.
- No assumption about the sources covariance.
- Better generalization between subjects and sessions.

Benefits of Riemannian Geometry

Benefits:

- Measure sources variance without source separation.
- No assumption about the sources covariance.
- Better generalization between subjects and sessions.

Inconvenient:

- Computational cost
- Numerical issue for high dimension

Classification of Imagined movement

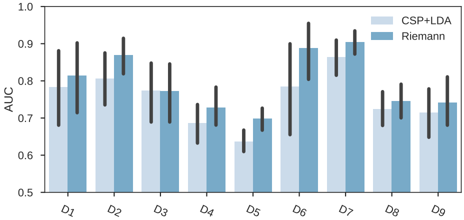
Dataset	Tasks	# subjects	# channels
D1	left hand, right hand	9	22
D2	feet, right hand	14	15
D3	feet, right hand	9	3
D4	feet, right hand	12	13
D5	left hand, right hand	109	64
D6	right hand, feet	8	16
D7	right hand, left hand	14	11
D8	right hand, left hand	52	64
D9	right hand, left hand	29	30

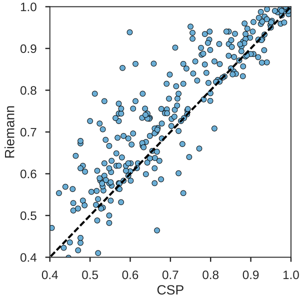
Preprocessing : BP filter, 7-35 Hz.

Classification :

1. CSP + LDA.
2. Cov + Tangent Space + Logistic Regression.

Results

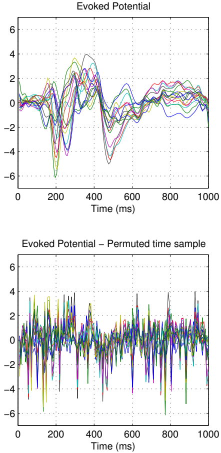

1.0
0.9
0.8
0.7
0.6
0.5
AUC
CSP+LDA
Riemann
D1
D2
D3
D4
D5
D6
D7
D8
D9


1.0
0.9
0.8
0.7
0.6
0.5
0.4
Riemann
0.4
0.5
0.6
0.7
0.8
0.9
1.0
CSP

Event Related Potential ?

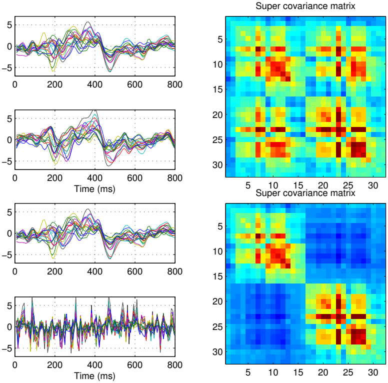
Event Related Potential ?

- Covariance estimation independent of the waveform



Event Related Potential ?

- Covariance estimation independent of the waveform



$$\tilde{\mathbf{X}}_i = \begin{bmatrix} \mathbf{P} \\ \mathbf{X}_i \end{bmatrix}$$

$$\tilde{\mathbf{\Sigma}}_i = \begin{bmatrix} \mathbf{\Sigma}_P & \mathbf{C}_{P,X_i}^T \\ \mathbf{C}_{P,X_i} & \mathbf{\Sigma}_i \end{bmatrix}$$

An indirect measure of correlation with a prototype in the source space :

- *One prototype for each class*
- *Independent of the reprensentation basis (Sensor / Sources)*
- *Prototype can be reused across session, subject or dataset.*

An indirect measure of correlation with a prototype in the source space :

- *One prototype for each class*
- *Independent of the reprensentation basis (Sensor / Sources)*
- *Prototype can be reused across session, subject or dataset.*

Encoding a prior about the shape of the ERP

- *Reduce the number of training data*

Multi-dataset ERP benchmark :

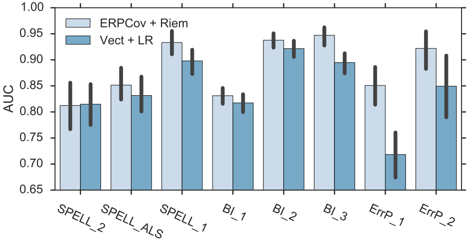
Dataset	Type	# subjects	# channels
BI_1	P300, Brain invaders	93	16 (dry)
BI_2	P300, Brain invaders	24	16
BI_3	P300, Brain invaders	38	32
SPELL_ALS*	P300, Speller, ALS	8	8
SPELL_1*	P300, Speller	8	16
SPELL_2*	P300, Speller	10	8
ErrP_1*	Error P, Speller	26	56
ErrP_2*	Error P, Speller	6	64

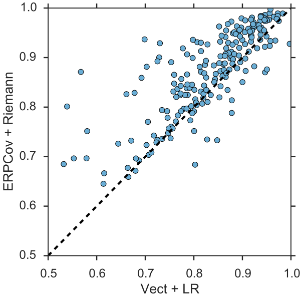
Preprocessing : BP filter, 1-20 Hz. 0.8 second epoch

Classification :

1. Vectorized epoch + l1 Logistic Regression.
2. ERPCov + Tangent Space + Logistic Regression.

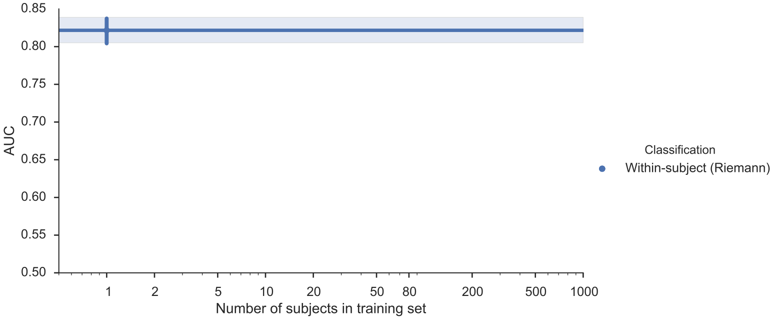
Results


ERPCov + Riem
Vect + LR
AUC
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
SPELL_2
SPELL_ALS
SPELL_1
BI_1
BI_2
BI_3
ErrP_1
ErrP_2

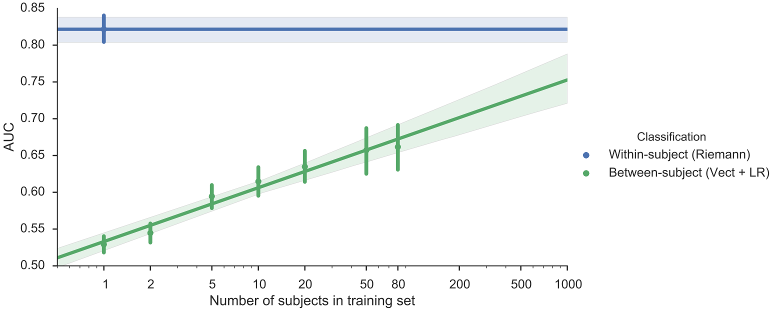

ERPCov + Riemann
Vect + LR
0.5
0.6
0.7
0.8
0.9
1.0

Transfer between subject (dataset BI_1 + BI_2)

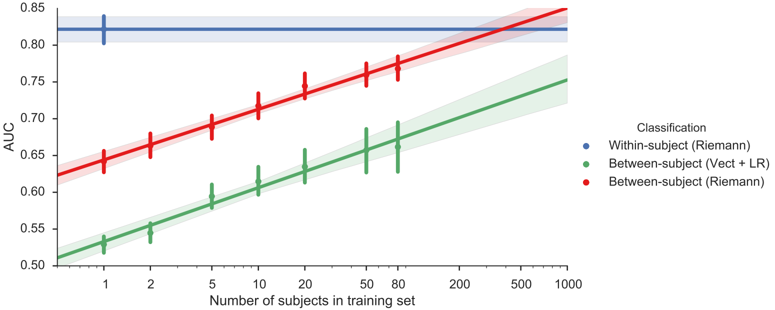
Transfer between subject (dataset BI_1 + BI_2)



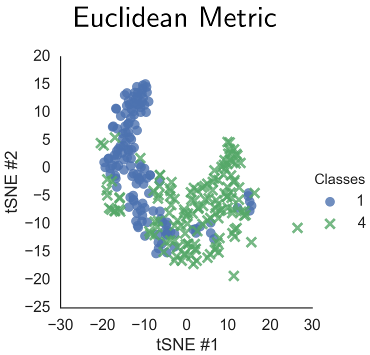
Transfer between subject (dataset BI_1 + BI_2)

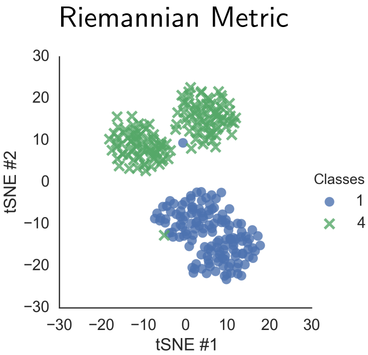


Transfer between subject (dataset BI_1 + BI_2)



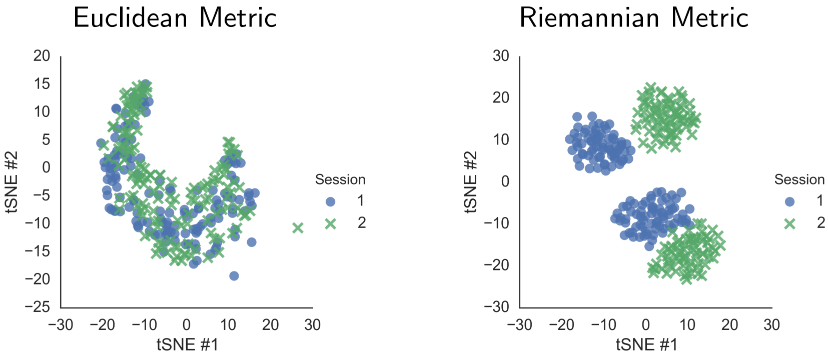
Visualize clusters in a dataset with tSNE[1]:





[1] t-distributed stochastic neighbor embedding

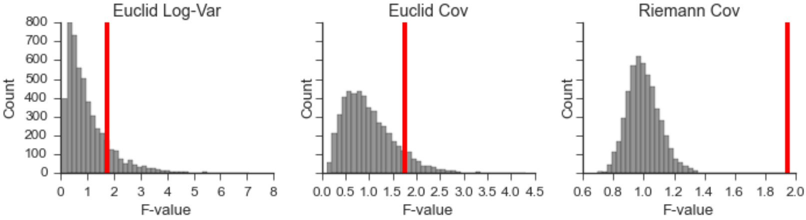
Visualize clusters in a dataset with tSNE[1]:



[1] t-distributed stochastic neighbor embedding

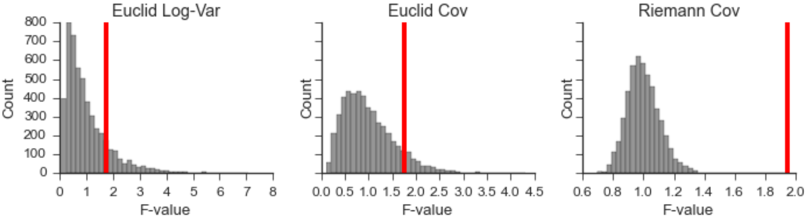
Riemannian Distance based permutation-test :

- *Equivalent to a Manova in the source space*

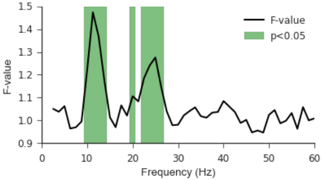


Riemannian Distance based permutation-test :

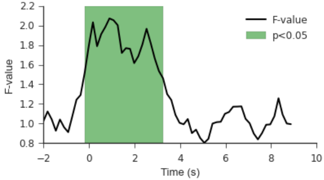
- *Equivalent to a Manova in the source space*



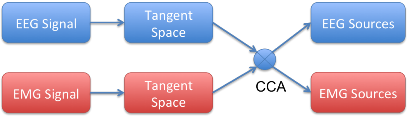
Frequency of interest



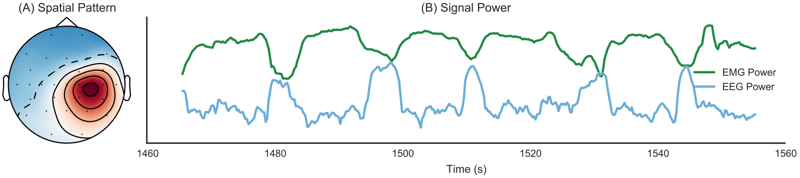
Time window



Find EEG sources that varies jointly with EMG sources[2]



Example : Motor patterns



[2] A. Barachant et. al. *Extraction of motor patterns from joint EEG/EMG recording: A Riemannian Geometry approach.*, BCI Meeting 2016

Covariances matrices as features

- *Efficient representation of M/EEG data*
- *Covariance estimation adapted to each problem*
  - Sample Cov, ERP Cov, Filter Bank, Hankel

Conclusion

Covariances matrices as features

- *Efficient representation of M/EEG data*
- *Covariance estimation adapted to each problem*
    - Sample Cov, ERP Cov, Filter Bank, Hankel

Riemannian Geometry

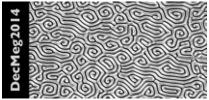
- *A simple way to manipulate covariance matrices*
- *Powerful invariance properties*

Conclusion

Covariance Matrices + Riemannian Geometry : A winning combination !

Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014

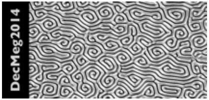


Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014 BCI@NER 2015

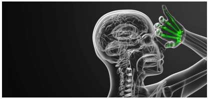
Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014



BCI@NER 2015

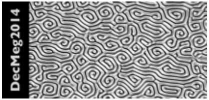
Grasp&Lift 2015



Conclusion

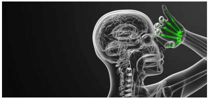
Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014

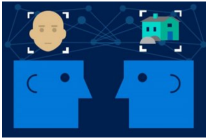


BCI@NER 2015

Grasp&Lift 2015



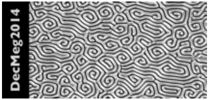
ML Azure 2016



Conclusion

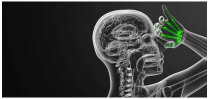
Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014

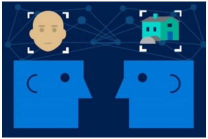


BCI@NER 2015

Grasp&Lift 2015



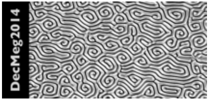
ML Azure 2016



DecMEG 2016

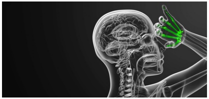
Covariance Matrices + Riemannian Geometry : A winning combination !

DecMEG 2014

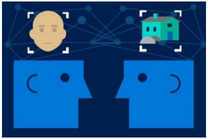


BCI@NER 2015

Grasp&Lift 2015

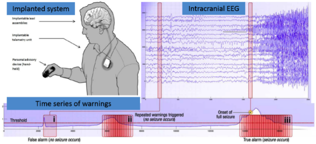


ML Azure 2016



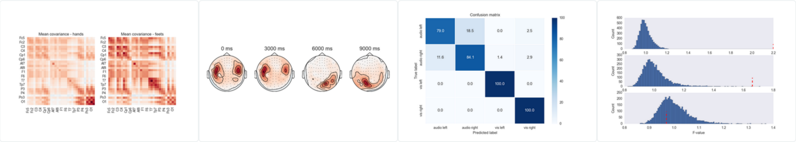
DecMEG 2016

NIH Seizure 2016



pyRiemann 0.2.3 API Gallery Site Page Search

pyRiemann: Biosignals classification with Riemannian Geometry



pyRiemann is a Python machine learning library based on scikit-learn API. It provides a high-level interface for classification and manipulation of multivariate signal through Riemannian Geometry of covariance matrices.

pyRiemann aim at being a generic package for multivariate signal classification but has been designed around applications of biosignal (M/EEG, EMG, etc) classification.

For a brief introduction to the ideas behind the package, you can read the introductory notes. More practical information is on the installation page. You may also want to browse the example gallery to get a sense for what you can do with pyRiemann and then check out the tutorial and API reference to find out how.

To see the code or report a bug, please visit the github repository.

Documentation

- Introduction to pyRiemann
- What's new in the package
- Installing pyRiemann
- Examples Gallery
- API reference

- Covariance estimation
- Classification and clustering
- Artifact detection
- Permutation test
- Channel selection
- Spatial filtering (Multiclass CSP, Xdawn)
- Joint diagonalization (Jade, uWedge, ...)
- ...

Thank you !

Contact

- email : alexandre.barachant@gmail.com
- website : alexandre.barachant.org

Code

- github : https://github.com/alexandrebarachant/
- pyriemann : https://github.com/alexandrebarachant/pyRiemann

Reference

- *"Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review*, Brain-Computer Interfaces, 2017