Geomstats:
A Python Package for Geometry in Machine Learning and Information Geometry

Nina Miolane, Nicolas Guigui[1], Alice Le Brigant, Hadi Zaatiti, Christian Shewmake, Hatem Hajri, Johan Mathe, Benjamin Hou, Yann Thanwerdas, Stefan Heyder, Olivier Peltre, Niklas Koep, Yann Cabanes, Thomas Gerald, Paul Chauchat, Daniel Brooks, Bernhard Kainz, Claire Donnat, Susan Holmes, Xavier Pennec

[1]Université Côte d'Azur, Inria Sophia Antipolis, Epione Research Project, France.

Keywords: **Riemannian Geometry, Python, Open-Source**

Motivation

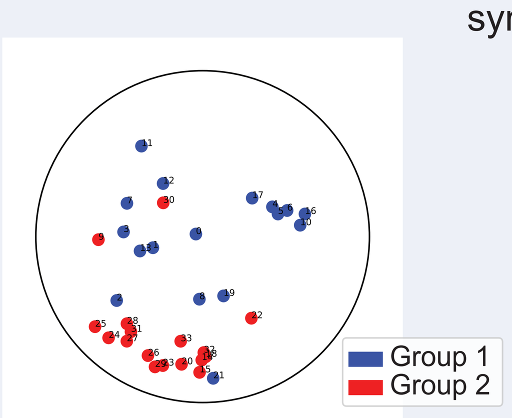
Data on manifolds:
- Geographic data,
- Interaction graphs,
- Correlation matrices,
- Rigid transformations, 3D frames, etc

Parameter Space as manifolds:
- Gaussian family: SPD(n)
- beta distributions: $\mathbb{R}^2_+$
- Hessian manifolds

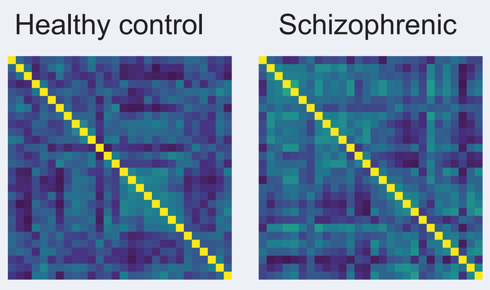
→ Learning that leverages the geometry of the data/parameter space

Cities coordinates
Data on the sphere

Karate Club Network
Data on the hyperbolic space



Brain connectomes
Data on the space of symmetric positive definite matrices



Objectives

- Teach "hands-on" Geometry, Learning and Information Geometry
- Democratize the use of Geometric Learning and Information Geometry in applications
- Support research in Geometric Learning and Information Geometry

Package Description

Geomstats is an open-source Python package for computations and statistics on nonlinear manifolds [Miolane et al., 2020]. geomstats relies on three different back-ends, `numpy`, `pytorch` and `tensorflow` with a generic common API

```
export GEOMSTATS_BACKEND=numpy
import geomstats.backend as gs
```

The package implements tools for over 15 manifolds, each endowed with one or many Riemannian metrics.
- Spaces of constant curvature (Hypersphere , Hyperbolic space with different representations)
- Symmetric Positive Definite matrices
- Lie Groups (Rotations, Rigid-body transformations)
- Linear Subspaces (Grassmannian, Stiefel)
- Curve spaces (Discrete curves, Landmark spaces)
- Probability Distributions (Gaussians, Beta)
- Products of the above

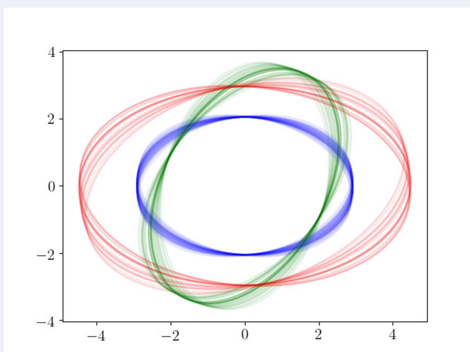


Figure 1: Random samples of SPD matrices from three classes.

Each module exposes methods to handle data that lie on the corresponding manifold, e.g. to sample random points, to check that a point indeed belongs to the manifold, to project vectors to tangent vectors to a point.

```
from geomstats.geometry.hypersphere import
  Hypersphere

sphere = Hypersphere(dim=2)
points = sphere.random_uniform(10)

print(sphere.belongs(points))
```

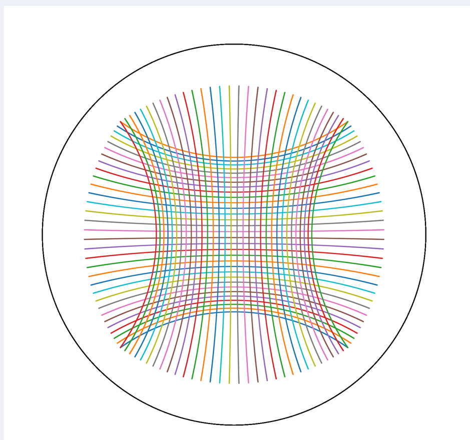


Figure 2: Geodesic grid on Poincaré disk.

Riemannian Metrics expose methods to compute inner-products, geodesics, exponential, logarithm and parallel transport maps. Numerical methods are used when no closed-form solution exist

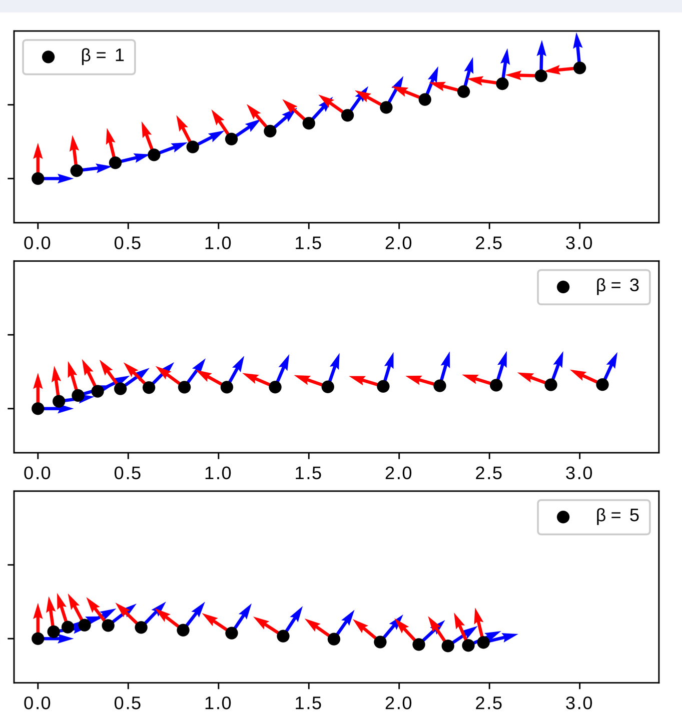


Figure 3: Geodesics in $SE(2)$ with different left-invariant metrics. β is an anisotropy coefficient of the metric. Automatic differentiation and gradient descent allow to compute logarithms by geodesic shooting

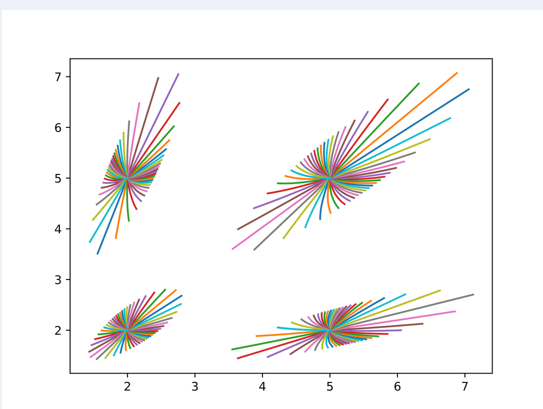


Figure 4: Geodesic balls on the manifold of Beta distributions with the Fisher metric

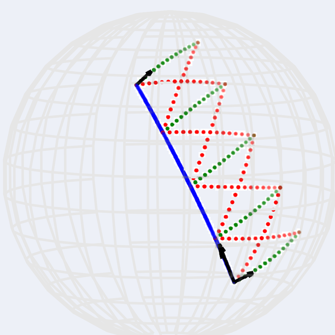


Figure 5: Schild's ladder: a numerical scheme to approach parallel transport [Guigui and Pennec, 2020]

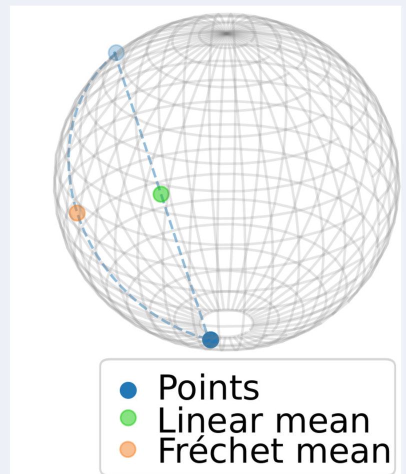
Fréchet Mean

- Generalized definition of the mean: $\bar{x} = \underset{x \in M}{\text{argmin}} \sum_{i=1}^n \text{dist}_M(x, x_i)^2$

→ Fréchet mean belongs to the manifold.

```
from geomstats.learning.frechet_mean import FrechetMean

estimator = FrechetMean(metric=sphere.metric)
estimator.fit(points)
frechet_mean = estimator.estimate_
```



From Logistic Regression to tangent Logistic Regression

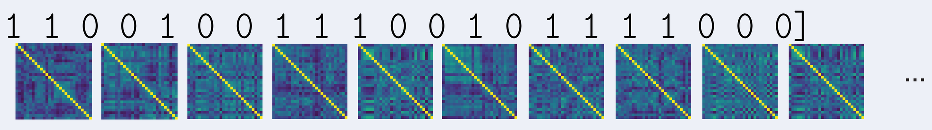
Use scikit-learn on the tangent space at the Fréchet mean

```
from geomstats.learning.preprocessing import ToTangentSpace
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline

data, patient_ids, labels = data_utils.load_connectomes()
X_train, X_test, y_train, y_test = train_test_split(data, labels, random_state=0)

metric_aff = SPDMetricAffine(n=28)
pipeline = make_pipeline(ToTangentSpace(geometry=metric_aff), LogisticRegression(C=2))
pipeline.fit(X_train, y_train)
pipeline.predict(X_test)
```

> [0 1 1 0 0 1 0 0 1 1 1 0 0 1 0 1 1 1 1 0 0 0]



From k-means to Riemannian k-means

Use Geomstats for unsupervised learning. Below an example of clustering on data from community network, embedded in the hyperbolic space [Gerald et al., 2020].

```
from geomstats.learning.kmeans import RiemannianKMeans

poincare_ball = PoincareBall(2)
karate_graph = data_utils.load_karate_graph()

hyperbolic_embedding = HyperbolicEmbedding()
embeddings = hyperbolic_embedding.embed(karate_graph)

kmeans = RiemannianKMeans(
    riemannian_metric=poincare_ball.metric,n_clusters=2,
    mean_method='frechet-poincare-ball')

centroids = kmeans.fit(X=embeddings, max_iter=100)
labels = kmeans.predict(X=embeddings)
```

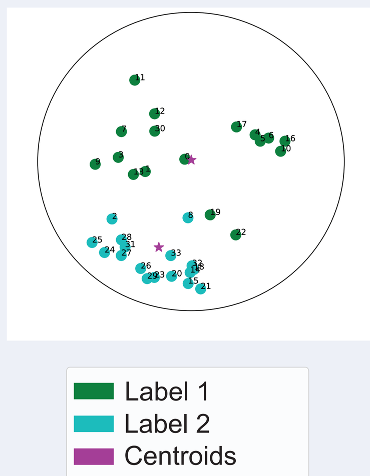


Figure 6: Riemannian k-means on the graph nodes after embedding in the Poincaré ball

Beta distributions to classify histograms

We use maximum likelihood to fit beta distributions to histograms of cortical thickness maps [Brigant et al., 2020]. The manifold of beta distributions endowed with the Fisher-Rao metric has negative curvature. Geodesic distances can be computed numerically to apply k-nearest neighbor or k-means algorithms.

```
from sklearn.neighbors import
  KNeighborsClassifier
from geomstats.geometry.beta_distributions
  import BetaDistributions

beta = BetaDistributions()
embeddings = beta.maximum_likelihood_fit(samples
  )

rnn = KNeighborsClassifier(n_neighbors=9, metric
  =beta.metric.dist)
rnn.fit(embeddings)
```

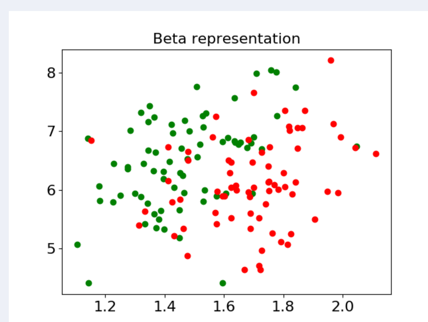


Figure 7: Embedding of the histograms in the 2d-manifold of beta distributions

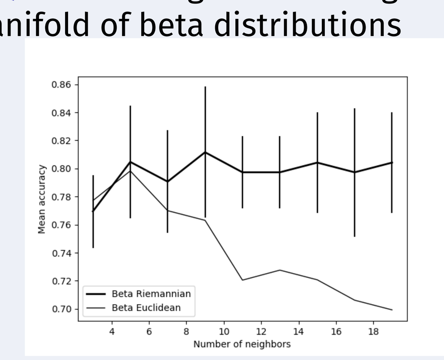


Figure 8: Mean accuracy of KNN classification on the cortical thickness data over 5-fold cross validation

References

Brigant, A. L., Guigui, N., Rebbah, S., and Puechmorel, S. (2020).
Classifying histograms of medical data using information geometry of beta distributions.

Gerald, T., Zaatiti, H., Hajri, H., Baskiotis, N., and Schwander, O. (2020).
From Node Embedding To Community Embedding : A Hyperbolic Approach.
arXiv: 1907.01662.

Guigui, N. and Pennec, X. (2020).
Numerical Accuracy of Ladder Schemes for Parallel Transport on Manifolds.

Miolane, N., Brigant, A. L., Mathe, J., Hou, B., Guigui, N., Thanwerdas, Y., Heyder, S., Peltre, O., Koep, N., Zaatiti, H., Hajri, H., Cabanes, Y., Gerald, T., Chauchat, P., Shewmake, C., Kainz, B., Donnat, C., Holmes, S., and Pennec, X. (2020).
Geomstats: A Python Package for Riemannian Geometry in Machine Learning.

http://geomstats.ai