Geomstats: a Python package for Riemannian Geometry and Geometric Statistics

Luís F. Pereira
Geometric Intelligence Lab @ UCSB

Geometric Intelligence workshop @ UNAM
April 1, 2025

Geomstats[1]

Computations and **statistics** on **manifolds** with geometric structures

- `Python` (powered by `numpy`, `autograd`, or `pytorch`)
- open-source (MIT license)
- object-oriented
- `sklearn`-inspired API
- thoroughly tested with (extended) `pytest`

[1] Guigui, Miolane, and Pennec, "Introduction to Riemannian Geometry and Geometric Statistics", 2022.

Useful links

1. `geomstats` documentation: https://geomstats.github.io/
2. Basic demo (with installation instructions): https://github.com/luisfpereira/giw25
3. Additional resources:
   3.1 `polpo`: https://geometric-intelligence.github.io/polpo/ (in particular, how-tos)
   3.2 `pyvista`: https://docs.pyvista.org/ (a proof it can be used to create impressive 3D figures: https://www.jeanfeydy.com/Talks/ShapeSeminar_2024/ShapeSeminar_2024.pdf)

Motivation: Riemannian geometry

Let $A, B \in \mathcal{X}$, where $\mathcal{X}$ is some non-linear space:

1. how to compute **distances** between A and B?
2. how to **interpolate** between A and B?
3. how to **extrapolate**?

Why these questions?

1. notion of distance $\implies$ can do statistics
2. interpolation $\implies$ can meaningfully move between points
3. extrapolation $\implies$ strong predicting ability

Motivation: why Geomstats?

A platform implementing **consistently** and **flexibly** different Riemannian manifolds (and more).

Consequences:

- code reuse
- (hopefully) less bugs (easier testing)
- easy experimentation
- productivity gains (after mastering structure)
- sense of community

Geomstats: main use case

1. Geometry: Instantiate a manifold

```python
from geomstats.geometry.hypersphere import Hypersphere
sphere = Hypersphere(dim=2)
```

2. Statistics/Learning: Run estimation or learning

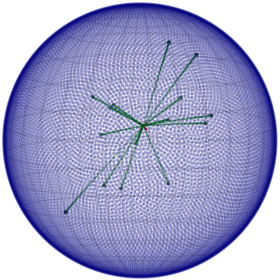
```python
from geomstats.learning.frechet_mean import FrechetMean
estimator = FrechetMean(space)
estimator.fit(points)
```

Statistics on manifolds: Fréchet mean

```
estimator = FrechetMean(space)
estimator.fit(X)

mean = estimator.estimate_
```

$$\bar{x} = \underset{x \in M}{\operatorname{argmin}} \sum_{i=1}^{n} d\left(x_i, x\right)^2$$

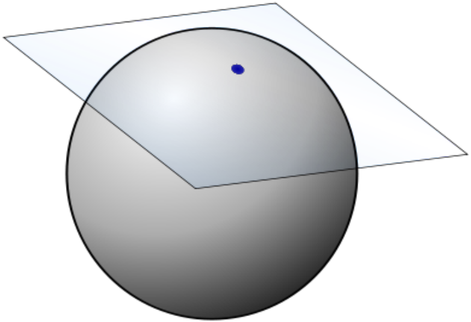


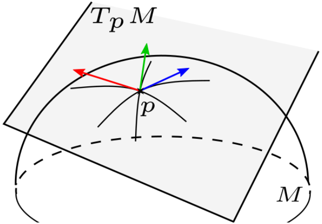
```
import geomstats.backend as gs

while condition:
    grad = -gs.sum(space.metric.log(X, mean)) / n_samples
    mean = space.metric.exp(step_size*grad, mean)
```

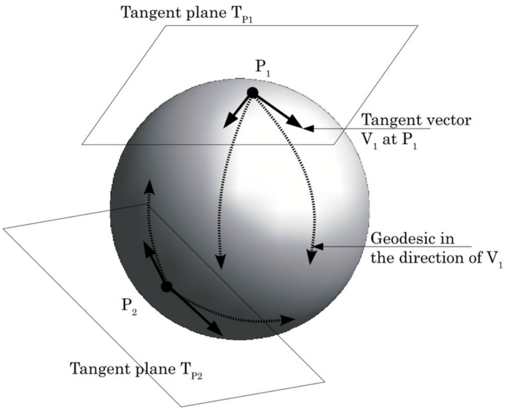
Tangent space

- Vector space associated to a point in a manifold

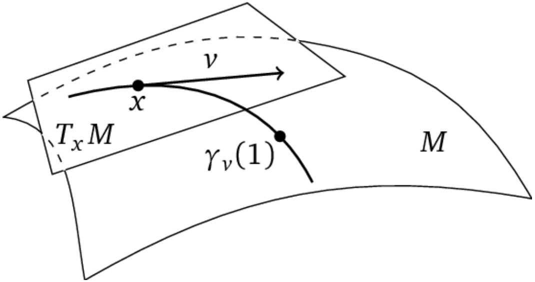




Generalizing straight lines: geodesics



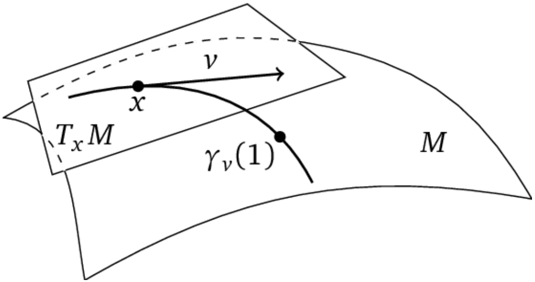
Exponential and logarithmic maps



$\mathrm{Exp}_x : T_x M \to M$ $\qquad y = \gamma_v(1) = \mathrm{Exp}_x(v)$ $\qquad$ *addition*

$\mathrm{Log}_x : M \to T_x M$ $\qquad v = \mathrm{Log}_x(y)$ $\qquad$ *subtraction*

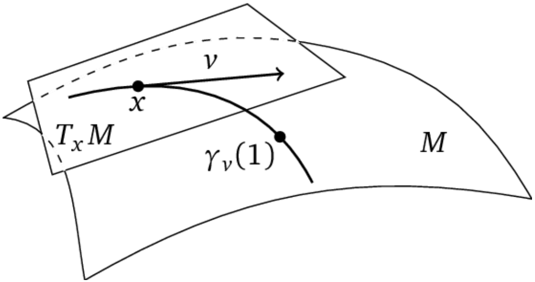
Exponential and logarithmic maps



$\mathrm{Exp}_x : T_xM \to M$ $\qquad y = \gamma_v(1) = \mathrm{Exp}_x(v)$ $\qquad$ *addition*

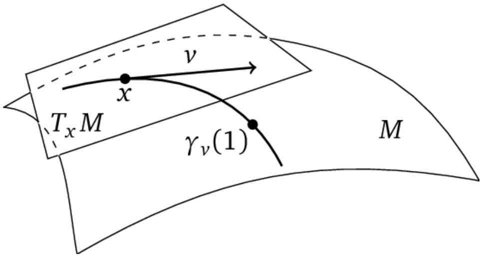
$\mathrm{Log}_x : M \to T_xM$ $\qquad v = \mathrm{Log}_x(y)$ $\qquad$ *subtraction*

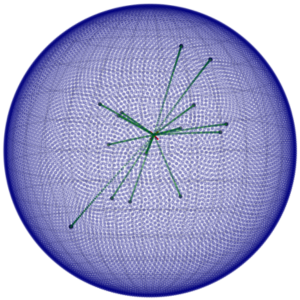
Exponential and logarithmic maps



$\mathrm{Exp}_x : T_xM \to M$	$y = \gamma_v(1) = \mathrm{Exp}_x(v)$	*addition*
$\mathrm{Log}_x : M \to T_xM$	$v = \mathrm{Log}_x(y)$	*subtraction*

Recap: Fréchet mean



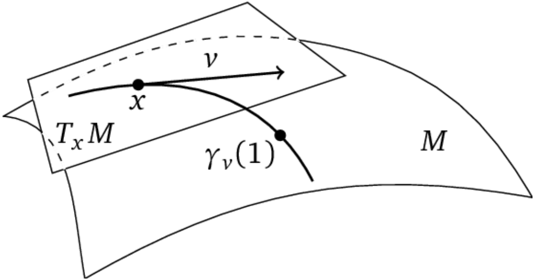


$$\mathrm{Exp}_x : T_x M \to M$$
$$\mathrm{Log}_x : M \to T_x M$$

$$\bar{x} = \underset{x \in M}{\operatorname{argmin}} \sum_{i=1}^{n} d\left(x_i, x\right)^2$$

```
while condition:
    grad = -gs.sum(space.metric.log(X, mean)) / n_samples
    mean = space.metric.exp(step_size*grad, mean)
```

Length, distance, and inner product



$$d(x_0, x_1) = \mathrm{Len}(\gamma(x_0, x_1))$$

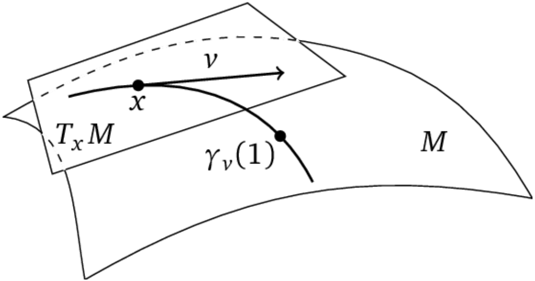
$g_x(\cdot, \cdot)$ on $T_x M$

$d(x, y)^2 = g_x(v, v)$, where $v = \mathrm{Log}_x(y)$

```
v = space.metric.log(y, x)
squared_dist = space.metric.inner_product(v, v, x)

# or
squared_dist = space.metric.squared_dist(x, y)
```

Length, distance, and inner product



$$d(x_0, x_1) = \mathrm{Len}(\gamma(x_0, x_1))$$

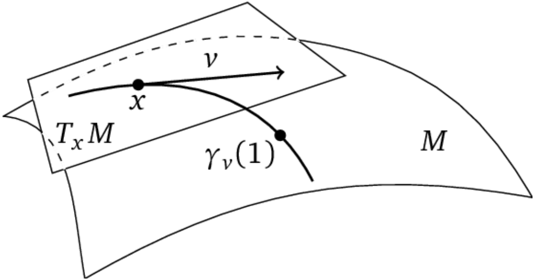
$g_x(\cdot,\cdot)$ on $T_x M$

$d(x, y)^2 = g_x(v, v)$, where $v = \mathrm{Log}_x(y)$

```
v = space.metric.log(y, x)
squared_dist = space.metric.inner_product(v, v, x)

# or
squared_dist = space.metric.squared_dist(x, y)
```

Length, distance, and inner product



$$d(x_0, x_1) = \mathrm{Len}(\gamma(x_0, x_1))$$

$$g_x(\cdot, \cdot) \text{ on } T_x M$$

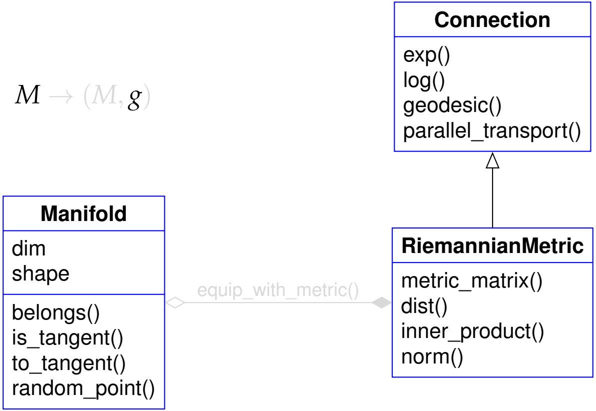
$$d(x, y)^2 = g_x(v, v), \text{where } v = \mathrm{Log}_x(y)$$

```
v = space.metric.log(y, x)
squared_dist = space.metric.inner_product(v, v, x)

# or
squared_dist = space.metric.squared_dist(x, y)
```

Equip with metric

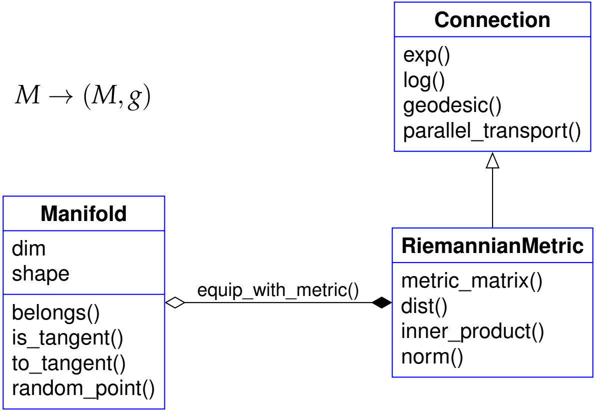
$$M \to (M, g)$$




```
my_manifold = MyManifold(equip=False)
my_manifold.equip_with_metric(MyRiemannianMetric)
```

Equip with metric

$$M \to (M, g)$$




```
my_manifold = MyManifold(equip=False)
my_manifold.equip_with_metric(MyRiemannianMetric)
```

Ways of defining a Riemannian manifolds

"Vanilla" ways:

- vector space:

$$\text{e.g. } \mathrm{Sym}(n) = \left\{X \in \mathrm{Mat}(n) \mid X^\top = X\right\}$$

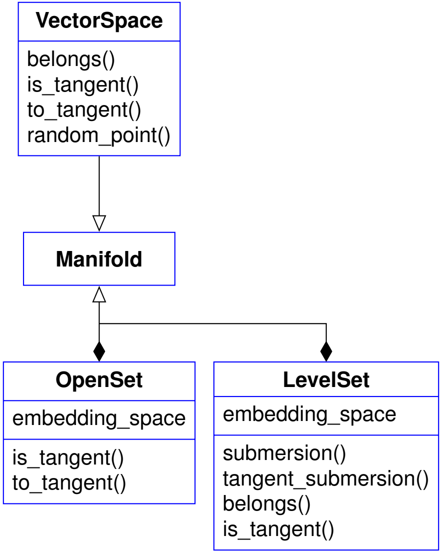
- open set (of a vector space):

$$\text{e.g. } \mathrm{Sym}^+(n) = \{\Sigma \in \mathrm{Sym}(n) \mid \Sigma > 0\}$$

- level set of a smooth submersion:

$$\text{e.g. } \mathrm{Cor}^+(n) = \left\{\Sigma \in \mathrm{Sym}^+(n) \mid \mathrm{Diag}(\Sigma) = I_n\right\}$$

Implementing manifolds



Ways of defining a Riemannian manifolds (cont'd)

"Vanilla" ways:

- vector space
- open set (of a vector space)
- level set of a smooth submersion

Taking advantage of existing structures:

- product manifold
- pullback using diffeomorphisms
- quotient space (action of a Lie group)

Ways of defining a Riemannian manifolds (cont'd)

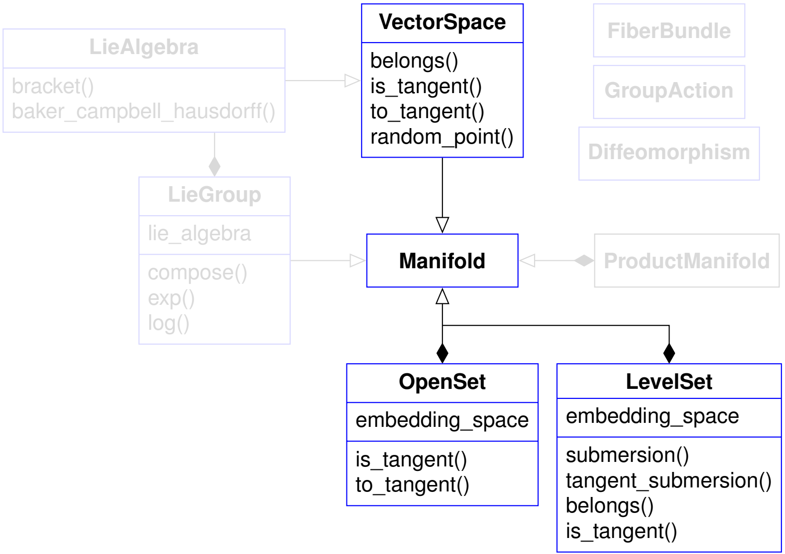
"Vanilla" ways:

- vector space
- open set (of a vector space)
- level set of a smooth submersion

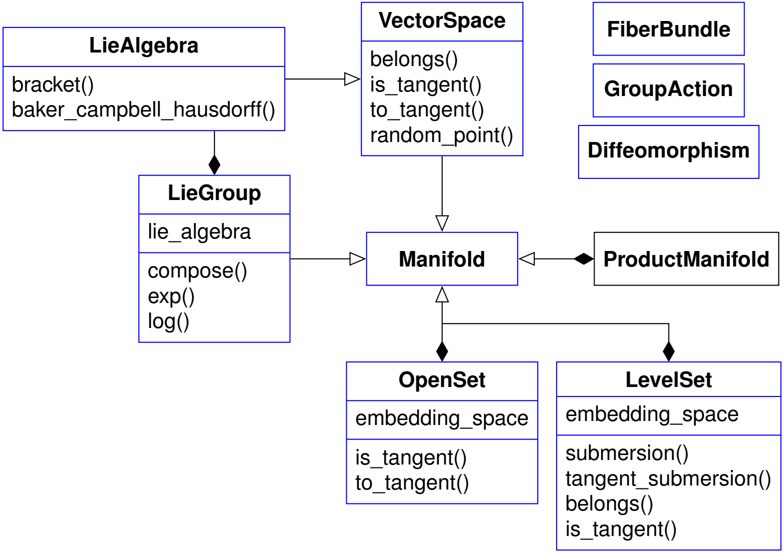
Taking advantage of existing structures:

- product manifold
- pullback using diffeomorphisms
- quotient space (action of a Lie group)

Implementing manifolds (cont'd)



Implementing manifolds (cont'd)



Ways of defining a Riemannian metrics

Mirror manifold structure.

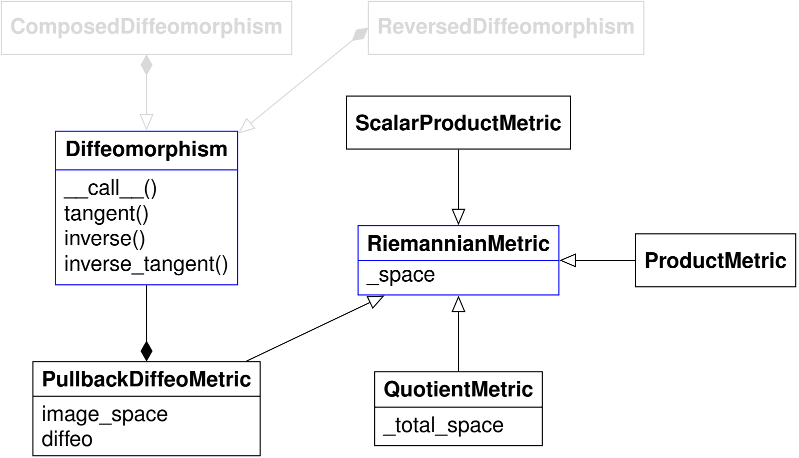
"Vanilla" ways: **closed-forms** usually available

- vector space
- open set (of a vector space)
- level set of a smooth submersion

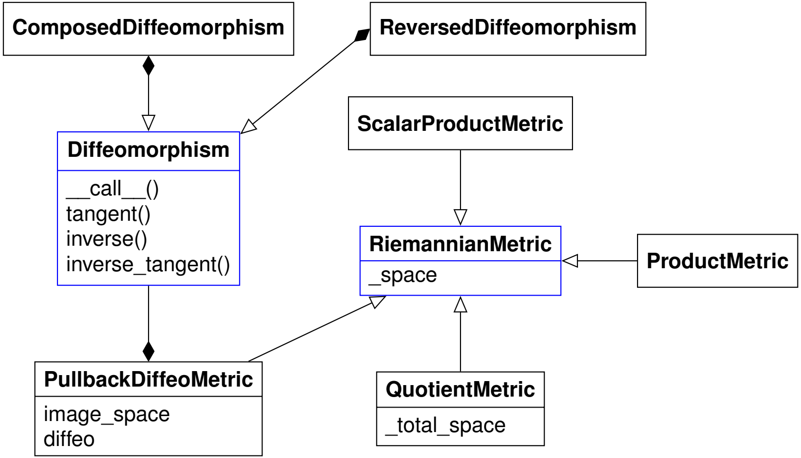
Taking advantage of existing structures:

- product manifold **-> product metric**
- pullback using diffeomorphisms **-> pullback metric**
- quotient space (action of a Lie group) **-> quotient metric**
- **scaled metric**

Implementing metrics

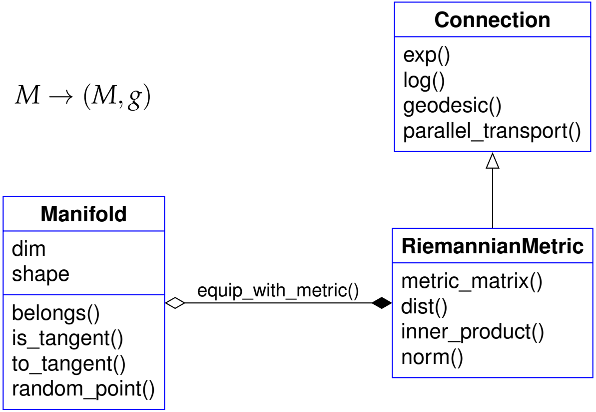


Implementing metrics



Equip with metric (recap)

$$M \to (M, g)$$




```
my_manifold = MyManifold(equip=False)
my_manifold.equip_with_metric(MyRiemannianMetric)
```

Reflection on design choices

- a smooth manifold exists without a metric: M
- multiple metrics can be put on a smooth manifold: (M, g_1), (M, g_2)
- a metric only makes sense if endowing a manifold: g (?)
- a metric can endow multiple manifolds (e.g. Frobenius metric)
- often manifold attributes need to be accessed within the metric: `metric._space`

Design is circular!

Reflection on design choices

- a smooth manifold exists without a metric: M
- multiple metrics can be put on a smooth manifold: (M, g_1), (M, g_2)
- a metric only makes sense if endowing a manifold: g (?)
- a metric can endow multiple manifolds (e.g. Frobenius metric)
- often manifold attributes need to be accessed within the metric: `metric._space`

Design is circular!

Example: metrics on $\mathrm{Cor}^+(n)$[2,3]

Riemannian Metric	Curvature
Quotient-Affine	$[\frac{-1}{2}, +\infty)$
Quotient Lie-Cholesky	?
Poly-Hyperbolic-Cholesky	≤ 0
Euclidean-Cholesky	0
log-Euclidean-Cholesky	0
Off-Log	0
Log-Scaled	0
Spherical Product	≥ 0

[2] Thanwerdas, “Riemannian and stratified geometries of covariance and correlation matrices”, 2022.
[3] Bisson, 2025. *To appear.*

Example: power-affine metric on $\mathrm{Sym}^+(n)$

$$\mathrm{Sym}^+(n) = \{\Sigma \in \mathrm{Sym}(n) \mid x^\top \Sigma x > 0 \; \forall x \in \mathbb{R}^n \backslash \{0\}\}$$

(an `OpenSet`)

$$g_\Sigma(V, W) = \mathrm{tr}\left(\Sigma^{-1} V \Sigma^{-1} W\right)$$

(a `RiemannianMetric`)

$$\mathrm{pow}_p : \Sigma \in \mathrm{Sym}^+(n) \longmapsto \Sigma^p \in \mathrm{Sym}^+(n)$$

(a `Diffeomorphism`)

$$g_\Sigma^{(p)}(X, X) = \frac{1}{p^2} g_{\Sigma^p}\left(d_\Sigma \mathrm{pow}_p(X), d_\Sigma \mathrm{pow}_p(X)\right)$$

(a `PullbackDiffeoMetric` of a `ScalarProductMetric`)

Example: power-affine metric on $\mathrm{Sym}^+(n)$

$$\mathrm{Sym}^+(n) = \{\Sigma \in \mathrm{Sym}(n) \mid x^\top \Sigma x > 0 \ \forall x \in \mathbb{R}^n \backslash \{0\}\}$$

(an `OpenSet`)

$$g_\Sigma(V, W) = \mathrm{tr}\left(\Sigma^{-1} V \Sigma^{-1} W\right)$$

(a `RiemannianMetric`)

$$\mathrm{pow}_p : \Sigma \in \mathrm{Sym}^+(n) \longmapsto \Sigma^p \in \mathrm{Sym}^+(n)$$

(a `Diffeomorphism`)

$$g_\Sigma^{(p)}(X, X) = \frac{1}{p^2} g_{\Sigma^p}\left(d_\Sigma \mathrm{pow}_p(X), d_\Sigma \mathrm{pow}_p(X)\right)$$

(a `PullbackDiffeoMetric` of a `ScalarProductMetric`)

Example: power-affine metric on $\mathrm{Sym}^+(n)$

$$\mathrm{Sym}^+(n) = \{\Sigma \in \mathrm{Sym}(n) \mid x^\top \Sigma x > 0 \; \forall x \in \mathbb{R}^n \backslash \{0\}\}$$

(an `OpenSet`)

$$g_\Sigma(V, W) = \mathrm{tr}\left(\Sigma^{-1} V \Sigma^{-1} W\right)$$

(a `RiemannianMetric`)

$$\mathrm{pow}_p : \Sigma \in \mathrm{Sym}^+(n) \longmapsto \Sigma^p \in \mathrm{Sym}^+(n)$$

(a `Diffeomorphism`)

$$g_\Sigma^{(p)}(X, X) = \frac{1}{p^2} g_{\Sigma^p}\left(d_\Sigma \,\mathrm{pow}_p(X), d_\Sigma \,\mathrm{pow}_p(X)\right)$$

(a `PullbackDiffeoMetric` of a `ScalarProductMetric`)

Example: poly-hyperbolic-Cholesky metrics on $\mathrm{Cor}^+(n)$

$$\text{Cor}^+(n) \xrightarrow{\text{Chol}} \mathcal{L}(n) \xrightarrow{\psi} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\varphi^{\mathbb{SH}}} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

$$\mathrm{Cor}^+(n) = \left\{ \Sigma \in \mathrm{Sym}^+(n) \mid \mathrm{Diag}(\Sigma) = I_n \right\}$$

(a `LevelSet`)

$$\mathrm{Chol}_{|\mathrm{Cor}^+(n)} : \mathrm{Cor}^+(n) \longrightarrow \mathcal{L}(n)$$

(a `Diffeomorphism`)

$\mathcal{L}(n)$: set of positive lower triangular matrices with unit normed rows

(a `DiffeomorphicManifold`)

$$\psi : \mathcal{L} \longrightarrow \mathrm{HS}^1 \times \cdots \times \mathrm{HS}^{n-1}$$

(a `Diffeomorphism` to a `ProductManifold`)

$$\mathrm{HS}^k = \left\{ x \in \mathbb{R}^{k+1} : \|x\| = 1 \text{ and } x_0 > 0 \right\}$$

(an `OpenSet` of a `LevelSet`)

Example: poly-hyperbolic-Cholesky metrics on $\mathrm{Cor}^+(n)$

$$\mathrm{Cor}^+(n) \xrightarrow{\text{Chol}} \mathcal{L}(n) \xrightarrow{\psi} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\varphi^{\mathbb{SH}}} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

$$\mathrm{Cor}^+(n) = \left\{\Sigma \in \mathrm{Sym}^+(n) \mid \mathrm{Diag}(\Sigma) = I_n\right\}$$

(a `LevelSet`)

$$\mathrm{Chol}_{|\mathrm{Cor}^+(n)} : \mathrm{Cor}^+(n) \longrightarrow \mathcal{L}(n)$$

(a `Diffeomorphism`)

$\mathcal{L}(n)$: set of positive lower triangular matrices with unit normed rows

(a `DiffeomorphicManifold`)

$$\psi : \mathcal{L} \longrightarrow \mathrm{HS}^1 \times \cdots \times \mathrm{HS}^{n-1}$$

(a `Diffeomorphism` to a `ProductManifold`)

$$\mathrm{HS}^k = \left\{x \in \mathbb{R}^{k+1} : \|x\| = 1 \text{ and } x_0 > 0\right\}$$

(an `OpenSet` of a `LevelSet`)

Example: poly-hyperbolic-Cholesky metrics on $\mathrm{Cor}^+(n)$

$$\text{Cor}^+(n) \xrightarrow{\text{Chol}} \mathcal{L}(n) \xrightarrow{\psi} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\varphi^{\mathbb{SH}}} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

$$\mathrm{Cor}^+(n) = \left\{ \Sigma \in \mathrm{Sym}^+(n) \mid \mathrm{Diag}(\Sigma) = I_n \right\}$$

(a `LevelSet`)

$$\mathrm{Chol}_{|\mathrm{Cor}^+(n)} : \mathrm{Cor}^+(n) \longrightarrow \mathcal{L}(n)$$

(a `Diffeomorphism`)

$$\psi : \mathcal{L} \longrightarrow \mathrm{HS}^1 \times \cdots \times \mathrm{HS}^{n-1}$$

(a `Diffeomorphism` to a `ProductManifold`)

$\mathcal{L}(n)$: set of positive lower triangular matrices with unit normed rows

(a `DiffeomorphicManifold`)

$$\mathrm{HS}^k = \left\{ x \in \mathbb{R}^{k+1} : \|x\| = 1 \text{ and } x_0 > 0 \right\}$$

(an `OpenSet` of a `LevelSet`)

Example: poly-hyperbolic-Cholesky metrics on $\mathrm{Cor}^+(n)$ (cont'd)

$$\text{Cor}^+(n) \xrightarrow{\text{Chol}} \mathcal{L}(n) \xrightarrow{\psi} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\varphi^{\mathbb{SH}}} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

$$\varphi : H\mathbb{S}^k \longrightarrow \mathbb{H}^k, \varphi^{\mathbb{SH}} = (\varphi^1, \dots, \varphi^{n-1})$$

(a `Diffeomorphism`)

$$\mathbb{H}^k = \left\{ x \in \mathbb{R}^{k+1} \mid -x_0^2 + \sum_{i=2}^{k+1} x_i^2 = -1 \right\}$$

(a `LevelSet`)

Multiple `Diffeomorphism` or `ComposedDiffeomorphism`

Example: poly-hyperbolic-Cholesky metrics on $\mathrm{Cor}^+(n)$ (cont'd)

$$\text{Cor}^+(n) \xrightarrow{\quad\text{Chol}\quad} \mathcal{L}(n) \xrightarrow{\quad\psi\quad} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\quad\varphi^{\mathbb{SH}}\quad} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

$$\varphi : H\mathbb{S}^k \longrightarrow \mathbb{H}^k, \varphi^{\mathbb{SH}} = (\varphi^1, \ldots, \varphi^{n-1})$$

(a `Diffeomorphism`)

$$\mathbb{H}^k = \left\{ x \in \mathbb{R}^{k+1} \mid -x_0^2 + \sum_{i=2}^{k+1} x_i^2 = -1 \right\}$$

(a `LevelSet`)

Multiple `Diffeomorphism` or `ComposedDiffeomorphism`

Example: `Diffeomorphism` or `ComposedDiffeomorphism`

$$\mathrm{Cor}^{+}(n) \xrightarrow{\text{Chol}} \mathcal{L}(n) \xrightarrow{\psi} \Pi_{k=1}^{n-1} H\mathbb{S}^{k} \xrightarrow{\varphi^{\mathbb{SH}}} \Pi_{k=1}^{n-1} \mathbb{H}^{k}$$

```python
class PolyHyperbolicCholeskyMetric(PullbackDiffeoMetric):
    def __init__(self, space):
        n = space.n
        diffeo = CholeskyMap()
        image_space = UnitNormedRowsPLTMatrices(n)

        super().__init__(
            space=space, diffeo=diffeo, image_space=image_space
        )
```

Example: `Diffeomorphism` or `ComposedDiffeomorphism`

$$\mathrm{Cor}^{+}(n) \xrightarrow{\quad \mathrm{Chol} \quad} \mathcal{L}(n) \xrightarrow{\quad \psi \quad} \Pi_{k=1}^{n-1} H\mathbb{S}^k \xrightarrow{\quad \varphi^{\mathbb{SH}} \quad} \Pi_{k=1}^{n-1} \mathbb{H}^k$$

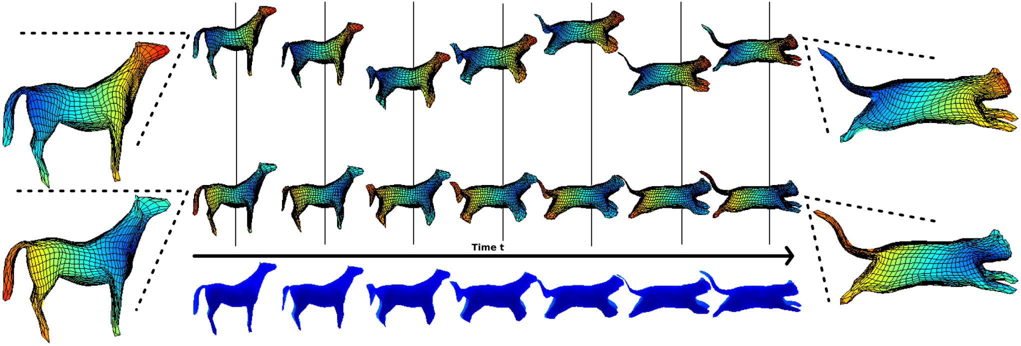
```python
class PolyHyperbolicCholeskyMetric(PullbackDiffeoMetric):
    def __init__(self, space):
        n = space.n
        diffeos = [CholeskyMap(), UnitNormedRowsPLTDiffeo(n)]

        if n == 2:
            diffeos.append(OpenHemisphereToHyperboloidDiffeo())
            image_space = Hyperboloid(dim=1)
        else:
            image_space = OpenHemispheresProduct(n=n)

        diffeo = ComposedDiffeo(diffeos)

        super().__init__(space=space, diffeo=diffeo, image_space=image_space)
```

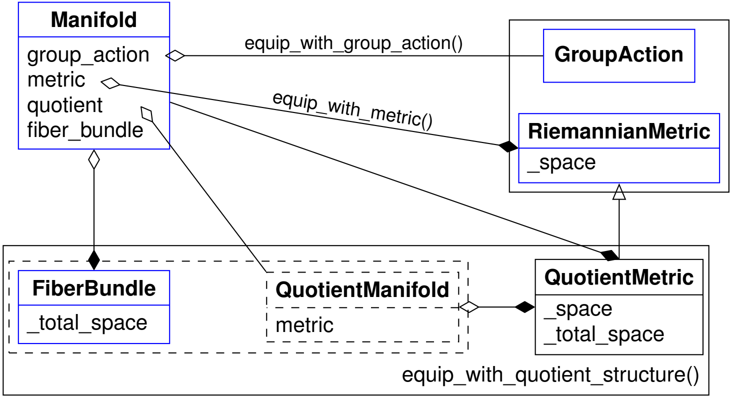
Quotient spaces: a visual perspective



4

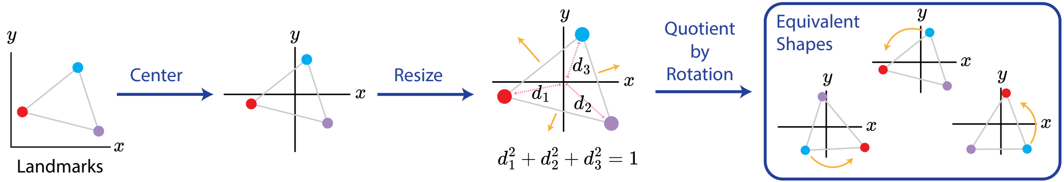
[4] Tumpach, "Some aspects of infinite-dimensional Geometry", 2022.

Equip with quotient structure




```
total_space = MyManifold(equip=False)
total_space.equip_with_metric(MyRiemannianMetric)
total_space.equip_with_group_action(my_group_action)
total_space.equip_with_quotient_structure()
```

Example: Kendall shape space



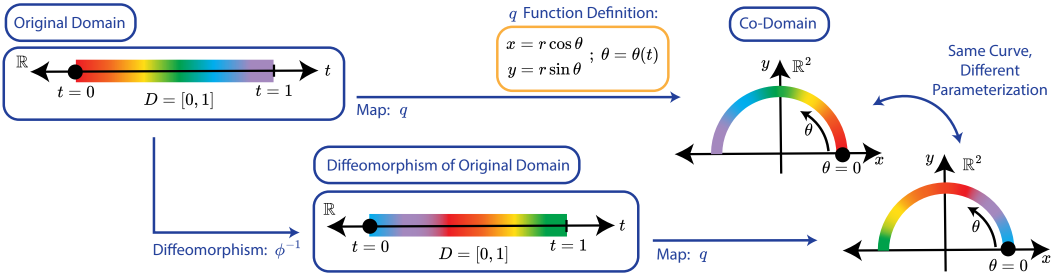
[5]

Pre-shape space and metric
(a `LevelSet` and a `PullbackDiffeoMetric`)

Rotations, pre-shape bundle, and Kendall metric
(a `GroupAction`, a `FiberBundle`, and a `QuotientMetric`)

[5] Pereira et al., Learning from landmarks, curves, surfaces, and shapes in Geomstats, 2024.

Example: elastic metrics on curves



[6]

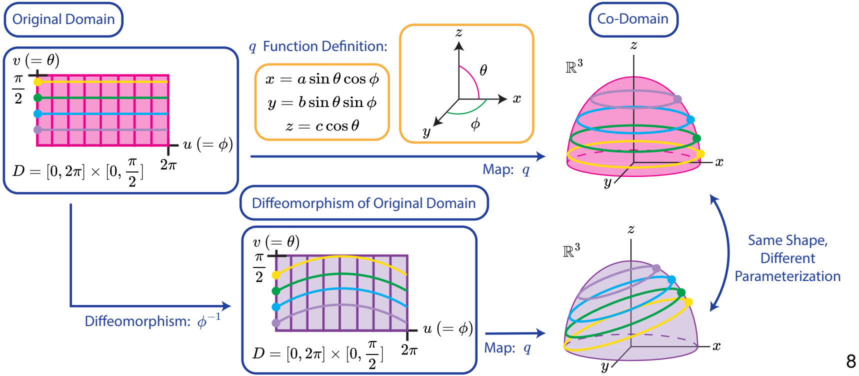
Isometry between

$$\left(\mathrm{Imm}_0\left([0,1],\mathbb{R}^d\right), G^{a,b}\right) \text{ and } \left(C^\infty\left([0,1],\mathbb{R}^d\backslash\{0\}\right), 4b^2 G^{L^2_\lambda}\right)^7$$

Main challenge: alignment wrt reparameterizations (`AlignerAlgoritm`).

[6] Pereira et al., Learning from landmarks, curves, surfaces, and shapes in Geomstats, 2024.
[7] Bauer et al., "Elastic Metrics on Spaces of Euclidean Curves", 2024.

Example: elastic metrics on surfaces



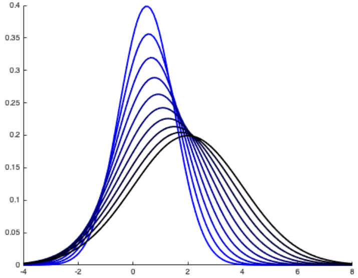
Challenges:

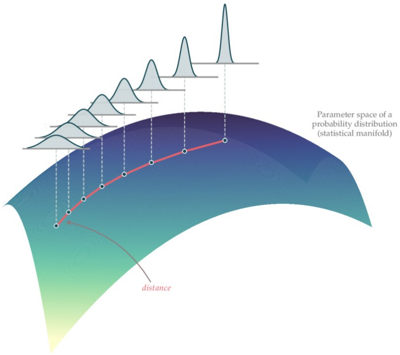
- no-closed form solutions for exp and log
- alignment wrt reparameterizations (varifolds and `pykeops`)

[8] Pereira et al., Learning from landmarks, curves, surfaces, and shapes in Geomstats, 2024.

Information geometry[9]

$$\mathcal{P} = \left\{ P_\theta, \theta \in \Theta \subseteq \mathbb{R}^d \right\}$$





How to compute distances between probability distributions?

[9] Brigant et al., Parametric information geometry with the package Geomstats, 2022.

Information geometry (cont'd)

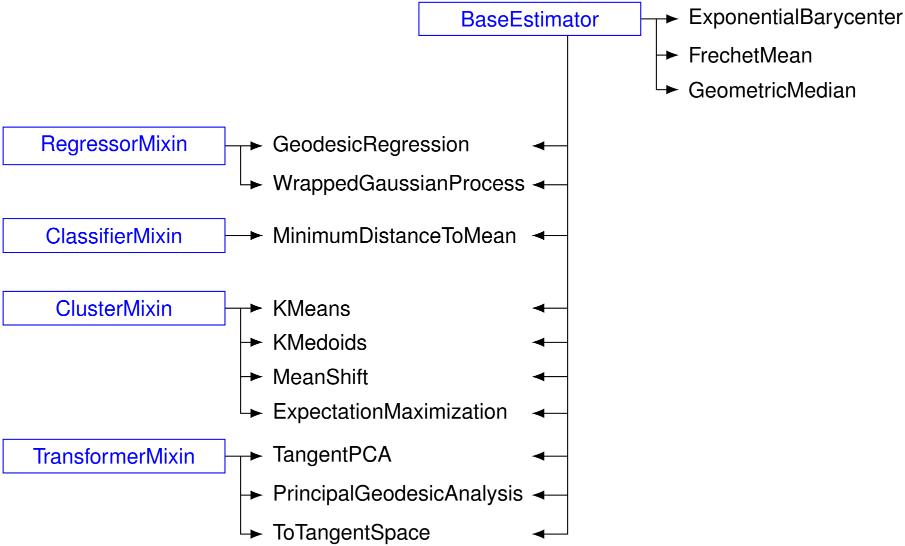
Fisher information: way of measuring the *amount of information* that an observable random variable X carries about *unknown parameters* θ of a distribution that models X.

$$I_\theta(x) = -\mathrm{E}\left[\nabla_\theta^2 \log p_\theta(x)\right]$$

Statistical manifold: $(\mathcal{P}, g)$

$$g(u, v) = u^\top I_\theta(x) v \text{ on } T_\theta \mathcal{P}$$

Geometric statistics

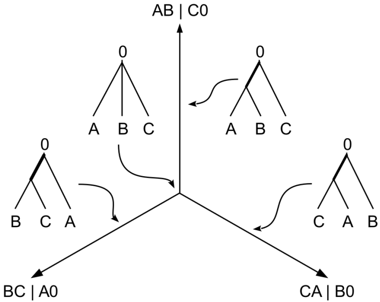


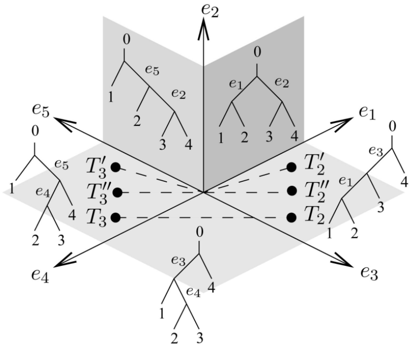
Beyond Riemannian

- GraphSpace X/T

$$d_{X/T}\left([x_1],[x_2]\right) = \min_{t\in T} d_X\left(x_1, t x_2\right)$$

- Spider
- BHVSpace, WaldSpace





Backend

```python
import geomstats.backend as gs
```

	numpy	autograd	pytorch
numerical precision		float64	
gpu			✓
automatic differentiation		✓	✓

Towards correctness

- (extended) `pytest`-based testing framework
- clear distinction between a test and the corresponding test data
- take advantage of:
  - ability to generate random points and vectors
  - known mathematical properties
  - composition with inverse

```
@pytest.mark.random
def test_exp_after_log(self, n_points, atol):
    base_point = self.data_generator.random_point(n_points)
    point = self.data_generator.random_point(n_points)

    tangent_vec = self.space.metric.log(point, base_point)
    point_ = self.space.metric.exp(tangent_vec, base_point)

    self.assertAllClose(point, point_, atol=atol)
```

Thank you for your attention!

https://geomstats.ai/
https://github.com/geomstats/geomstats

Nicolas Guigui, Nina Miolane, Xavier Pennec. 2022. **Introduction to Riemannian Geometry and Geometric Statistics: from basic theory to implementation with Geomstats**. Foundations and Trends in Machine Learning.

- Bauer, Martin et al. "Elastic Metrics on Spaces of Euclidean Curves: Theory and Algorithms". en. In: Journal of Nonline 34.3 (Apr. 2024), p. 56. ISSN: 1432-1467. DOI: 10.1007/s00332-024-10035-5. URL: https://doi.org/10.1007/s00332-024-10035-5.
- Brigant, Alice Le et al. Parametric information geometry with the package Geomstats. arXiv:2211.11643 [cs]. Nov. 2022. DOI: 10.48550/arXiv.2211.11643. URL: http://arxiv.org/abs/2211.11643.
- Guigui, Nicolas, Nina Miolane, and Xavier Pennec. "Introduction to Riemannian Geometry and Geometric Statistics: from basic theory to implementation with Geomstats". en. In: Foundations and Trends in Machine Learning (2022). URL: https://hal.inria.fr/hal-03766900.
- Pereira, Luís F. et al. Learning from landmarks, curves, surfaces, and shapes in Geomstats. arXiv:2406.10437 [cs]. June 2024. DOI: 10.48550/arXiv.2406.10437. URL: http://arxiv.org/abs/2406.10437 (visited on 02/10/2025).
- Thanwerdas, Yann. "Riemannian and stratified geometries of covariance and correlation matrices". Theses. Université Côte d'Azur, May 2022. URL: https://hal.archives-ouvertes.fr/tel-03698752 (visited on 09/29/2022).
- Tumpach, Alice Barbara. "Some aspects of infinite-dimensional Geometry: Theory and Applications". en. thesis. Lille University, Dec. 2022. URL: https://hal.science/tel-04052429.