Riemannian continuous models

Vladimirov Eduard, group M05-304a

18 октября 2024 г.

1 Introduction

Continuous models like Neural ODEs and normalizing flows have revolutionized machine learning by modeling complex data dynamics over continuous time. However, these models traditionally assume that data resides in Euclidean space, limiting their applicability to datasets with inherent non-Euclidean geometries. Many real-world datasets naturally lie on *Riemannian manifolds* — curved spaces that require specialized mathematical frameworks to capture their intrinsic geometry.

Riemannian geometry extends calculus to curved spaces, providing tools like *tangent spaces* and the *exponential map* to navigate manifolds. Incorporating this geometry into continuous models allows for more accurate and efficient learning from manifold-valued data. This essay explores two significant advancements in this area:

1. **Riemannian Continuous Normalizing Flows:** These models generalize normalizing flows to manifolds by defining flows via manifold ODEs, enabling the modeling of complex probability distributions while respecting geometric constraints.

2. **Neural Manifold ODE:** Extending Neural ODEs to manifolds, these models utilize techniques like the *dynamic chart method* to solve ODEs intrinsically on manifolds.

By leveraging the intrinsic geometry of manifolds, these Riemannian continuous models enhance performance in fields where data resides on curved spaces, such as robotics and medical imaging.

2 Background on Riemannian Geometry and Continuous Models

2.1 Riemannian Geometry Fundamentals

Let $\mathcal{M}$ be a smooth manifold of dimension n. A *smooth manifold* is a topological manifold equipped with an atlas of coordinate charts $\{(U_\alpha, \varphi_\alpha)\}$, where each U_α is an open subset of $\mathcal{M}$, and $\varphi_\alpha : U_\alpha \to \mathbb{R}^n$ is a homeomorphism onto its image, such that the transition maps $\varphi_\beta \circ \varphi_\alpha^{-1} : \varphi_\alpha(U_\alpha \cap U_\beta) \to \varphi_\beta(U_\alpha \cap U_\beta)$ are smooth whenever $U_\alpha \cap U_\beta \neq \varnothing$.

At each point $p \in \mathcal{M}$, the *tangent space* $T_p\mathcal{M}$ is a real vector space consisting of the equivalence classes of curves passing through p, or alternatively, the set of derivations at p. Elements of $T_p\mathcal{M}$ can be thought of as the possible directions in which one can tangentially pass through p.

A *Riemannian metric* on $\mathcal{M}$ is a smooth assignment of an inner product $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$ for each $p \in \mathcal{M}$, such that for any smooth vector fields X and Y on $\mathcal{M}$, the function $p \mapsto g_p(X_p, Y_p)$ is smooth. This metric allows for the measurement of angles, lengths, and volumes on the manifold.

Given a piecewise smooth curve $\gamma : [a, b] \to \mathcal{M}$, the *length* of γ is defined by

$$L(\gamma) = \int_a^b \|\dot{\gamma}(t)\|_{g_{\gamma(t)}} \, dt,$$

where $\dot{\gamma}(t)$ denotes the derivative of γ at t, and $\|\dot{\gamma}(t)\|_{g_{\gamma(t)}} = \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))}$ is the norm induced by the Riemannian metric.

A *geodesic* on $\mathcal{M}$ is a curve $\gamma : [a, b] \to \mathcal{M}$ that locally minimizes the length functional or, equivalently, satisfies the geodesic equation:

$$\frac{D}{dt}\dot{\gamma}(t) = 0,$$

where $\frac{D}{dt}$ is the covariant derivative along γ. Geodesics generalize the concept of straight lines to curved spaces and represent the paths of shortest distance between points on the manifold.

The *exponential map* at a point $p \in \mathcal{M}$ is a map

$$\exp_p : T_p\mathcal{M} \to \mathcal{M},$$

defined by $\exp_p(v) = \gamma_v(1)$, where $\gamma_v : [0,1] \to \mathcal{M}$ is the unique geodesic starting at p with initial velocity v, i.e., $\gamma_v(0) = p$ and $\dot{\gamma}_v(0) = v$. The exponential map provides a diffeomorphism from a neighborhood of the origin in $T_p\mathcal{M}$ to a neighborhood of p in $\mathcal{M}$.

The exponential map is crucial in transferring linear structures from the tangent space to the manifold. It allows for operations such as adding a tangent vector to a point on the manifold, analogous to vector addition in Euclidean space. This property is essential for defining manifold-valued differential equations and for extending continuous models to Riemannian manifolds.

2.2 Continuous Models in Machine Learning

Continuous models in machine learning, such as Neural Ordinary Differential Equations (Neural ODEs) and normalizing flows, have gained prominence due to their ability to model complex, continuous-time dynamics and flexible probability distributions.

Neural Ordinary Differential Equations (Neural ODEs). Neural ODEs model the hidden states of a neural network as the solution to an ODE parameterized by a neural network function f_θ:

$$\frac{dh(t)}{dt} = f_\theta(h(t), t), \quad h(0) = h_0.$$

This framework allows for adaptive computation and memory efficiency, as the continuous-depth model can be evaluated at arbitrary time points.

Normalizing Flows. Normalizing flows transform a simple base distribution $\pi(z)$ into a complex target distribution $p(x)$ by applying an invertible, differentiable mapping $f : \mathbb{R}^n \to \mathbb{R}^n$:

$$x = f(z), \quad z \sim \pi(z).$$

The change of variables formula computes the density of x:

$$p(x) = \pi(z) \left| \det \left(\frac{\partial f^{-1}}{\partial x} \right) \right| = \pi(z) \left| \det \left(\frac{\partial f(z)}{\partial z} \right)^{-1} \right|.$$

Taking the logarithm, the log-likelihood of x is:

$$\log p(x) = \log \pi(z) - \log \left| \det \left(\frac{\partial f(z)}{\partial z} \right) \right|,$$

where $z = f^{-1}(x)$.

Computing the determinant of the Jacobian $\frac{\partial f(z)}{\partial z}$ can be challenging, especially in high dimensions, leading to the development of continuous normalizing flows (CNFs), which use Neural ODEs to define f.

3 Riemannian Continuous Normalizing Flows

Normalizing flows are powerful tools for constructing complex probability distributions by transforming a simple base distribution through a sequence of invertible and differentiable mappings. When dealing with data that resides on a Riemannian manifold $\mathcal{M}$, it is essential to incorporate the manifold's intrinsic geometry into the model. *Riemannian Continuous Normalizing Flows* (RCNFs) extend continuous normalizing flows to Riemannian manifolds by defining flows via manifold ordinary differential equations (ODEs).

3.1 Theoretical Framework

3.1.1 Manifold ODEs

Let $\mathcal{M}$ be a d-dimensional Riemannian manifold with metric tensor $G(z)$. Consider a time-dependent vector field $f_\theta : \mathcal{M} \times [0, T] \to T\mathcal{M}$, parameterized by θ, where $T\mathcal{M}$ denotes the tangent bundle of $\mathcal{M}$. The flow $\varphi_t : \mathcal{M} \to \mathcal{M}$ induced by f_θ is obtained by solving the manifold ODE:

$$\frac{dz(t)}{dt} = f_\theta(z(t), t), \quad z(0) = z_0 \in \mathcal{M}, \tag{1}$$

where $z(t) \in \mathcal{M}$ for all $t \in [0, T]$. Under appropriate conditions on f_θ, the flow φ_t is a diffeomorphism at each time t.

3.1.2 Change in Log-Density

The change in the log-density along the flow is governed by the continuity equation on the manifold. The derivative of the log-density $p_t(z(t))$ with respect to time is:

$$\frac{d}{dt} \log p_t(z(t)) = -\operatorname{div}(f_\theta)(z(t), t), \tag{2}$$

where $\operatorname{div}(f_\theta)$ is the *Riemannian divergence* of the vector field f_θ.

Integrating over time from $t = 0$ to $t = T$, we obtain:

$$\log p_T(z(T)) = \log p_0(z(0)) - \int_0^T \operatorname{div}(f_\theta)(z(t), t)\, dt. \tag{3}$$

This equation allows us to compute the log-density of the transformed variable $z(T)$ given the base density $p_0(z(0))$.

3.1.3 Riemannian Divergence

The Riemannian divergence of a vector field f_θ at point $z \in \mathcal{M}$ is defined as:

$$\operatorname{div}(f_\theta)(z) = \frac{1}{\sqrt{\det G(z)}} \frac{\partial}{\partial z^i} \left(\sqrt{\det G(z)}\, f_\theta^i(z) \right), \tag{4}$$

where:

- $\det G(z)$ is the determinant of the metric tensor $G(z)$.
- $f_\theta^i(z)$ are the components of f_θ in local coordinates.
- The Einstein summation convention is used over repeated indices i.
- d represents the dimensionality of the manifold $\mathcal{M}$, i.e., $f_\theta(z) \in \mathbb{R}^d$.

Computing the divergence directly involves calculating derivatives of both the vector field components and the metric tensor, which can be computationally intensive, especially as the dimensionality d increases.

To avoid computing the full Jacobian matrix $\nabla_z f_\theta(z, t) \in \mathbb{R}^{d \times d}$, we can use *Hutchinson's estimator* to stochastically estimate the divergence:

$$\operatorname{div}(f_\theta)(z) \approx \frac{1}{K} \sum_{k=1}^{K} \varepsilon_k^\top \left(\nabla_z f_\theta(z, t)\, \varepsilon_k \right), \tag{5}$$

where:

- K is the number of random samples used for the estimation (typically $K = 1$ for efficiency).
- $\varepsilon_k \in \mathbb{R}^d$ are random vectors sampled from a distribution with zero mean and identity covariance, usually $\varepsilon_k \sim \mathcal{N}(0, I_d)$.

- $\nabla_z f_\theta(z,t) \in \mathbb{R}^{d\times d}$ is the Jacobian matrix of f_θ with respect to z, representing the gradient in the ambient space $\mathbb{R}^d$.

Since $f_\theta(z,t)$ is defined in the tangent space $T_z\mathcal{M}$ but expressed in ambient coordinates, its Jacobian is naturally a $d \times d$ matrix in $\mathbb{R}^d$.

This stochastic approximation reduces computational complexity from $O(d^2)$ to $O(Kd)$, where K is typically small.

3.1.4 Ensuring Tangency

To ensure that the vector field $f_\theta(z,t)$ lies in the tangent space $T_z\mathcal{M}$ at each point z, we define it using a projection:

$$f_\theta(z,t) = P_z\,\tilde{f}_\theta(z,t), \tag{6}$$

where:

- $\tilde{f}_\theta(z,t) \in \mathbb{R}^d$ is an unconstrained vector field in the ambient space.
- $P_z \in \mathbb{R}^{d\times d}$ is the orthogonal projection matrix onto $T_z\mathcal{M}$.

3.1.5 Solving the Manifold ODE

Solving the manifold ODE requires numerical integration methods that preserve the manifold structure. One approach is to use the *exponential map*:

$$z(t+\Delta t) = \exp_{z(t)}\left(\Delta t\, f_\theta(z(t),t)\right), \tag{7}$$

where $\exp_z$ is the exponential map at point z.

Alternative Numerical Methods - Projection-Based Solvers: Use standard ODE solvers in $\mathbb{R}^d$ and project the solution back onto $\mathcal{M}$ at each step.

- Manifold-Specific Solvers: Utilize integrators designed for manifolds, which intrinsically respect the manifold constraints.

3.1.6 Log-Likelihood Computation

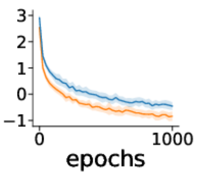
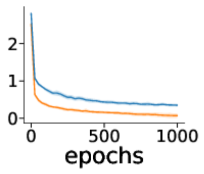
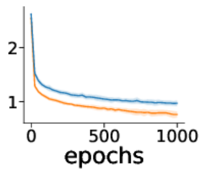
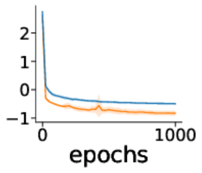
The total log-likelihood of a sample $x \in \mathcal{M}$ is given by:

$$\log p(x) = \log \pi(z(0)) - \int_0^T \operatorname{div}(f_\theta)(z(t),t)\,dt, \tag{8}$$

with $z(T) = x$.

3.2 RCNF for Earthquake Density Estimation on the Sphere

Table 3: Negative test log-likelihood of continuous normalizing flows on $\mathbb{S}^2$ datasets.

	Volcano	Earthquake	Flood	Fire
Stereographic 🟦	$-0.54_{\pm 0.14}$	$0.34_{\pm 0.04}$	$0.96_{\pm 0.05}$	$-0.50_{\pm 0.04}$
Riemannian 🟧	$\mathbf{-0.92}_{\pm 0.15}$	$\mathbf{0.07}_{\pm 0.06}$	$\mathbf{0.75}_{\pm 0.06}$	$\mathbf{-0.84}_{\pm 0.08}$
Learning curves				
Data size	829	6124	4877	12810

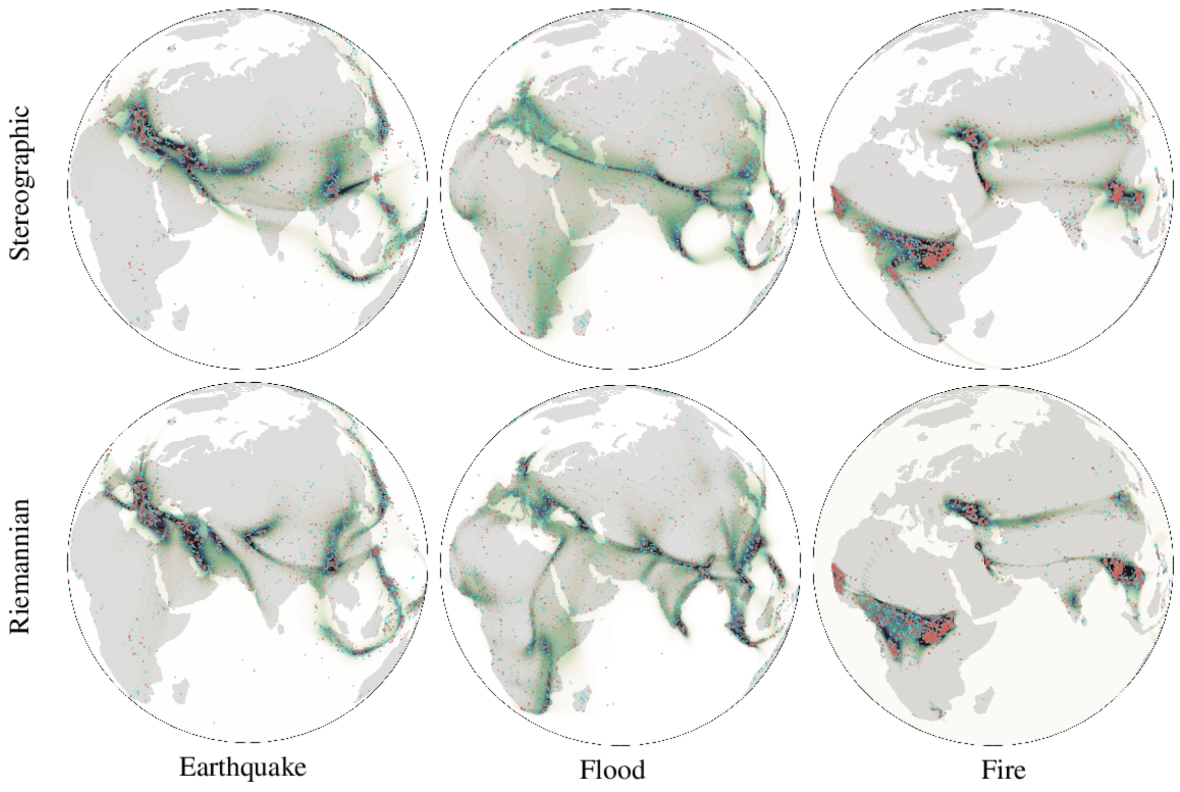


Figure 6: Density estimation for earth sciences data. Blue and red dots represent training and testing datapoints

1. **Data Preparation**
   - Convert latitude and longitude from degrees to radians:

$$\varphi_i \leftarrow \varphi_i \times \frac{\pi}{180}, \quad \lambda_i \leftarrow \lambda_i \times \frac{\pi}{180}$$

- Convert spherical coordinates to Cartesian coordinates on $\mathbb{S}^2$:

$$\begin{cases} x_i = \cos(\varphi_i)\cos(\lambda_i), \\ y_i = \cos(\varphi_i)\sin(\lambda_i), \\ z_i = \sin(\varphi_i) \end{cases}$$

- Form data points $z_i = [x_i, y_i, z_i]^\top \in \mathbb{R}^3$.
- Ensure each z_i lies on the unit sphere:

$$z_i \leftarrow \frac{z_i}{\|z_i\|_2}$$

2. **Initialize Base Distribution**

- Set the base distribution to the uniform distribution on $\mathbb{S}^2$:

$$p_0(z(0)) = \frac{1}{4\pi}, \quad \forall z(0) \in \mathbb{S}^2$$

3. **Define Vector Field $f_\theta(z,t)$**

(a) **Neural Network Parameterization**

- **Input Layer**: For each point z, compute geodesic distances to a set of anchor points $\{a_j\} \subset \mathbb{S}^2$:

$$d_j(z) = \arccos(z^\top a_j)$$

- **Neural Network Architecture**:
  - Inputs: $[d_1(z), d_2(z), \ldots, d_m(z), t]$
  - Hidden Layers: 3 layers with 64 units each, activation function tanh
  - Output Layer: Unconstrained vector $\tilde{f}_\theta(z,t) \in \mathbb{R}^3$

(b) **Ensuring Tangency**

- Project $\tilde{f}_\theta(z,t)$ onto the tangent space $T_z\mathbb{S}^2$:

$$f_\theta(z,t) = \left(I_3 - zz^\top\right)\tilde{f}_\theta(z,t)$$

where I_3 is the 3×3 identity matrix.

4. **Set Up Manifold ODE**

(a) **Define the manifold ODE governing the flow**

$$\frac{dz(t)}{dt} = f_\theta(z(t), t), \quad z(0) \sim p_0(z(0))$$

(b) **Use the Change of Variables Formula**

$$\frac{d}{dt} \log p_t(z(t)) = -\operatorname{div}(f_\theta)(z(t), t)$$

(c) **Compute Riemannian Divergence**

- Since $\mathbb{S}^2$ has constant metric determinant, simplify divergence to:
$$\operatorname{div}(f_\theta)(z) = \nabla_z \cdot f_\theta(z, t)$$

(d) **Approximate Divergence using Hutchinson's Estimator**

- Generate K random vectors $\varepsilon_k \sim \mathcal{N}(0, I_3)$.
- Approximate divergence:

$$\operatorname{div}(f_\theta)(z) \approx \frac{1}{K} \sum_{k=1}^{K} \varepsilon_k^\top \left(\nabla_z f_\theta(z, t)\, \varepsilon_k\right)$$

(e) **Integrate Log-Density**

$$\log p_T(z(T)) = \log p_0(z(0)) - \int_0^T \operatorname{div}(f_\theta)(z(t), t)\, dt$$

5. **Numerical Integration**

   (a) Use an ODE solver (e.g., Runge-Kutta method) to integrate $z(t)$ and $\log p_t(z(t))$ from $t = 0$ to $t = T$.

   (b) After each integration step, normalize $z(t)$ to ensure it

6. **Compute Loss Function** We minimize the *negative log-likelihood (NLL)* over the observed data $\{z_i\}_{i=1}^N$:

$$L(\theta) = -\frac{1}{N} \sum_{i=1}^{N} \log p_T(z_i), \tag{9}$$

7. **Compute Gradients** Use the adjoint sensitivity method to compute gradients with respect to θ.

- **Adjoint State Evolution**
  - Initialize adjoint state at $t = T$:

$$a(T) = \frac{\partial L}{\partial z(T)} = 0$$

  - Integrate backward in time:

$$\frac{da(t)}{dt} = -\left(\nabla_{z(t)} f_\theta(z(t), t)\right)^\top a(t) - \nabla_{z(t)} \operatorname{div}(f_\theta)(z(t), t)$$

- **Gradient with Respect to Parameters**

$$\frac{dL}{d\theta} = -\int_0^T a(t)^\top \frac{\partial f_\theta(z(t), t)}{\partial \theta}\, dt$$

8. **Baseline Comparison with Stereographic Projection**

- Forward Projection to Plane:

$$\begin{cases} u_i = \dfrac{x_i}{1 - z_i}, \\ v_i = \dfrac{y_i}{1 - z_i} \end{cases}$$

- Apply Euclidean CNF in $\mathbb{R}^2$.
- Inverse Projection to Sphere:

$$\begin{cases} x = \dfrac{2u}{u^2 + v^2 + 1}, \\ y = \dfrac{2v}{u^2 + v^2 + 1}, \\ z = \dfrac{u^2 + v^2 - 1}{u^2 + v^2 + 1} \end{cases}$$

4 Conclusion

Riemannian continuous models extend traditional continuous frameworks, such as Neural Ordinary Differential Equations and normalizing flows, to data residing on Riemannian manifolds. By leveraging the intrinsic geometry of manifolds, these models provide a principled approach to modeling manifold-valued data

Список литературы

Alimisis, Foivos и др. (2020). «A continuous-time perspective for modeling acceleration in Riemannian optimization». В: *International Conference on Artificial Intelligence and Statistics*. PMLR, с. 1297—1307.

Huang, Chin-Wei и др. (2022). «Riemannian diffusion models». В: *Advances in Neural Information Processing Systems* 35, с. 2750—2761.

Jeong, Seungwoo и др. (2023). «Deep efficient continuous manifold learning for time series modeling». В: *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Katsman, Isay и др. (2024). «Riemannian residual neural networks». В: *Advances in Neural Information Processing Systems* 36.

Lou, Aaron и др. (2020). «Neural manifold ordinary differential equations». В: *Advances in Neural Information Processing Systems* 33, с. 17548—17558.

Mathieu, Emile и Maximilian Nickel (2020). «Riemannian continuous normalizing flows». В: *Advances in Neural Information Processing Systems* 33, с. 2503—2515.

Papaioannou, Panagiotis G и др. (2022). «Time-series forecasting using manifold learning, radial basis function interpolation, and geometric harmonics». В: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 32.8.