Neural Geodesic Flows

Extending to Pseudo-Riemannian Geometry

Mihir Talati (after J. Bürge, ETH Zurich 2025)

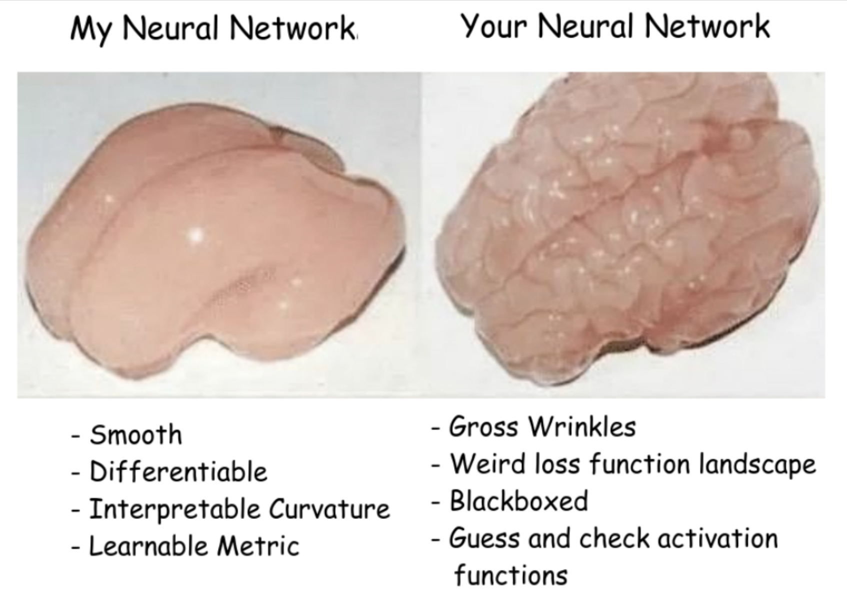
December 15, 2025

Overview

Presentation Outline

- Overview
- Foundations & Geometry
- Model, Improvements, Results

Motivation


My Neural Network
Your Neural Network
- Smooth
- Differentiable
- Interpretable Curvature
- Learnable Metric
- Gross Wrinkles
- Weird loss function landscape
- Blackboxed
- Guess and check activation functions

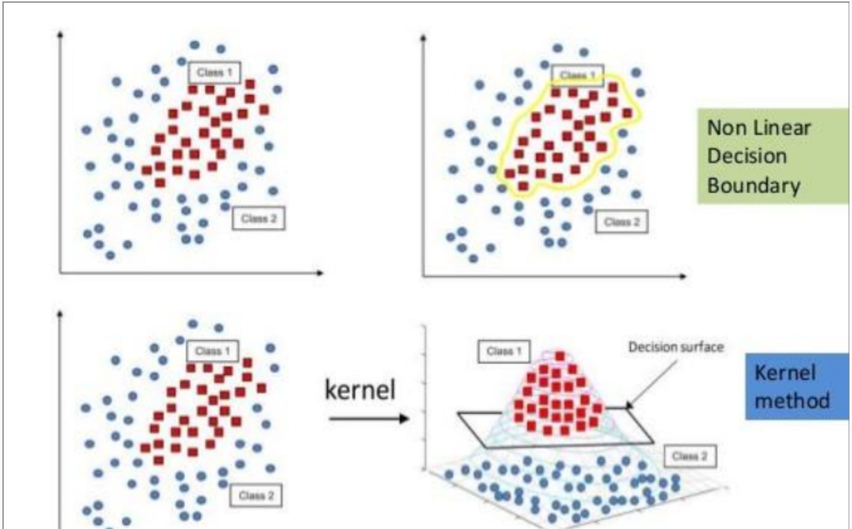
Foundations & Geometry

Classical NNs and the need for an ansatz

- Universal approximation: NNs can fit many functions, but we need a **good ansatz** for stability and meaning.
- Our ansatz: observed dynamics are geodesic flows of a learned metric $\Rightarrow$ geometric invariants and structure.

Manifold hypothesis

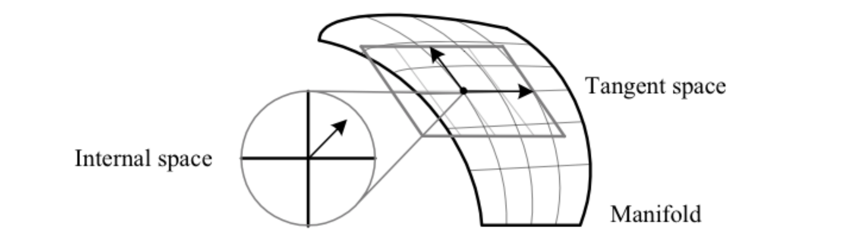
- Data in high-D $\mathbb{R}^n$ lie near a low-D manifold.



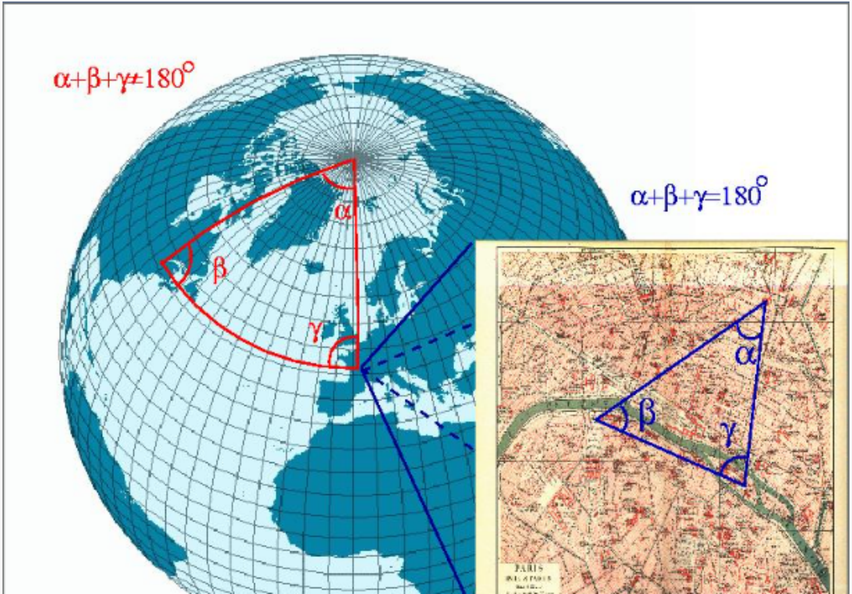
Geometry primer

- Manifold + chart: local coordinates; tangent bundle TM = positions + velocities.
- Geodesics: shortest paths for a metric; governed by the geodesic equation $\ddot{x}^k + \Gamma^k_{ab}\dot{x}^a\dot{x}^b = 0$.

Tangent Planes



Example



Riemannian vs pseudo-Riemannian metrics

- Riemannian: SPD metric, all spacelike (good for Euclidean-like systems).
- Pseudo-Riemannian: mixed signature (timelike + spacelike), needed for Lorentzian/relativistic systems.
- Extension: fixed-signature metric net + log-det regularizer; future: gating network to select signature automatically.

Model, Improvements, Results

ResNets to NGF (condensed)

- A (very) simplified ResNet block:

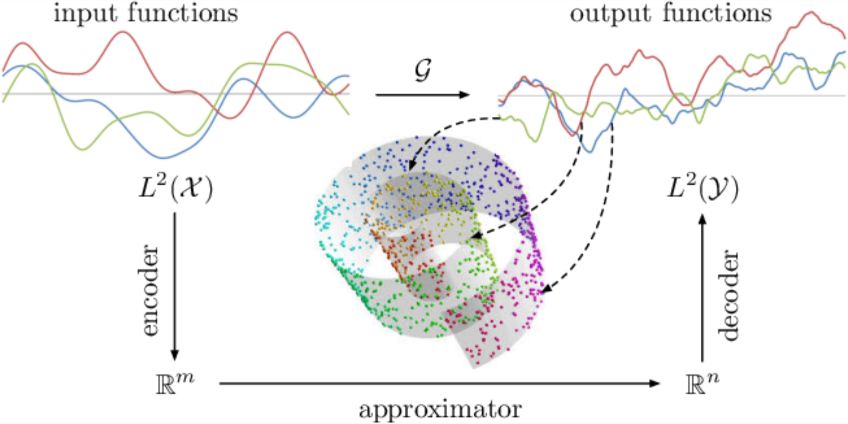
$$h_{k+1} = h_k + f_\theta(h_k)$$

  where k is the layer index.
- If you think of k as discrete time and f_θ as a velocity field, this is exactly a forward Euler step for the ODE

$$\frac{dh}{dt} = f_\theta(h(t)).$$

- Idea of *Neural ODEs*: instead of stacking many discrete layers, treat depth as continuous time and let an ODE solver play the role of the network.
- Forward pass $\Rightarrow$ solve an ODE; backward pass $\Rightarrow$ differentiate *through* the ODE solver (adjoint method, automatic differentiation, ...).
- NGF: a structured Neural ODE where the vector field is the geodesic flow of a learned metric g.

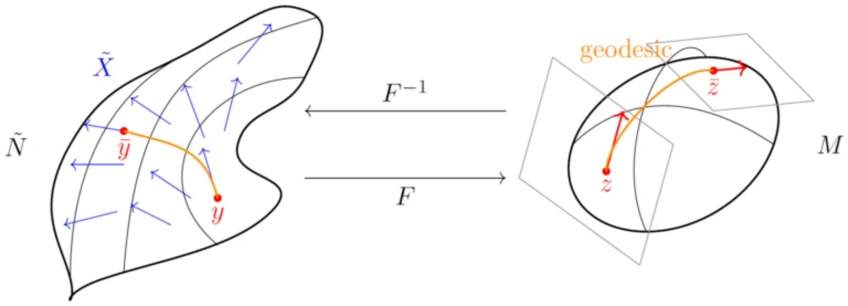
Neural Geodesic Flows



Encoder, metric, decoder + flow

- Learn encoder ψ_θ (to (x, v)), decoder ϕ_θ (back to data), metric $g_\theta(x)$.
- Baseline g_θ: SPD via $I + LL^\top$.
- Extension g_θ: fixed-signature pseudo-R (positive *and* negative eigenvalues) $+$ log-det regularization; future gating for signature.

NGF forward pass (visual)



Changes (model & loss)

- **Metric network**: new pseudo-R module with fixed signature (p, q); softplus scales with signs.
- **Conditioning**: `min_diagonal`/`min_scale` to keep g non-degenerate near horizons.
- **Regularizer**: loss on $\log|\det g|$ to penalize near-singular metrics (configurable floor/weight).
- **Training**: PR runs use signatures $(1, 1)$ or $(2, 2)$, stronger metric regularization, gentler LR.

Why signature matters (physics alignment)

- Lorentzian metrics encode causal structure (timelike vs spacelike).
- Learned eigenvalues at a probe point:
  - Minkowski_prngf: $[-0.12, 0.29]$ (one timelike, one spacelike).
  - AdS2_prngf: $[-0.036, 0.017]$ (Lorentzian, small curvature).
  - RNGF baselines: both > 0 (purely spatial).
- Interpretation: mixed signs capture lightcones/horizons; SPD forces all spacelike $\Rightarrow$ distorted distances and rollout drift.

Findings & next steps (pseudo-R vs R)

- Findings: pseudo-R keeps correct signature on Lorentzian toy spacetimes (Minkowski, AdS2, Schwarzschild); Riemannian baselines drift on curved Lorentzian rollouts.
- Limitations: single chart; flat datasets don't stress signature; Schwarzschild checkpoints need clean retrain.
- Next steps: gating network to select signature adaptively; atlas-based NGFs; stronger pseudo-R regularizers and long-horizon tests.