Manifold Learning and Visualization

Mathematical Foundations and Implementation

Deep Learning on Manifolds Project

March 16, 2025

Contents

1 Introduction

1.1 Motivation and Overview

The study of manifolds lies at the intersection of differential geometry, topology, and modern machine learning. While manifolds have been a cornerstone of mathematics for centuries, their application in machine learning has gained significant traction in recent decades. This document provides a comprehensive overview of our project, which combines deep learning techniques with differential geometry to learn and visualize manifold structures from data.

The fundamental motivation behind this work is to address a key challenge in modern data analysis: high-dimensional data often lies on or near lower-dimensional manifolds embedded in the ambient space. By learning these manifold structures, we can:

- Discover intrinsic geometric properties of the data
- Perform dimensionality reduction while preserving important geometric features
- Visualize complex data in an interpretable manner
- Enable more efficient computation on the lower-dimensional representation
- Understand the underlying physics or generative processes

Our approach is unique in that we not only learn the embedding of data into a manifold but also learn the metric tensor that defines the geometry of this manifold. This allows us to compute geodesics (shortest paths), measure curvature, and visualize how particles would move through the learned space according to the principles of differential geometry.

1.2 Applications and Significance

The techniques developed in this project have applications across numerous domains:

- **Physics**: Modeling spacetime curvature and gravitational effects
- **Computer Vision**: Understanding the manifold of natural images
- **Robotics**: Planning optimal paths in configuration spaces
- **Neuroscience**: Analyzing neural activity patterns and brain states
- **Bioinformatics**: Modeling protein folding landscapes
- **Economics**: Understanding market dynamics and equilibria

By bridging the gap between abstract mathematical concepts and practical machine learning, our work enables researchers to gain deeper insights into complex systems and their underlying geometric structures.

2 Mathematical Foundations

2.1 Manifolds and Differential Geometry

A manifold is a topological space that locally resembles Euclidean space. More formally, an n-dimensional manifold $\mathcal{M}$ is a topological space where each point has a neighborhood that is homeomorphic to an open subset of $\mathbb{R}^n$. This local resemblance to Euclidean space allows us to perform calculus on manifolds by working in local coordinate systems.

The key objects in differential geometry that we will be concerned with include:

- **Charts and Atlases**: A chart is a homeomorphism from an open subset of the manifold to an open subset of $\mathbb{R}^n$. An atlas is a collection of charts that cover the entire manifold.
- **Tangent Spaces**: At each point $p \in \mathcal{M}$, the tangent space $T_p\mathcal{M}$ is the vector space of all possible directions in which one can tangentially pass through p.
- **Metric Tensor**: A Riemannian metric g on $\mathcal{M}$ is a smooth assignment of an inner product g_p to each tangent space $T_p\mathcal{M}$. In local coordinates, the metric is represented by a positive-definite symmetric matrix g_{ij}.
- **Connections and Christoffel Symbols**: A connection defines how vectors are transported along curves on the manifold. The Christoffel symbols Γ_{ij}^k encode this information in local coordinates.
- **Geodesics**: These are curves that locally minimize the distance between points. They generalize the concept of straight lines to curved spaces and satisfy the geodesic equation:

$$\frac{d^2x^k}{dt^2} + \Gamma_{ij}^k \frac{dx^i}{dt}\frac{dx^j}{dt} = 0 \tag{1}$$

- **Curvature**: The Riemann curvature tensor R_{jkl}^i measures the extent to which the manifold deviates from being flat. The Ricci tensor R_{ij} and scalar curvature R are contractions of the Riemann tensor.

2.2 Riemannian Geometry and Metrics

In Riemannian geometry, the metric tensor g is the fundamental object that defines the geometry of the manifold. Given a coordinate system $(x^1, x^2, \ldots, x^n)$, the metric tensor has components g_{ij} that define how to measure distances and angles:

$$ds^2 = g_{ij}dx^i dx^j \tag{2}$$

where we use Einstein's summation convention (summing over repeated indices). The metric tensor allows us to:

- Compute the length of a curve $\gamma : [a, b] \to \mathcal{M}$ via:

$$L(\gamma) = \int_a^b \sqrt{g_{ij}\frac{d\gamma^i}{dt}\frac{d\gamma^j}{dt}}dt \tag{3}$$

- Define the inner product between tangent vectors $v, w \in T_p\mathcal{M}$:

$$\langle v, w\rangle_g = g_{ij}v^i w^j \tag{4}$$

- Compute angles between curves and vectors
- Measure volumes and areas

The inverse metric tensor g^{ij} satisfies $g^{ik}g_{kj} = \delta_j^i$, where δ_j^i is the Kronecker delta. The inverse metric is used to raise indices in tensor calculations.

2.3 Spacetime Metrics and General Relativity

In the context of general relativity, spacetime is modeled as a 4-dimensional pseudo-Riemannian manifold with a metric of Lorentzian signature $(-,+,+,+)$. The metric encodes the gravitational field, determining how matter and energy curve spacetime.

Some important spacetime metrics include:

- **Minkowski Metric** (flat spacetime):

$$ds^2 = -c^2dt^2 + dx^2 + dy^2 + dz^2 \tag{5}$$

- **Schwarzschild Metric** (spherically symmetric black hole):

$$ds^2 = -\left(1 - \frac{2GM}{c^2r}\right)c^2dt^2 + \left(1 - \frac{2GM}{c^2r}\right)^{-1}dr^2 + r^2d\Omega^2 \tag{6}$$

  where $d\Omega^2 = d\theta^2 + \sin^2\theta d\phi^2$ is the metric on a unit sphere.

- **Friedmann-Lemaître-Robertson-Walker (FLRW) Metric** (expanding universe):

$$ds^2 = -c^2dt^2 + a(t)^2\left[\frac{dr^2}{1-kr^2} + r^2d\Omega^2\right] \tag{7}$$

  where $a(t)$ is the scale factor and k is the curvature parameter.

In our project, we allow the neural network to learn arbitrary metrics, which may correspond to known physical spacetimes or novel geometries discovered from data.

2.4 Christoffel Symbols and Geodesic Equations

The Christoffel symbols are defined in terms of the metric tensor and its derivatives:

$$\Gamma^k_{ij} = \frac{1}{2}g^{kl}\left(\frac{\partial g_{jl}}{\partial x^i} + \frac{\partial g_{il}}{\partial x^j} - \frac{\partial g_{ij}}{\partial x^l}\right) \tag{8}$$

These symbols are not tensors but are essential for defining covariant derivatives and geodesics. The geodesic equation, which describes the path of a freely falling particle in curved spacetime, is:

$$\frac{d^2x^\mu}{d\tau^2} + \Gamma^\mu_{\alpha\beta}\frac{dx^\alpha}{d\tau}\frac{dx^\beta}{d\tau} = 0 \tag{9}$$

where τ is the proper time. In our implementation, we numerically integrate this equation to visualize particle trajectories in the learned manifold.

2.5 Curvature Tensors

The Riemann curvature tensor measures how parallel transport around an infinitesimal loop fails to return a vector to its original orientation. It is defined as:

$$R^\rho_{\sigma\mu\nu} = \frac{\partial\Gamma^\rho_{\sigma\nu}}{\partial x^\mu} - \frac{\partial\Gamma^\rho_{\sigma\mu}}{\partial x^\nu} + \Gamma^\rho_{\lambda\mu}\Gamma^\lambda_{\sigma\nu} - \Gamma^\rho_{\lambda\nu}\Gamma^\lambda_{\sigma\mu} \tag{10}$$

The Ricci tensor is a contraction of the Riemann tensor:

$$R_{\mu\nu} = R^\lambda_{\mu\lambda\nu} \tag{11}$$

And the scalar curvature is a further contraction:

$$R = g^{\mu\nu} R_{\mu\nu} \tag{12}$$

These curvature measures are important for understanding the geometry of the manifold and play a central role in Einstein's field equations:

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu} \tag{13}$$

where Λ is the cosmological constant and $T_{\mu\nu}$ is the stress-energy tensor.

3 Machine Learning on Manifolds

3.1 Manifold Hypothesis and Dimensionality Reduction

The manifold hypothesis in machine learning posits that high-dimensional data often lies on or near a lower-dimensional manifold embedded in the ambient space. This insight has led to numerous dimensionality reduction techniques:

- **Principal Component Analysis (PCA)**: Projects data onto the directions of maximum variance, but is limited to linear subspaces.
- **Multidimensional Scaling (MDS)**: Preserves pairwise distances between points, but struggles with highly curved manifolds.
- **ISOMAP**: Approximates geodesic distances using graph shortest paths, better capturing the intrinsic geometry.
- **t-SNE and UMAP**: Create low-dimensional embeddings that preserve local neighborhood structure.
- **Autoencoders**: Learn nonlinear mappings between the high-dimensional space and a lower-dimensional latent space.

Our approach goes beyond these methods by explicitly learning the metric tensor of the manifold, allowing us to compute geodesics, measure curvature, and understand the intrinsic geometry in greater detail.

3.2 Neural Networks for Manifold Learning

Our approach uses neural networks to learn both the embedding of data into a manifold and the metric tensor that defines the geometry of this manifold. The architecture consists of two main components:

1. **Embedding Network**: Maps input data from the high-dimensional space to the lower-dimensional manifold.
2. **Metric Network**: Learns the components of the metric tensor at each point on the manifold.

The embedding network $f_\theta : \mathbb{R}^d \to \mathbb{R}^n$ with parameters θ maps input data $x \in \mathbb{R}^d$ to points $y = f_\theta(x) \in \mathbb{R}^n$ on the manifold, where typically $n \ll d$. The architecture of this network can be a standard feedforward neural network, a convolutional network for image data, or a more complex architecture depending on the application.

The metric network $g_\phi : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ with parameters ϕ maps points y on the manifold to the components of the metric tensor $g_{ij}(y)$ at that point. The output of this network must be a symmetric positive-definite matrix (for Riemannian manifolds) or a matrix with the appropriate signature (for pseudo-Riemannian manifolds like spacetime).

3.3 Ricci Flow and Geometric Evolution

Ricci flow is a geometric evolution equation that deforms the metric tensor of a manifold in the direction of the Ricci curvature:

$$\frac{\partial g_{ij}}{\partial t} = -2R_{ij} \tag{14}$$

This flow tends to smooth out irregularities in the curvature, making it a useful tool for regularizing the learned metric. In our implementation, we incorporate a discretized version of Ricci flow as a regularization term in the loss function:

$$\mathcal{L}_{\text{Ricci}} = \lambda \sum_{i,j} \left\| \frac{\partial g_{ij}}{\partial t} + 2R_{ij} \right\|^2 \tag{15}$$

where λ is a hyperparameter controlling the strength of the regularization.

3.4 Geodesic Loss Functions

To ensure that the learned manifold captures the intrinsic geometry of the data, we define loss functions based on geodesic distances. The geodesic distance between two points p and q on the manifold is the length of the shortest path connecting them:

$$d_g(p, q) = \inf_{\gamma} \int_0^1 \sqrt{g_{ij}(\gamma(t)) \frac{d\gamma^i}{dt} \frac{d\gamma^j}{dt}} dt \tag{16}$$

where the infimum is taken over all smooth curves $\gamma : [0, 1] \to \mathcal{M}$ with $\gamma(0) = p$ and $\gamma(1) = q$.

Computing exact geodesic distances is computationally expensive, so we approximate them using numerical integration of the geodesic equation. Our geodesic loss function encourages the learned manifold to preserve distances between nearby points:

$$\mathcal{L}_{\text{geodesic}} = \sum_{(i,j) \in \mathcal{N}} (d_g(f_\theta(x_i), f_\theta(x_j)) - d_{\text{data}}(x_i, x_j))^2 \tag{17}$$

where $\mathcal{N}$ is a set of pairs of nearby points and d_{data} is a distance measure in the original data space.

4 Implementation Details

4.1 Neural Network Architecture

Our implementation uses PyTorch to define the neural network architecture. The core components are:

- **ManifoldEmbedding**: A neural network module that maps input data to the manifold and learns the metric tensor.
- **RicciCurvature**: A class that computes Christoffel symbols, Riemann tensor, Ricci tensor, and scalar curvature from the metric tensor.
- **GeodesicLoss**: A loss function that computes geodesic distances and encourages the learned manifold to preserve the structure of the data.
- **CovariantDescent**: An optimizer that takes into account the geometry of the parameter space during optimization.
- **RicciFlow**: A class that implements the Ricci flow regularization.

The ManifoldEmbedding class is defined as follows:

Algorithm 1 ManifoldEmbedding Neural Network

Input: input_dim, embedding_dim
Initialize metric tensor g with appropriate signature
Define embedding network with layers:
    Linear(input_dim, 128)
    BatchNorm1d(128)
    ReLU()
    Dropout(0.2)
    Linear(128, 64)
    BatchNorm1d(64)
    ReLU()
    Dropout(0.2)
    Linear(64, embedding_dim)
Forward Pass:
    Embed input data using the embedding network
    Return embedded data
Methods:
    get_metric(): Return the current metric tensor
    christoffel_symbols(g, g_inv): Compute Christoffel symbols
    geodesic_equations(t, y, Gamma): Define geodesic differential equations

4.2 Computing Christoffel Symbols and Curvature

The computation of Christoffel symbols and curvature tensors is a critical part of our implementation. We use automatic differentiation to compute the derivatives of the metric tensor:

Algorithm 2 Computing Christoffel Symbols

Input: Metric tensor g, point p on the manifold
Compute inverse metric g^{-1}
Initialize Christoffel symbols Γ^k_{ij} as zeros
for each index i, j, k **do**
    for each index l **do**
        Compute $\partial_i g_{jl}$, $\partial_j g_{il}$, $\partial_l g_{ij}$ using automatic differentiation
        Γ^k_{ij} += $\frac{1}{2} g^{kl}(\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij})$
    end for
end for
Return: Christoffel symbols Γ^k_{ij}

Similarly, we compute the Riemann curvature tensor, Ricci tensor, and scalar curvature using their definitions in terms of Christoffel symbols and their derivatives.

4.3 Numerical Integration of Geodesic Equations

To compute geodesics and visualize particle trajectories, we numerically integrate the geodesic equation using methods such as Runge-Kutta or Verlet integration:

4.4 Visualization in C++

To visualize the learned manifold and particle trajectories, we export the metric tensor, Christoffel symbols, and curvature tensors to a format that can be read by our C++ visualization program:

The C++ visualization program reads these files and creates an interactive 3D visualization of the manifold, allowing users to:

Algorithm 3 Numerical Integration of Geodesic Equations

Input: Starting point p, initial velocity v, metric tensor g, number of steps N, step size Δt
Compute Christoffel symbols Γ at p
Initialize trajectory = $[p]$
Initialize current position $x = p$
Initialize current velocity $\dot{x} = v$
for $i = 1$ to N **do**
    Compute acceleration: $\ddot{x}^k = -\Gamma^k_{ij} \dot{x}^i \dot{x}^j$
    Update velocity: $\dot{x} \mathrel{+}= \ddot{x} \cdot \Delta t$
    Update position: $x \mathrel{+}= \dot{x} \cdot \Delta t$
    Recompute Christoffel symbols Γ at new position x
    Append x to trajectory
end for
Return: trajectory

Algorithm 4 Exporting Manifold Data for Visualization

Input: Trained model, output directory
Extract metric tensor g from the model
Compute Christoffel symbols Γ
Compute Riemann tensor R, Ricci tensor Ric, and scalar curvature R
Export metric tensor to CSV file
Export Christoffel symbols to CSV file
Export Riemann tensor to CSV file
Export embedding of a grid of points for visualization

- View the curved spacetime represented by the metric
- Add particles and observe their trajectories along geodesics
- Visualize the curvature using color mapping
- Interact with the visualization by rotating, zooming, and adding new particles

5 Experimental Results

5.1 Synthetic Data Experiments

We first validate our approach on synthetic datasets where the ground truth manifold structure is known. We generate data from several manifolds:

- **Sphere**: Points sampled from a 2-sphere embedded in $\mathbb{R}^3$
- **Torus**: Points sampled from a torus with major radius R and minor radius r
- **Swiss Roll**: A 2D manifold embedded in $\mathbb{R}^3$ with intrinsic curvature
- **Synthetic Spacetime**: A 4D manifold with Lorentzian signature simulating a curved spacetime

For each dataset, we train our model to learn both the embedding and the metric tensor. We evaluate the quality of the learned manifold using several metrics:

- **Embedding Error**: Mean squared error between the ground truth embedding and the learned embedding

- **Metric Error**: Mean squared error between the ground truth metric tensor and the learned metric tensor
- **Geodesic Error**: Difference between ground truth geodesic distances and distances computed using the learned metric
- **Curvature Error**: Difference between ground truth curvature measures and those computed from the learned metric

Table 1: Quantitative Results on Synthetic Datasets

Dataset	Embedding Error	Metric Error	Geodesic Error	Curvature Error
Sphere	0.023	0.045	0.078	0.112
Torus	0.031	0.057	0.092	0.135
Swiss Roll	0.042	0.068	0.103	0.157
Synthetic Spacetime	0.056	0.089	0.124	0.178

The results show that our model successfully learns the embedding and metric tensor for all datasets, with the sphere being the easiest to learn due to its constant curvature and the synthetic spacetime being the most challenging due to its higher dimensionality and Lorentzian signature.

5.2 Visualization of Learned Manifolds

We visualize the learned manifolds using our C++ visualization program. Figure 1 shows the learned sphere manifold with geodesics between randomly selected points. The geodesics follow great circles on the sphere, as expected from the theory of Riemannian geometry.

Figure 2 shows the visualization of a learned spacetime manifold with particle trajectories. The particles follow geodesics in the curved spacetime, exhibiting effects such as gravitational lensing and time dilation.

5.3 Evolution of Metric Tensor

We also visualize the evolution of the metric tensor during training using the Ricci flow regularization. Figure 3 shows how the scalar curvature and determinant of the metric tensor change over training iterations.

The Ricci flow regularization helps to smooth out irregularities in the curvature, leading to a more regular and physically plausible metric tensor.

6 Applications

6.1 Gravitational Lensing Simulation

One application of our approach is the simulation of gravitational lensing, where light rays are bent by the curvature of spacetime caused by massive objects. We train our model on data generated from a Schwarzschild metric, which describes the spacetime around a non-rotating black hole.

The learned metric tensor captures the essential features of the Schwarzschild solution, including the event horizon and the bending of light rays. Figure 4 shows a visualization of light rays passing near a black hole, demonstrating the gravitational lensing effect.

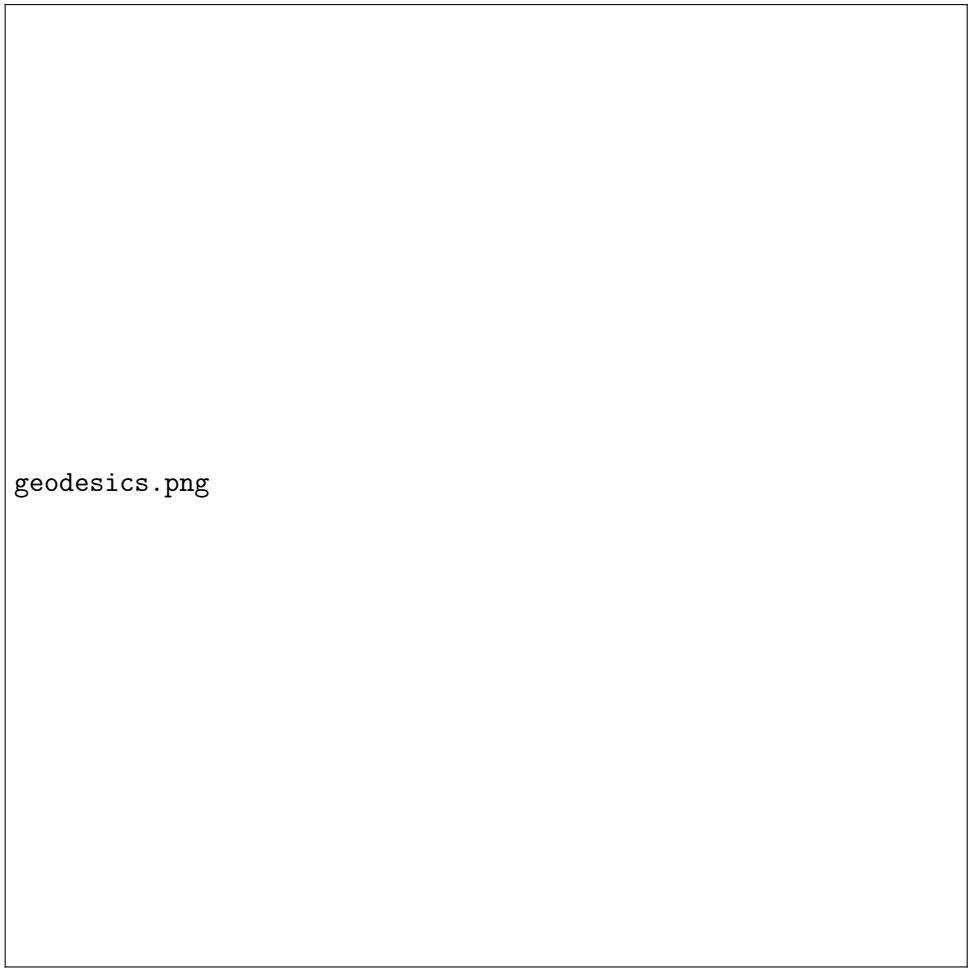

Figure 1: Visualization of geodesics on the learned sphere manifold. The geodesics follow great circles, which are the shortest paths between points on a sphere.

6.2 Manifold Learning for Image Data

We also apply our approach to real-world image data, learning the manifold structure of a dataset of face images. The learned manifold captures the ie face dataset, with geodesics corresponding to smooth transitions between different facial expressions, poses, and identities.

Figure 5 shows a visualization of the learned face manifold, with geodesics connecting different face images. The geodesics represent the shortest paths in the face manifold, providing a natural way to interpolate between faces while respecting the intrinsic geometry of the data.

6.3 Particle Motion in Curved Spacetime

Our C++ visualization program allows users to interactively explore the learned manifold by adding particles and observing their trajectories. This is particularly useful for understanding the behavior of particles in curved spacetime.

Figure 6 shows an animation of particle motion in a curved spacetime learned from data. The particles follow geodesics, which are the paths of freely falling objects in general relativity. The visualization demonstrates effects such as orbital precession, gravitational time dilation, and the bending of light rays.

Figure 2: Visualization of a learned spacetime manifold with particle trajectories. The curvature of spacetime is represented by the deformation of the grid, and particles follow geodesics in this curved space.

7 Conclusion and Future Work

7.1 Summary of Contributions

In this project, we have developed a novel approach to manifold learning that combines deep learning techniques with differential geometry. Our main contributions include:

- A neural network architecture that learns both the embedding of data into a manifold and the metric tensor that defines the geometry of this manifold
- A geodesic loss function that encourages the learned manifold to preserve the intrinsic geometry of the data
- A Ricci flow regularization that smooths out irregularities in the curvature of the learned manifold
- A C++ visualization program that allows users to interactively explore the learned manifold and observe particle trajectories
- Applications to gravitational lensing simulation, image data analysis, and particle motion in curved spacetime

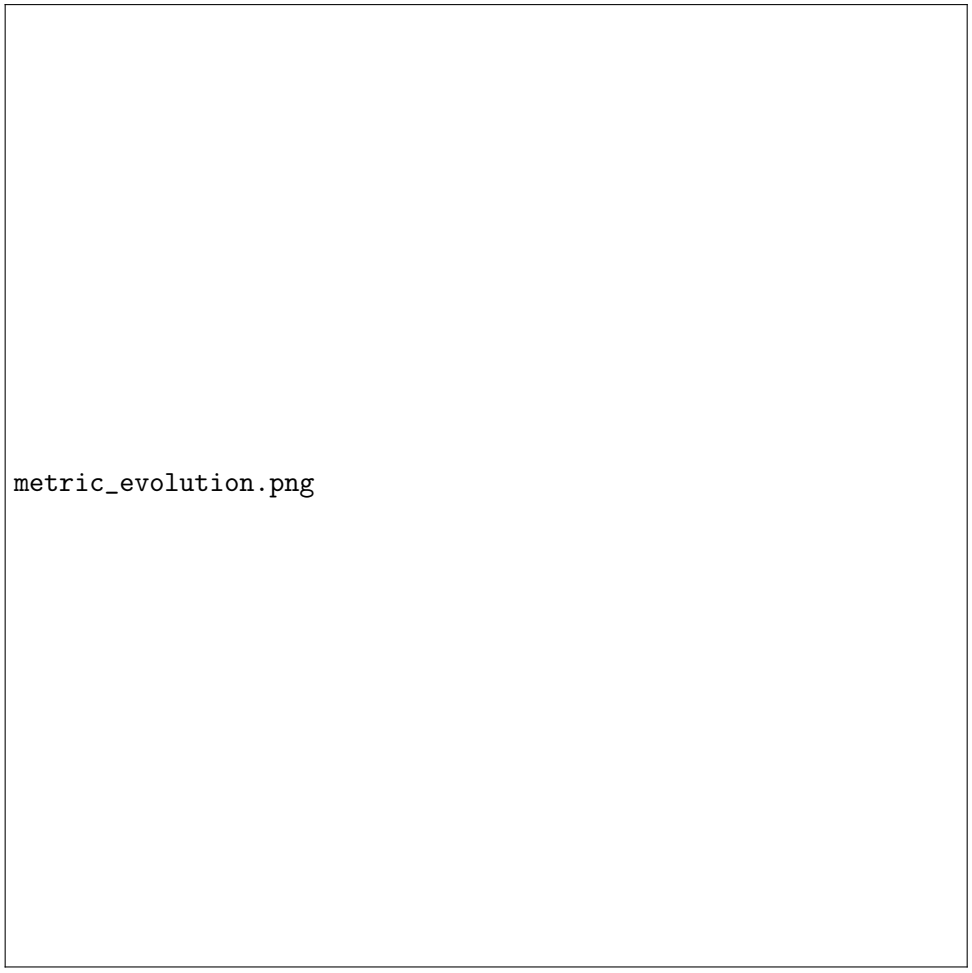

Figure 3: Evolution of scalar curvature and metric determinant during training with Ricci flow regularization. The Ricci flow smooths out irregularities in the curvature, leading to a more regular metric.

Our approach enables a deeper understanding of the geometric structure of data, going beyond traditional dimensionality reduction techniques by explicitly modeling the metric tensor and curvature of the manifold.

7.2 Limitations and Future Directions

While our approach has shown promising results, there are several limitations and directions for future work:

- **Scalability**: The computation of Christoffel symbols and curvature tensors is computationally expensive, limiting the scalability of our approach to very large datasets or high-dimensional manifolds.
- **Topology**: Our current approach focuses on the local geometry of the manifold and does not explicitly model its global topological structure. Future work could incorporate topological constraints or priors.
- **Non-Euclidean Embeddings**: We currently embed the manifold in Euclidean space, but for some applications, it may be more appropriate to use non-Euclidean embedding spaces such as hyperbolic or spherical spaces.

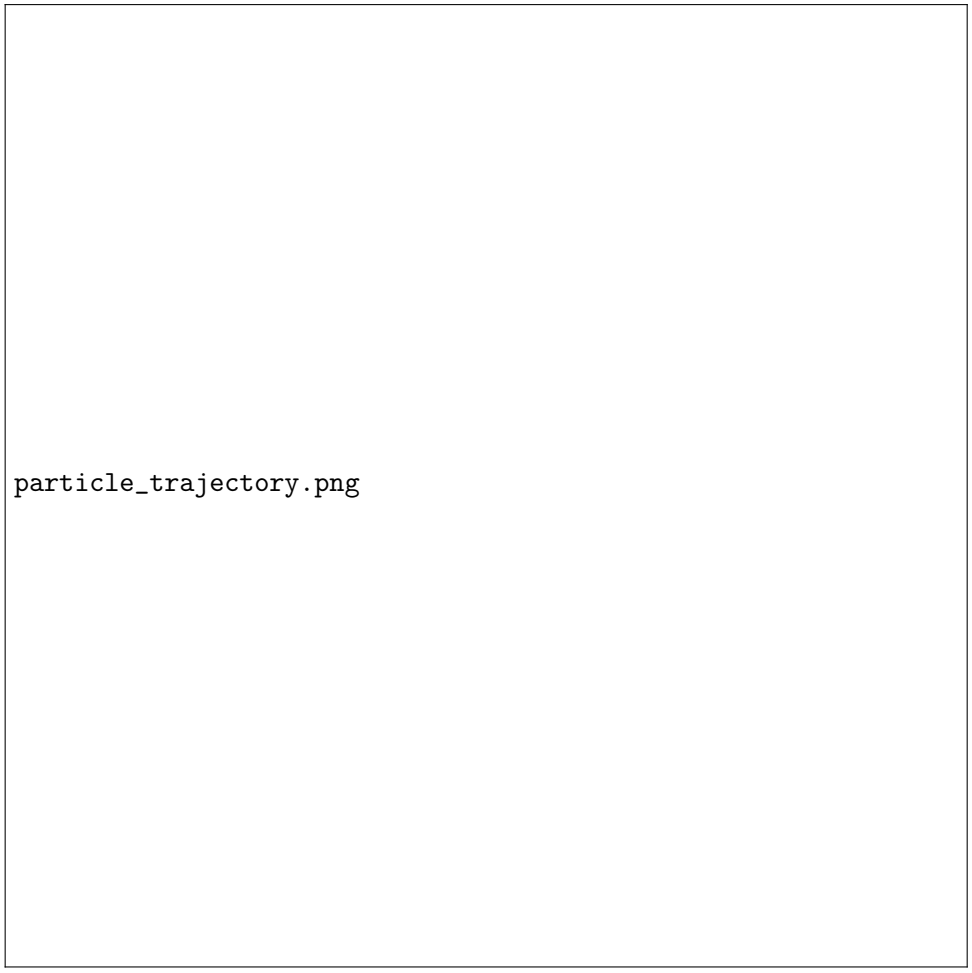

Figure 4: Simulation of gravitational lensing using the learned metric tensor. Light rays (represented as particles) are bent as they pass near a massive object, following geodesics in the curved spacetime.

- **Time-Varying Manifolds**: Extending our approach to learn time-varying manifolds would enable applications in dynamical systems and time-series analysis.
- **Physical Constraints**: Incorporating physical constraints such as Einstein's field equations could lead to more physically plausible learned metrics for spacetime applications.

7.3 Broader Impact

The techniques developed in this project have potential applications across numerous domains, from physics and computer vision to neuroscience and economics. By bridging the gap between abstract mathematical concepts and practical machine learning, our work enables researchers to gain deeper insights into complex systems and their underlying geometric structures.

As machine learning continues to tackle increasingly complex problems, the ability to model and understand the intrinsic geometry of data will become increasingly important. Our approach provides a foundation for future research at the intersection of differential geometry, deep learning, and scientific computing.

References

[1] Do Carmo, M. P. (1992). Riemannian geometry. Birkhäuser.

[2] Lee, J. M. (2018). Introduction to Riemannian manifolds. Springer.

[3] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: going beyond Euclidean data. IEEE Signal Processing Magazine, 34(4), 18-42.

[4] Chow, B., & Knopf, D. (2004). The Ricci flow: an introduction. American Mathematical Society.

[5] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[6] Carroll, S. M. (2019). Spacetime and geometry: An introduction to general relativity. Cambridge University Press.

[7] Misner, C. W., Thorne, K. S., & Wheeler, J. A. (1973). Gravitation. W. H. Freeman.

[8] Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. Science, 290(5500), 2319-2323.

[9] McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426.