# Geometric Flows and Dimension Reduction

Orton Babb  Aneesh Malhotra  Yemeen Ayub

George Mason University

## Introduction

Given a Riemannian manifold $(\mathcal{M}, g)$ with a specified metric $g$ the goal of minimal embedding is to find a smooth map $F : \mathcal{M} \to \mathbb{R}^m$ such that:

1. $F(\mathcal{M})$ Isometric
2. Has Minimal Extrinsic Curvature, i.e. to find:

$$F' = \underset{F}{\operatorname{argmin}} \underbrace{\int_{\mathcal{M}} ||\det HF(x)^{\perp}|| \, d\text{vol}}_{\text{Extrinsic Curvature}} \text{ subject to } \underbrace{DF(x)^{\top} DF(x) = g_x}_{\text{Preserve the geometry}}.$$

We can solve this analytically for simple cases. Consider the circle $S^1$ embedded into $\mathbb{R}^4$ via

$$\mathbf{F} = \frac{1}{\sqrt{1+k^2}} (\cos\theta, \sin\theta, \cos k\theta, \sin k\theta)$$

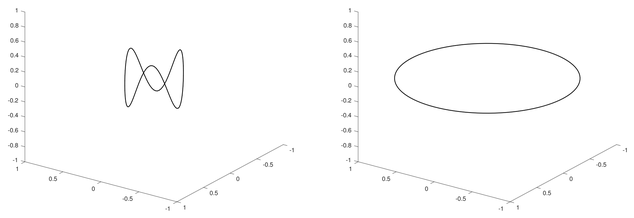


Figure 1: [First 3 coordinates] Our initial embedding of $S^1$ (left) and its solution, a 2D embedding (right)

## Goal

Consider a data set of $N$ points sampled from $\mathcal{M}$ given by $X = \{x_i\}_{i=1}^N$. We seek to perform a similar minimization on the extrinsic curvature of the data while preserving the geometry. The example we will work with is a discretization of the earlier example.

$$x_i = \frac{1}{\sqrt{1+k^2}} (\cos(\theta_i), \sin(\theta_i), \cos(k\theta_i), \sin(k\theta_i))$$

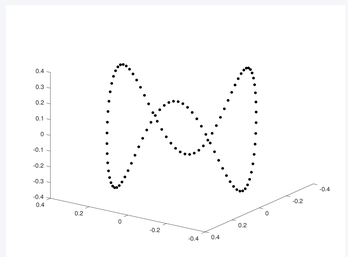


Figure 2: First three coordinates of our discretized data-set

## Diffusion Maps

Our method relies on being able to *parameterize* our data. In order to do this we will use the diffusion maps algorithm [1] to approximate eigenfunctions of the Laplacian $\{\phi_k\}_k$ which form a basis for $L^2(\mathcal{M})$ i.e. in the continuous case we would rewrite each coordinate of the embedding function as

$$f_j(z) = \sum_{k=1}^{\infty} C_{kj} U_k(z) \text{ with } z \in \mathcal{M}$$

### *Theorem*

*Following Coifman and Lafon, let* $k(k,y) = \exp\left(-\frac{\|x-y\|^2}{\epsilon^2}\right)$ *so that* $L_{ij} = \frac{k(x_i,x_j)}{\sum_j k(x_i,x_j)}$. *Then* $L\vec{f} \to \Delta f$ *as* $N \to \infty$ *and* $\epsilon \to 0$, *where* $\Delta$ *denotes the Laplace operator. The eigenfunctions of the Laplacian* $\Delta$ *can be approximated in the discrete case by the eigenvectors of the graph Laplacian* $L$.

## Approximating Eigenfunctions

Find the matrix of eigenvectors, $U$ such that $L = UVU^T$. This allows us to write our data as $X_i = \sum_k U_{ik} C_{k*}$ so $X = UC$.

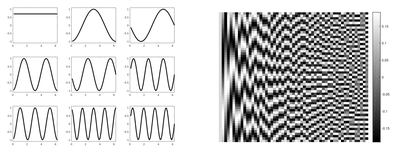


Figure 3: Continuous eigenfunctions of $S^1$ (left), Eigenvectors recovered from the data (right)

## Discrete Minimization Problem

- We will start with an initial choice for the Fourier coefficients $C_0 = U^t X$
- However, there are many choices of $C$ that will preserve the geometry
- We can use gradient descent to minimize extrinsic curvature along these $C$'s

Given the diagonal mat. $(D_X)_{ii} = \sum_k k(x_i, x_k)$, we have

$$\min \mathcal{L}(C, \Lambda) = \underbrace{\sum_{ij} (K_C)_{ij}}_{\text{Curvature estimate*, Q}} + \underbrace{\sum_{ij} \Lambda_{ij} (D_X^{-1} K_X - D_C^{-1} K_C)_{ij}}_{\text{Isometry Constraint, } I}$$

*The curvature estimate extends a result of Hein et al.

## Minimization by Gradient Descent

For speed and accuracy we first need to find $\nabla_C \mathcal{L}$ analytically.

Curvature estimate term:

$$\frac{\partial Q}{\partial c_{uv}} = \frac{-2c_{uv}}{\epsilon^2} \sum_{ij} (K_C)_{ij} Z_u^{ij}$$
$$Z_u^{ij} = (U_{iu} - U_{ju})^2$$

Isometry constraint:

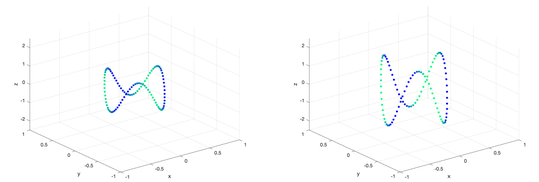
$$\frac{\partial I}{\partial c_{uv}} = \frac{-2c_{uv}}{\epsilon^2} \sum_{ij} \Lambda_{ij} (K_C)_{ij} \frac{1}{[\sum_r (K_C)_{ir}]^2} \times \left( \sum_r (K_C)_{ir} (Z_u^{ij} - Z_u^{ir}) \right)$$

## Algorithm 1. Minimize $\mathcal{L}(C, \Lambda)$ given update param. $\nu$

Our initial approach to cheaply minimizing the Lagrangian relies on gradient descent:

1. $X \leftarrow$ Input data
2. $C \leftarrow U^t X$
3. for $i = 1, ..., MaxIters$
4. $\quad C \leftarrow C - \nu \nabla_C \mathcal{L}$
5. $\quad \Lambda \leftarrow \Lambda - \nu (D_X^{-1} K_X - D_C^{-1} K_C)$

## Results



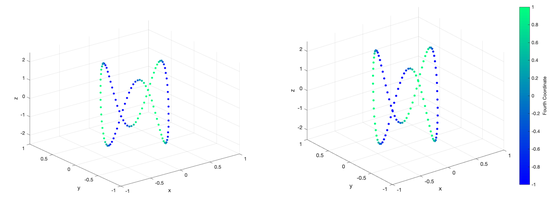


Figure 4: This method failed to enforce the constraint $D_X^{-1} K_X = D_C^{-1} K_C$

## Algorithm 2. Gradient Continuation given data $X$ and eigenvectors. $U$

In the future, we can try enforcing the constraint explicitly:

1. $C_1 \leftarrow U^t X$
2. for $k = 1, ..., MaxIters$
3. $\quad C_{k+1} \leftarrow C_k - \nu \nabla_C Q(C_k)$
4. $\quad$ Solve $D_X^{-1} K_X = D_C^{-1} K_C$ with init. cond. $C_{k+1}$ and replace

We expect this method to be cheaper than a Newton type method and easily implementable.

## References

[1] Ronald R Coifman and Stéphane Lafon.
Diffusion maps.
*Applied and Computational Harmonic Analysis*, 21(1):5--30, 2006.

[2] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg.
From Graphs to Manifolds -- Weak and Strong Pointwise Consistency of Graph Laplacians.
In Peter Auer and Ron Meir, editors, *Learning Theory*, pages 470--485, Berlin, Heidelberg, 2005. Springer.