Nonlinear Data Analysis

Tyrus Berry
George Mason University

December 11, 2015

Collaborators and Sponsors

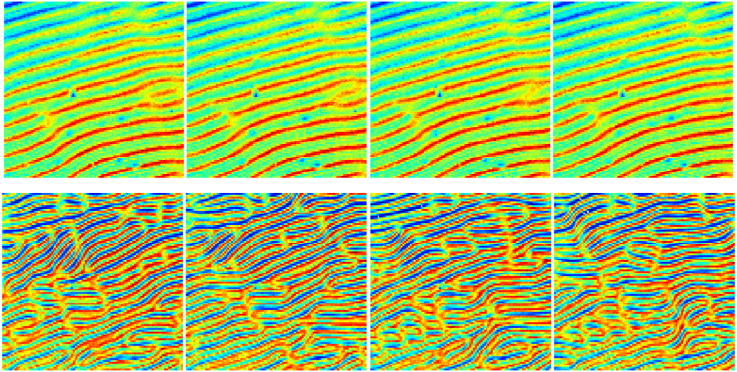
This presentation includes joint work with:

- Tim Sauer, George Mason University
- John Harlim, Penn State
- Dimitris Giannakis, Courant Institute

Work presented here was supported by NSF grant DMS-1250936 and ONR MURI grant N00014-12-1-0912

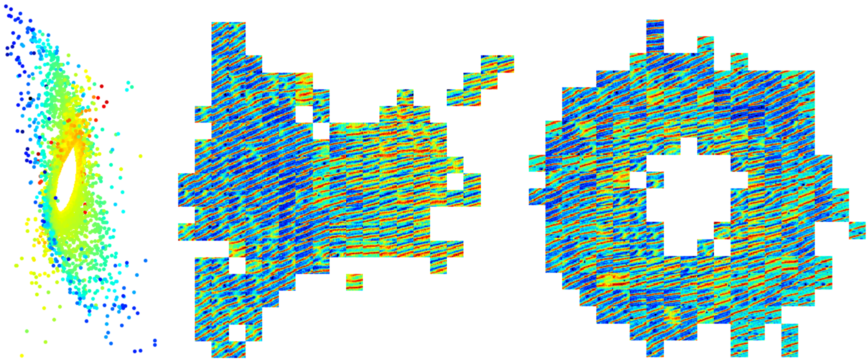
Low Dimensional Structure in High Dimensional Data

Example of High Dimensional Data:



Low Dimensional Structure in High Dimensional Data

The sub-image geometry:

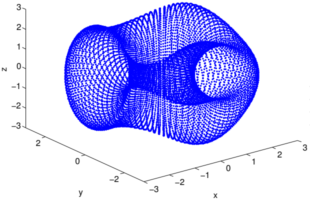


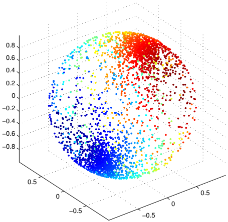
Overview/Key Points

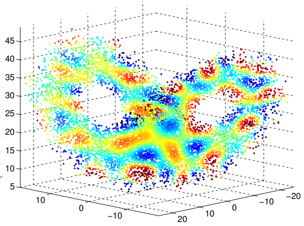
1. **Goal**: Learn geometric structure of data
2. **Tools:** Diffusion Maps and Local Kernels
3. **Applications:**
   - Smooth/Simplify Data
   - Understand Nonlinear Relationships
   - Feature Identification

The Geometric Assumption

- Data does not actually fill the high-dimensional data space







- Assume data are sampled from a manifold (curved subspace)
- **Goal:** Represent geometry via Laplacian operator

Why the Laplacian?

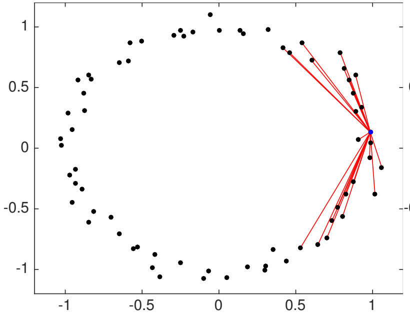
- **Key Fact:** Laplacian encodes all geometric information
- Laplacian generalizes calculus to manifolds

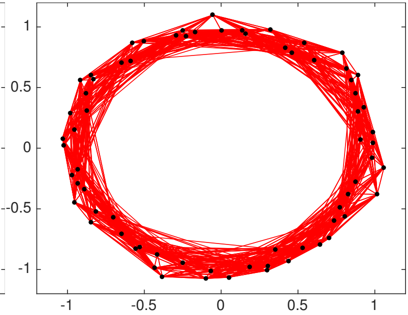
$$\Delta = \sum_{i,j} \frac{1}{\sqrt{|g|}} \partial_i \sqrt{|g|} (g^{-1})_{ij} \partial j$$

- On $\mathbb{R}^2$: $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial x^2}$
- On a circle: $\Delta = \frac{\partial^2}{\partial \theta^2}$
- On an ellipse: $\Delta = \frac{1}{\sqrt{a^2 \sin^2 \theta + b^2 \cos^2 \theta}} \frac{\partial}{\partial \theta} \left(\frac{1}{\sqrt{a^2 \sin^2 \theta + b^2 \cos^2 \theta}} \frac{\partial}{\partial \theta} \right)$

So how do we find the Laplacian from data?

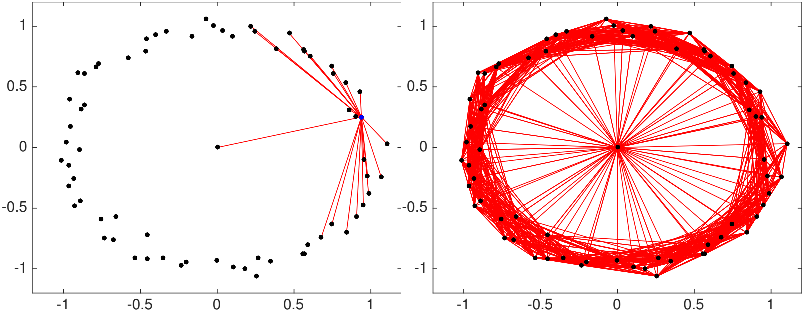
- Assume data lies on (or at least near) a manifold
- Approximate manifold with graph ⇒ Connect nearby points





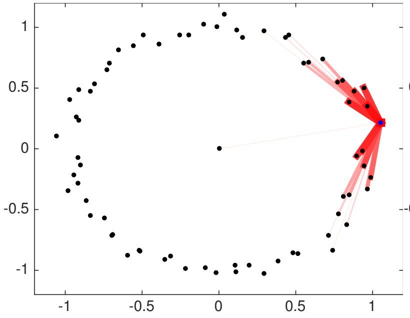
So how do we find the Laplacian from data?

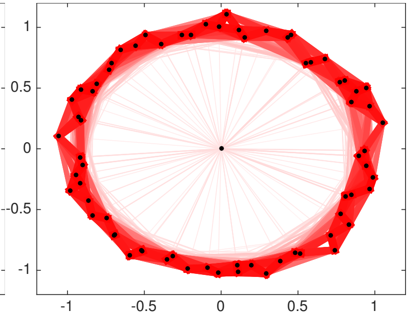
- **Problem:** Noise and outliers can lead to *bridging*



So how do we find the Laplacian from data?

- To prevent bridging we weight the edges
- Edges are given weights $K(x,y) = e^{-\frac{||x-y||^2}{4\epsilon}}$





So how do we find the Laplacian from data?

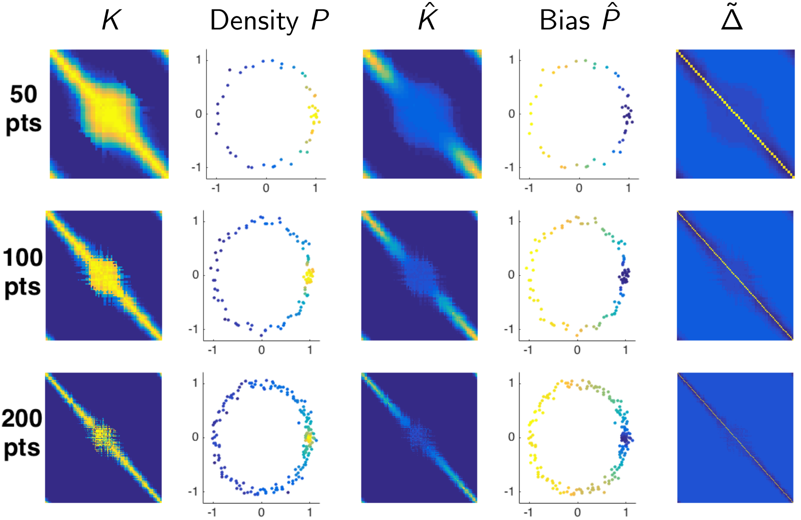
- We have converted our data set to a *weighted graph*
- Vertices $\Rightarrow$ Data points $\{x_1, x_2, ..., x_N\}$
- Edges $\Rightarrow$ Pairs of nearest neighbors
- Edge Weights $\Rightarrow K(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{4\epsilon}}$
- Represented as matrix $K_{ij} = K(x_i, x_j)$

Diffusion Maps: The Key Result

1. Start with the matrix $K_{ij} = e^{-\frac{||x_i - x_j||^2}{4\epsilon}}$
2. Find the row sums $P_i = \sum_{j=1}^{N} K(x_i, x_j)$
3. Normalize the matrix $\hat{K}_{ij} = \frac{K_{ij}}{P_i P_j}$
4. Find the row sums again $\hat{P}_i = \sum_{j=1}^{N} \hat{K}(x_i, x_j)$
5. Normalize again $\tilde{K}_{ij} = \frac{\hat{K}_{ij}}{\hat{P}_i}$
6. Form the Laplacian matrix $\tilde{\Delta} = \frac{I - \tilde{K}}{\epsilon}$

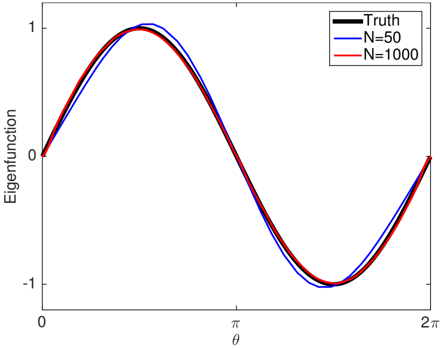
Theorem: As $N \to \infty$ and $\epsilon \to 0$ we have $\tilde{\Delta} \to \Delta$

Diffusion Maps Construction



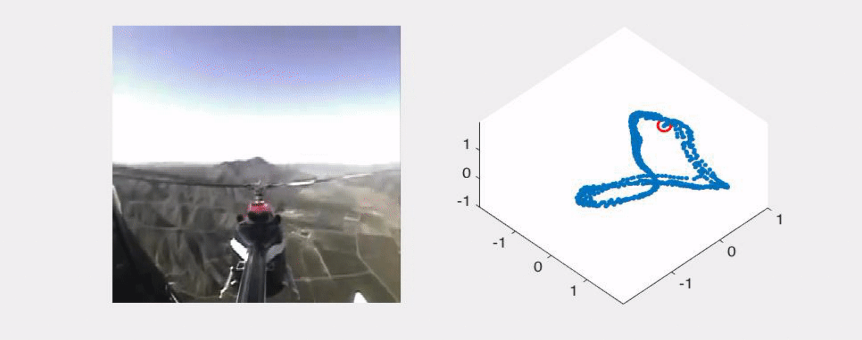
Diffusion Maps Construction

- $\tilde{\Delta}$ approximates the Laplacian Δ
- $\tilde{\Delta}$ encodes the geometry of the data
- Eigenvectors of $\tilde{\Delta}$ approximate eigenfunctions of Δ



Diffusion Maps for Video Data

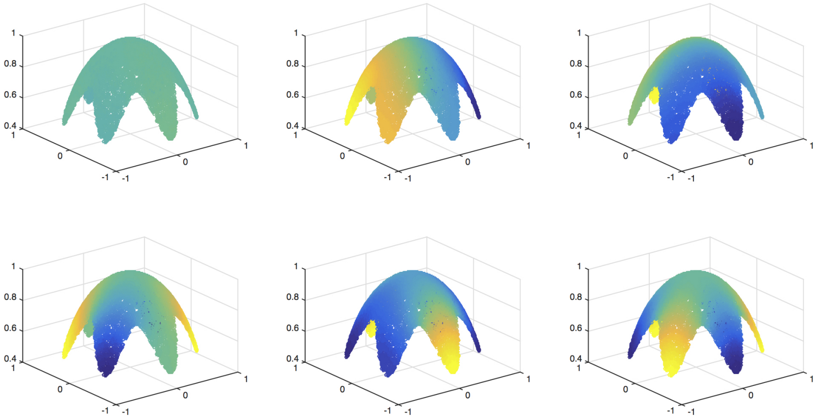
Eigenvectors of $\tilde{\Delta}$ give a low-dimensional representation:



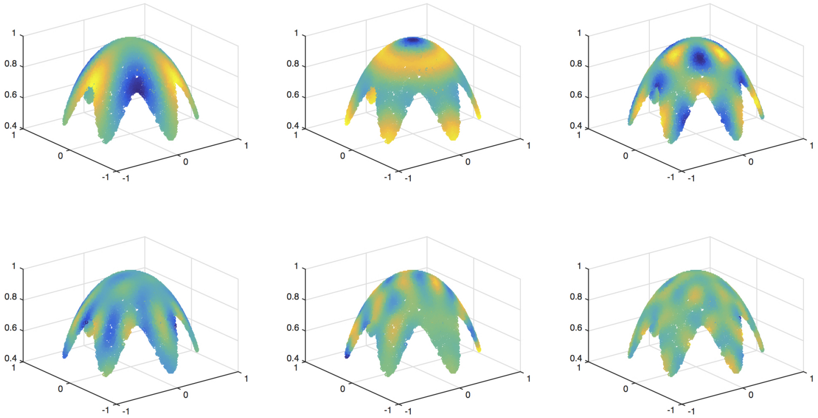
Fourier Basis on Manifolds

- Fourier functions $\sin(k\theta)$ are eigenfunctions of $\frac{d^2}{d\theta^2}$
- Eigenvectors of matrix $\tilde{\Delta}$ approximate eigenfunctions of Δ
- What is so great about these functions?
- Smoothest possible functions on $\mathcal{M}$
- $\varphi_0 =$ constant
- φ_1 contains a single oscillation
- φ_j is as smooth as possible and orthogonal to all previous

Fourier Basis on Manifolds



Fourier Basis on Manifolds



Using Fourier Basis to Smooth the Data

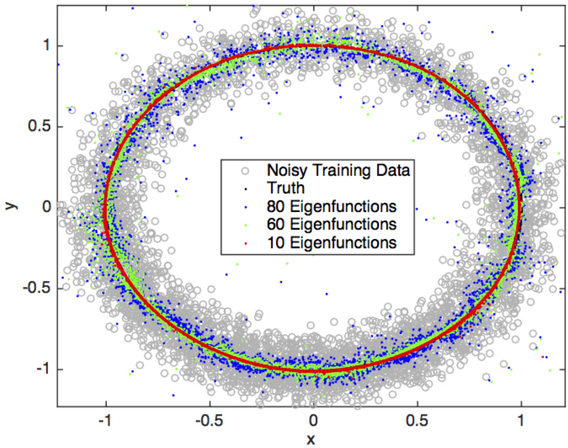
- Use generalized Fourier basis $\{\varphi_i\}$ to smooth data
- Project data into the basis:

$$c_j = \langle x, \varphi_j \rangle = \frac{1}{N} \sum_{k=1}^{N} x_k \varphi_j(x_k)$$

- Reconstruct smoothed data (low pass filter):

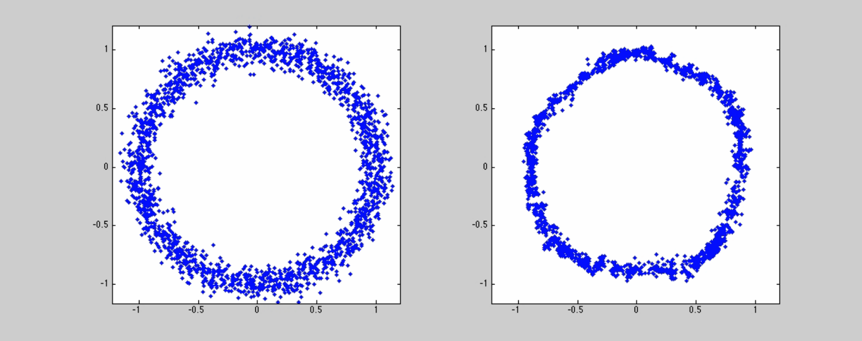
$$\tilde{x}_i = \sum_{j=1}^{L} c_j \varphi_j(x_i)$$

Using Fourier Basis to Smooth the Data



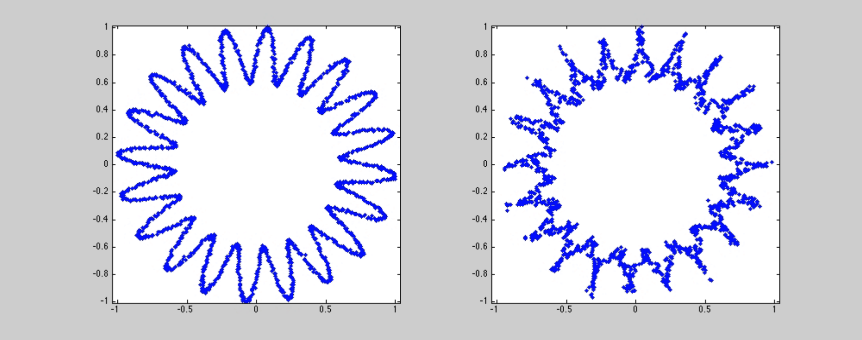
Using Fourier Basis to Smooth the Data

- Smooths noise:



Using Fourier Basis to Smooth the Data

- Smooths out the fine details of the geometry:



Local Kernels

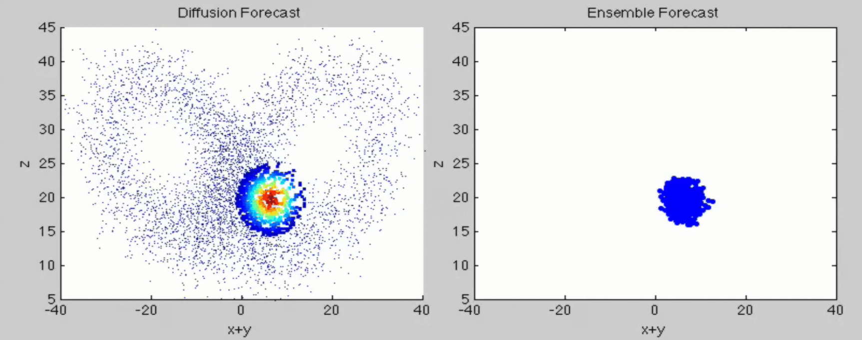
- A *local kernel* is a map $K : [0,\infty) \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$

$$K(\delta, x, x+\delta z) < ae^{-b||z||^2}$$

- Many ad hoc kernel methods exist in data science
  - Kernel Principal Component Analysis (KPCA)
  - Kernel Support Vector Machines (KSVM)
  - Kernel Density Estimation (KDE)
  - Spectral Clustering
  - Reproducing Kernel Hilbert Spaces (RKHS)
- Almost all kernels in use are local kernels
- **Theorem:** Every local kernel defines a geometry

Example: Forecasting without a Model

No Model | Perfect Model



Forecasting without a Model

$$
\begin{array}{ccc}
p(x,t) & \xrightarrow{\text{Nonparametric Forecast}} & p(x,t+\tau) \\
\Big\downarrow \langle p,\varphi_j\rangle & & \Big\uparrow \sum_j c_j \varphi_j p_{\mathrm{eq}} \\
\vec{c}(t) & \xrightarrow{A_{lj}\equiv\mathbb{E}[\langle\varphi_j,S\varphi_l\rangle_{p_{\mathrm{eq}}}]} & \vec{c}(t+\tau) = A\vec{c}(t).
\end{array}
$$

- $\vec{c}(t)$ are the generalized Fourier coefficients of p
- Nonlinear dynamics become linear (matrix A) in this basis

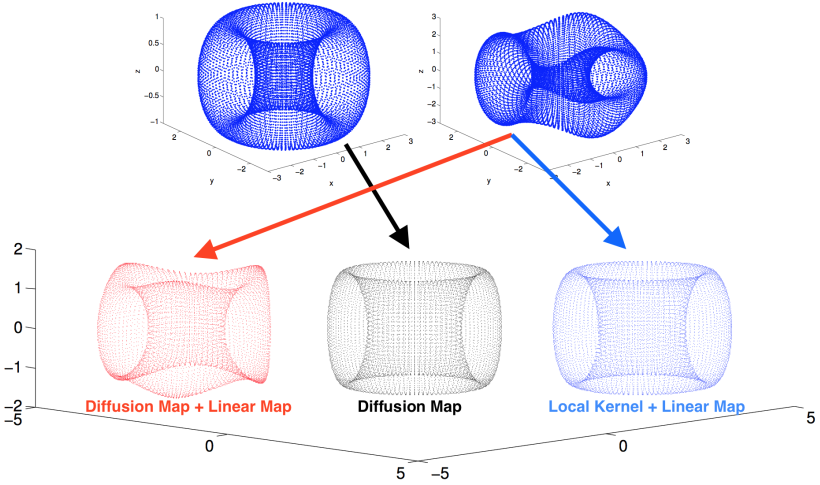
Learning Nonlinear Maps

- Assume we have two data sets $\{x_i\}_{i=1}^N$ and $\{y_i\}_{i=1}^N$
- Related by nonlinear map $y_i = \mathcal{H}(x_i)$

$$
\begin{array}{ccc}
\mathcal{M} & \xrightarrow{\quad \mathcal{H} \quad} & \mathcal{H}(\mathcal{M}) \\
\downarrow \tilde{\Phi} & & \downarrow \Phi \\
L^2(\mathcal{M}, \tilde{g}) \approx \mathbb{R}^{\hat{n}} & \xrightarrow{\quad U \quad} & L^2(\mathcal{H}(\mathcal{M}), g) \approx \mathbb{R}^{\hat{m}}
\end{array}
$$

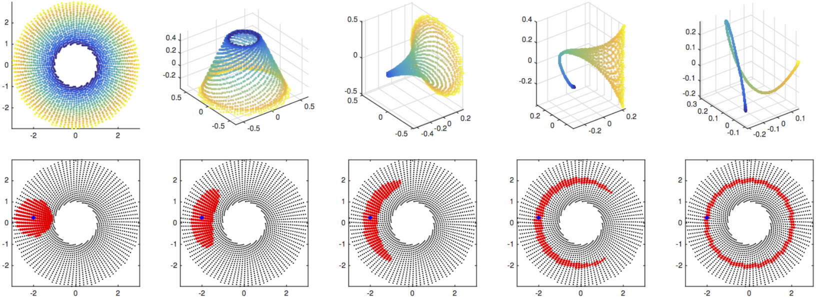
- $\tilde{\Phi}$ and Φ are built with Local Kernels
- U is linear $\Rightarrow$ Easy to fit

Learning Nonlinear Maps



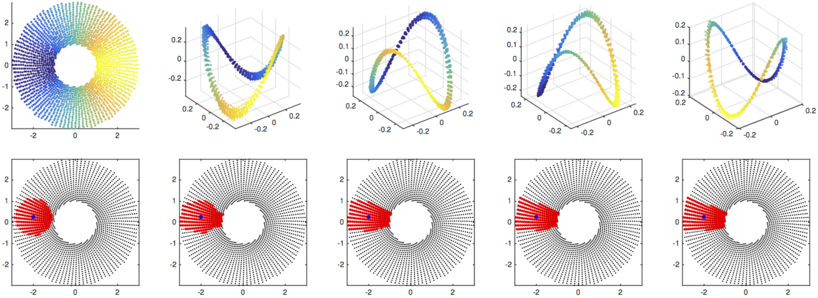
Feature Identification

Iterated Diffusion Map (IDM) isolates a feature of interest (radius)



Feature Identification

Iterated Diffusion Map (IDM) isolates a feature of interest (angle)



Key Points

1. **Goal**: Learn geometric structure of data
2. **Tools:** Diffusion Maps and Local Kernels
3. **Applications:**
    - **Geometry** $\Rightarrow$ Custom Fourier basis
        - Smooth/Simplify data
    - **Fourier Basis** $\Rightarrow$ Nonlinear relationships become linear
        - Forecast operator becomes linear
        - Nonlinear maps between data sets become linear
    - **Understanding Geometry** $\Rightarrow$ Feature identification
        - Feature identification via Iterated Diffusion Map (IDM)
        - Learn the dimension, volume, and other topological features

For more information: http://math.gmu.edu/~berry/

Building the basis

- Coifman and Lafon, *Diffusion maps.*
- Berry and Harlim, *Variable Bandwidth Diffusion Kernels.*

Nonparametric forecast

- Berry, Giannakis, and Harlim, *Nonparametric forecasting of low-dimensional dynamical systems.*
- Berry and Harlim, *Semiparametric forecasting and filtering: correcting low-dimensional model error in parametric models.*

Nonlinear Maps and Feature Identification

- Berry and Sauer, *Local Kernels and the Geometric Structure of Data.*
- Berry and Harlim, *Iterated Diffusion Maps for Feature Identification.*